T.C.
İstanbul
YENİ YÜZYIL ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİYOMEDİKAL MÜHENDİSLİĞİ ANABİLİM DALI

Beyin Tümörü Teşhisinde Kullanılan Yapay Zeka Yöntemlerinin Araştırılması ve Karşılaştırılması

YÜKSEK LİSANS TEZİ

Gizem AKGÖNÜL

Tez Danışmanı
Dr. Öğretim Üyesi Diyadin CAN

İSTANBUL
Ağustos-2024

Etik Beyan

İstanbul Yeni Yüzyıl Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;
• Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
• Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
• Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
• Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
• Bu tezde sunduğum çalışmanın özgün olduğunu,
bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

08/2024
Gizem AKGÖNÜL

Teşekkür

Tezimin hazırlanması sırasında yardımını esirgemeyen tez yöneticisi hocam, Dr. Öğretim Üyesi Diyadin Can'a , tez çalışmalarım esnasında, yapay zeka algoritmaları konusunda yardımını ve desteğini esirgemeyen ve bana örnek teşkil eden arkadaşım Berke Karaduman'a ve canım aileme çok teşekkür ederim.

İÇİNDEKİLER

Sembol ve Kısaltma Listesi

BT	Bilgisayarlı Tomografi / Computed Tomography (CT).
MR / MRG	Manyetik Rezonans (Görüntüleme) / Magnetic Resonance (Imaging, MRI).
fMRG / fMRI	Fonksiyonel Manyetik Rezonans Görüntüleme / functional MRI.
RF	Radyo Frekansı (Radio Frequency).
3B / 4B	Üç Boyutlu / Dört Boyutlu (3D/4D).
CNN	Konvolüsyonel Sinir Ağları / Convolutional Neural Networks.
RNN	Tekrarlayan Sinir Ağları / Recurrent Neural Networks.
GAN	Üretici Çekişmeli Ağlar / Generative Adversarial Networks.
SVM	Destek Vektör Makineleri / Support Vector Machines.
SVR	Destek Vektör Regresyonu / Support Vector Regression.
RBF	Radial Basis Function çekirdeği (SVM bağlamında).
DBSCAN	Density-Based Spatial Clustering of Applications with Noise.
PCA	Ana Bileşen Analizi / Principal Component Analysis.
ICA	Türetilmiş (Bağımsız) Bileşen Analizi / Independent Component Analysis.
t-SNE	t-distributed Stochastic Neighbor Embedding.
MAE	Ortalama Mutlak Hata / Mean Absolute Error.
MSE	Ortalama Kare Hatası / Mean Squared Error.
RMSE	Kök Ortalama Kare Hatası / Root Mean Squared Error.
F1	F1 skoru (kesinlik-geri çağırma harmonik ortalaması).
GPU	Graphics Processing Unit (TensorFlow bağlamında hızlandırma).
API	Application Programming Interface (Keras/TensorFlow bağlamında).
TFX	TensorFlow Extended.
WHO	Dünya Sağlık Örgütü (World Health Organization).
NF1	Nörofibromatozis Tip 1.
VHL	Von Hippel–Lindau sendromu.
AVM	Arteriovenöz Malformasyon(lar).
AI / YZ	Artificial Intelligence / Yapay Zeka.

FDA	Food and Drug Administration (ABD Gıda ve İlaç Dairesi).

Şekil ve Grafik Listesi

Tablo Listesi

ÖZET

Beyin tümörlerinin teşhisi ve sınıflandırılması, tedavi sürecinde kritik bir rol oynar. Bu tez, beyin tümörlerinin tespitinde ve sınıflandırılmasında yapay zeka yöntemlerinin etkinliğini araştırmayı ve karşılaştırmayı amaçlamaktadır. Beyin tümörleri, normal hücre kontrol mekanizmalarının dışında büyüyen ve çoğalan anormal hücre kümeleridir. Geleneksel teşhis yöntemleri invaziv olup zaman alıcıdır ve hata payı yüksektir. Bu nedenle, Manyetik Rezonans Görüntüleme (MRG) kullanılarak tümörlerin radyomik ve genomik özelliklerinin belirlenmesi önem kazanmıştır.

Tezde, beyin tümörlerinin 3 boyutlu MRG görüntüleri kullanılarak çeşitli yapay zeka algoritmaları ile sınıflandırılması gerçekleştirilmiştir. Çalışma, konvolüsyonel sinir ağları (CNN) ve tekrarlayan sinir ağları (RNN) gibi modern makine öğrenme yöntemlerinin etkinliğini değerlendirmektedir. Her bir algoritmanın doğruluk oranı, hız, verimlilik, aşırı öğrenme eğilimleri ve bellek kullanımı gibi performans kriterleri karşılaştırılmıştır.

Sonuçlar, CNN ve RNN modellerinin beyin tümörü sınıflandırmasında yüksek doğruluk oranlarına sahip olduğunu göstermektedir. Özellikle CNN'ler, görüntü işleme ve sınıflandırma konularında üstün performans sergilemiştir. RNN'ler ise zaman serisi verilerinde güçlü sonuçlar vermiştir. Destek vektör makineleri (SVM) ve rastgele ormanlar gibi diğer yöntemler de belirli senaryolarda etkili olmuştur ancak genelleme yetenekleri sınırlıdır.

Bu çalışma, yapay zeka tabanlı yöntemlerin beyin tümörü teşhis ve sınıflandırmasında potansiyelini ortaya koymakta ve klinik uygulamalarda kullanımını desteklemektedir. Gelecekteki araştırmalar, daha büyük ve çeşitli veri setleri ile bu yöntemlerin etkinliğini artırmayı hedeflemelidir.

Anahtar kelimeler: *Yapay Zeka, Beyin Tümörleri, Derin Öğrenme, Makine Öğrenmesi, Konvolüsyonel Sinir Ağları, Görüntü İşleme*.

ABSTRACT

The diagnosis and classification of brain tumors play a critical role in the treatment process. This thesis aims to investigate and compare the effectiveness of artificial intelligence methods in the detection and classification of brain tumors. Brain tumors are clusters of abnormal cells that grow and proliferate uncontrollably outside the mechanisms that control normal cells. Traditional diagnostic methods are invasive, time-consuming, and have a high margin of error. Therefore, it has become important to determine the radiomic and genomic features of tumors using Magnetic Resonance Imaging (MRI).

In this thesis, the classification of brain tumors was performed using various artificial intelligence algorithms on 3D MRI images. The study evaluates the effectiveness of modern machine learning methods such as convolutional neural networks (CNN) and recurrent neural networks (RNN). Each algorithm's performance was compared based on criteria such as accuracy, speed, efficiency, overfitting tendencies, and memory usage.

The results show that CNN and RNN models have high accuracy rates in brain tumor classification. CNNs, in particular, demonstrated superior performance in image processing and classification tasks. RNNs also provided strong results in time series data. Other methods like support vector machines (SVM) and random forests were effective in certain scenarios but had limited generalization capabilities.

This study reveals the potential of AI-based methods in the diagnosis and classification of brain tumors and supports their use in clinical applications. Future research should aim to enhance the effectiveness of these methods with larger and more diverse datasets.

Keywords: *Artificial Intelligence, Brain Tumors, Deep Learning, Machine Learning, Convolutional Neural Networks, Image Processing*.

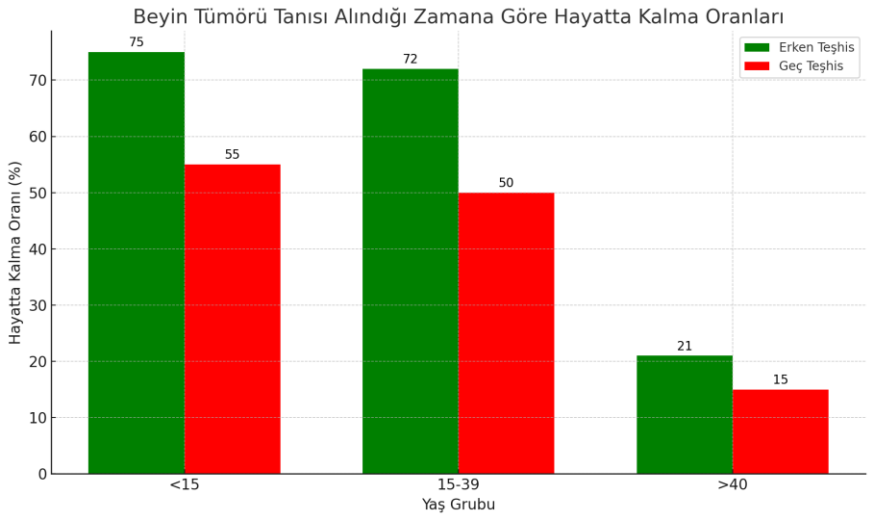
1. GİRİŞ

1.1 Başlangıçtaki durum ve motivasyon

Dünya Sağlık Örgütü'nün son verilerine göre, beyin tümörleri dünya çapında en yaygın kanser ölüm nedenlerinden biridir ve her yaşta ortaya çıkabilir. Her yaşta görülebilir, ancak özellikle 40 yaşın üstündeki kişilerde daha fazla ölüme neden olur. Beyin tümörlerinin erken teşhisi tedavi için büyük önem taşımaktadır. Ayrıca, hastaların hayatta kalma oranı ve erken evrede teşhis edilen hastaların sayısı önemli ölçüde artırılabilir. Bu nedenle görüntülerin hızlı ve doğru bir şekilde değerlendirilmesi, beyin tümörlerinin tespiti ve tedavi planlaması için çok önemlidir [2][37].

Yapay zeka (AI), özellikle makine öğrenimi ve derin sinir ağları, bu zorlukların üstesinden gelme potansiyeline sahiptir. Yapay zeka sistemleri, insan analistlerin tanımlaması zor olan büyük veri kümelerindeki karmaşık kalıpları ve anormallikleri tanıyabilir. Bu yetenek, yapay zekayı MR görüntü analizinde, özellikle de patolojik özelliklerin tespiti ve sınıflandırılmasında değerli bir araç haline getirmektedir.

MR görüntü analizinde önceki yapay zeka yaklaşımları önemli ilerlemeler kaydetmiştir. Örneğin görüntü kalitesinin iyileştirilmesi, görüntü segmentasyonunun otomatikleştirilmesi ve görüntü segmentasyonu ve belirli hastalıkların teşhisine destek sağlamıştır. Bu başarılara rağmen, mevcut yapay zeka yöntemleri, özellikle teşhis doğruluğu ve verimliliği açısından sınırlarına ulaşmaktadır. Temel sorunlardan biri, birçok yapay zeka modelinin belirli kullanım durumları için eğitilmiş olmasıdır ve performanslarının diğer hastalık modellerine veya farklı klinik ortamlara kolayca aktarılmasıdır. Bu sınırlama kısmen eğitim için mevcut kapsamlı ve çeşitlendirilmiş veri setlerinin eksikliğinden kaynaklanmaktadır [10][35].



Şekil 1: Beyin tümöründe tanı zamanına göre hayatta kalma oranları

Beyin tümörlerinin teşhisinde kullanılan yapay zeka yöntemleri, yüksek hassasiyet ve hız sağlayarak klinik karar alma süreçlerini iyileştirmeyi amaçlamaktadır. Bu tez, beyin tümörü teşhisinde yapay zeka teknolojilerinin kullanımını araştırmayı ve farklı yapay zeka yöntemlerini karşılaştırmayı hedeflemektedir [39].

1.2. Hedef ve yapı

Bu tezin amacı, beyin tümörlerinin teşhisinde ve sınıflandırılmasında yapay zeka tabanlı yöntemlerin etkinliğini araştırmak ve bu yöntemlerin performansını karşılaştırmaktır. Yapay zeka, özellikle makine öğrenimi ve derin öğrenme algoritmaları, büyük veri kümelerindeki karmaşık kalıpları ve anormallikleri tanıma konusunda insan analizine kıyasla büyük bir potansiyele sahiptir. Bu potansiyelin, beyin tümörü teşhis süreçlerinde nasıl kullanılabileceği ve mevcut yöntemlere göre avantajları bu çalışmanın temel motivasyonunu oluşturmaktadır. Bu bağlamda, çalışmanın hedefleri arasında beyin tümörlerinin tespiti ve sınıflandırılmasında kullanılan mevcut yapay zeka yöntemlerini detaylı bir şekilde incelemek, Manyetik Rezonans Görüntüleme (MRG) verileri kullanarak farklı makine öğrenimi algoritmalarını uygulamak ve performanslarını karşılaştırmak, Konvolüsyonel Sinir Ağları (CNN) ve Tekrarlayan Sinir Ağları (RNN) gibi derin öğrenme modellerinin beyin tümörü teşhisindeki etkinliğini değerlendirmek, elde edilen sonuçları doğruluk, hız, verimlilik ve aşırı öğrenme gibi kriterler açısından analiz etmek ve yapay zeka tabanlı teşhis yöntemlerinin klinik uygulamalarda kullanılabilirliğini artırmak için önerilerde bulunmak yer almaktadır.

Bu tez altı ana bölümden oluşmaktadır: Giriş bölümünde çalışmanın amacı, kapsamı ve motivasyonu açıklanmaktadır. Beyin ve Tümör ile İlgili Genel Bilgiler bölümünde beyin tümörlerinin tanımı, türleri, belirtileri ve tedavi yöntemleri hakkında bilgi verilmektedir. Yapay Zeka ile İlgili Temel Kavramlar bölümünde yapay zeka, makine öğrenimi ve derin öğrenme kavramları açıklanmaktadır. Yöntemler bölümünde çalışmada kullanılan veri setleri, algoritmalar ve deneysel düzenekler detaylandırılmaktadır. Sonuçlar ve Tartışma bölümünde elde edilen bulgular analiz edilmekte ve yorumlanmaktadır. Sonuç ve Öneriler bölümünde ise çalışmanın genel sonuçları özetlenmekte ve gelecekteki araştırmalar için öneriler sunulmaktadır. Bu yapı, tezde ele alınan konuların sistematik bir şekilde incelenmesini ve elde edilen sonuçların açıkça sunulmasını sağlamaktadır [11].

2. GENEL BİLGİLER

2.1 YAPAY ZEKA İLE İLGİLİ TEMEL KAVRAMLAR

Bu bölümde, yapay zeka operasyonlarında yaygın olarak kullanılan temel kavramlar olan yapay sinir ağı, görüntü işleme ve makine öğrenimi açıklanmaktadır. Ayrıca, bu süreçlerde kullanılan yazılım dili, kütüphaneler ve sitelerden bahsedilecektir. Son olarak, veri setinin oluşturulması ve eğitilmesi için kullanılan yerler ve programlar detaylı olarak açıklanacaktır.

2.1.1 Yapay Zeka

Yapay zeka terimi, resmi olarak ilk olarak, 1955 yılında Dartmouth College'daki konferansta John McCarthy tarafından kullanılmıştır. Yapay Zeka alanı makinelerin kendi başlarına öğrenmek, karar almak ve düşünmek gibi eylemleri yapabilmelerini amaçlayan bir uygulama türüdür [21]. Teknolojik gelişme ile birlikte bu konuda araştırmalar yapılırken ve çeşitli çalışmalar meydana gelmektedir. Bu çalışmaların sonucunda bu yen modeller doğmaktadır. Bu modellerden biri, makine öğrenmesidir. Bu modelde, veriler algoritmalar yardımıyla analiz edilir. Daha sonra model bu verilerle eğitilir. Son adımda ise yeni verileri bu bilgilerle analiz eder. Bu model birçok sistem ve makineye entegre edilmiştir. Grafik kartları ve işlemcilerdeki gelişmeler sayesinde yeni ve farklı makine öğrenmesi ve veri işleme teknikleri geliştirilmiştir. Bu tekniklerden en önemlilerinden biri, yapay sinir ağıdır. Bu teknikte daha karmaşık ve detaylı sistemler tasarlanır. Aynı zamanda bu sistemlerle yeni yapay zeka modelleri geliştirilir. Bu modellerden biri de Derin Öğrenme'dir. Derin Öğrenme, makine öğrenmesinin bir alt dalıdır. Derin Öğrenme görüntüler ve sesler gibi karmaşık veri yapılarını işleyebilir. Bu sayede, nesne tanıma ve nesne tespiti gibi birçok alanda çalışmalara katkı sağlamıştır. (LeCun et al., 2015)

Şekil 2: Yapay Zeka-Makine Öğrenmesi-Derin Öğrenme ilişkisi

2.1.2. Makine Öğrenimi

Makine öğrenmesi, yapısal ve işlevsel olarak öğrenmeye açık, yorumlama yeteneğine sahip, verileri anlayan ve sonuçları ortaya çıkaran bir sistemdir. Bu sistem, kendi modelini oluşturarak çalışır. Giriş verilerini belirli bir algoritmaya göre işleyerek ve çıktı parametresi ile istatistiksel tahminlerde bulunarak doğru tahminler yapmaya çalışır. Makine öğrenmesine ve veri madenciliğine olan ilginin artması bu alanın öneminin büyümesine neden olmuştur. Artan veri türleri ve hacimleri, daha ucuz ve daha güçlü işlem yeteneklerinin kullanılması ve verilerin ekonomik olarak depolanması, makine öğrenmesinin öneminin artmasına katkıda bulunmuştur.

2.1.2.1 Denetimli Öğrenme

Denetimli öğrenme, makine öğrenmesi alanında en yaygın kullanılan tekniklerden biridir. Bu yöntem, bilinen verilerden (etiketli veriler) öğrenerek bilinmeyen veriler üzerinde tahmin yapmayı amaçlar. Temel olarak, denetimli öğrenme algoritmaları bir eğitim kümesi üzerinden model oluşturur ve bu modeli kullanarak yeni ve bilinmeyen verilere ilişkin tahminlerde bulunur. Denetimli öğrenmenin en önemli avantajlarından biri, modelin karmaşık veri yapıları ve ilişkileri üzerinde yüksek doğrulukta tahminler yapabilmesidir [8]. Ayrıca, denetimli öğrenme yöntemleri, geniş veri kümeleri üzerinde verimli bir şekilde çalışabilir ve sonuçları hızlı bir şekilde verebilir.

Denetimli öğrenme, iki ana bileşenden oluşur: giriş verileri (input) ve çıkış etiketleri (output). Giriş verileri, modele sunulan özellikler kümesidir, çıkış etiketleri ise her giriş verisi için beklenen

sonuçları temsil eder. Bu veriler kullanılarak modelin eğitilmesi, modelin, verilen bir girdi için doğru sonucu tahmin etme yeteneğini geliştirir. Genellikle veri setler eğitim ve test olmak üzere veri ikiye ayrılır. Modelin öğrenmesi için kullanılan verilerden oluşurken verilere eğitim veri seti denir. Test veri seti ise modelin genelleme yeteneğini ve doğruluğunu değerlendirmek için kullanılır. Eğitim sürecinde model, giriş verileri ile çıkış etiketleri arasındaki ilişkiyi öğrenir ve bu ilişkiyi kullanarak test verileri üzerinde tahminler yapar. Test verileri üzerindeki tahminlerin doğruluğu modelin performansını belirler [43].

Sınıflandırma, denetimli öğrenme algoritmalarının verileri belirli sınıflara ayırmak için kullandığı bir yöntemdir. Sınıflandırma problemlerinde, modelin amacı, giriş verilerine bakarak bunların hangi sınıfa ait olduğunu belirlemektir. Yaygın sınıflandırma algoritmaları arasında karar ağaçları, destek vektör makineleri (SVM), k-en yakın komşu (k-NN) ve lojistik regresyon bulunur. Karar ağaçları, veri sınıflandırmak için dallara ayrılan ve karar kurallarına dayanan bir model kullanır. Destek vektör makineleri, veriyi en iyi şekilde ayıran bir hiper düzlem bulmaya çalışır. K-en yakın komşu algoritması, bir veri noktasının sınıfını belirlemek için en yakın k komşusuna bakar. Lojistik regresyon ise olasılık tabanlı bir yaklaşımla veri noktalarının sınıflarını tahmin eder [42].

Regresyon, denetimli öğrenme algoritmalarının sayısal bir çıktıyı tahmin etmek için kullanıldığı bir yöntemdir. Regresyon problemlerinde, modelin amacı, giriş verileri kullanılarak sürekli bir değişkenin değerini tahmin etmektir. Yaygın regresyon algoritmaları arasında doğrusal regresyon, polinom regresyon ve destek vektör regresyonu (SVR) bulunur. Doğrusal regresyon, bağımsız ve bağımlı değişkenler arasında doğrusal bir ilişki kurar. Genellikle doğrusal olmayan ilişkileri modellemek için polinom regresyon kullanılır. Doğrusal regresyonun destek vektör makineleri ile genelleştirilmiş haline destek vektör regresyonu denir. Regresyon analizinde kullanılan modeller, bağımsız değişkenlerin bağımlı değişken üzerindeki etkisini tahmin etmek için kullanılır ve bu sayede gelecekteki olaylar hakkında öngörülerde bulunmak mümkün olur [14].

Denetimli öğrenme modellerinin başarısı, çeşitli değerlendirme parametreleri kullanılarak ölçülür. Bu parametreler modelin tahmin performansını ve genelleme yeteneğini değerlendirmek için kullanılır. Doğruluk, modelin doğru tahmin ettiği örneklerin toplam örneklere oranını gösterir. Kesinlik, doğru pozitif tahminlerin, toplam pozitif tahminlere oranını ifade ederken, geri çağırma, doğru pozitif tahminlerin, gerçek pozitif örnekler içindeki oranını ifade eder. F1 skoru, kesinlik ve geri çağırmanın harmonik ortalamasını alarak modelin genel performansını tek bir değer ile özetler. Regresyon problemlerinde yaygın olarak kullanılan hata metrikleri arasında ortalama mutlak hata (MAE), ortalama kare hatası (MSE) ve kök ortalama kare hatası (RMSE) bulunur. Ortalama mutlak hata (MAE) gerçek değerler ile tahmin edilen değerler arasındaki farkların ortalamasıdır. Ortalama kare hatası (MSE) tahmin edilen değerler ile gerçek değerler arasındaki farkların karelerinin ortalamasıdır. RMSE ise MSE'nin kareköküdür ve birim hatayı temsil eder. Bu metrikler, modelin tahmin doğruluğunu ve genelleme yeteneğini değerlendirirken kullanılır ve modelin performansını optimize etmek için önemli ipuçları sağlar [25].

2.1.2.2 Denetimsiz Öğrenme

Denetimsiz öğrenme, makine öğrenmesi alanında denetimli öğrenmeden farklı olarak, etiketlenmemiş verilerle çalışmayı amaçlayan bir tekniktir. Bu yöntem, veri setindeki gizli yapıların veya kalıpların keşfedilmesine odaklanır. Denetimsiz öğrenme algoritmaları, verilerin doğal gruplarını veya ilişkilerini ortaya çıkarmak için kullanılır. Bu teknik, özellikle veri madenciliği, özellik çıkarımı ve boyut indirgeme gibi alanlarda yaygın olarak uygulanır [18].

Denetimsiz öğrenmenin iki ana bileşeni, giriş verileri (input) ve bu verilerin altında yatan yapıları belirlemeye yönelik algoritmalardır. Giriş verileri, modelin analiz edeceği özellikler kümesini içerir. Ancak bu veriler etiketlenmemiştir, yani her veri noktası için belirlenmiş bir çıkış etiketi yoktur. Denetimsiz öğrenme algoritmaları, veri kümesindeki benzerlikleri veya farklılıkları analiz ederek veriler arasındaki ilişkileri anlamaya çalışır [33].

Kümeleme, denetimsiz öğrenme algoritmalarının verileri benzer gruplara ayırmak için kullandığı bir yöntemdir. Kümeleme problemlerinde, modelin amacı, giriş verilerini doğal gruplara veya kümelere ayırmaktır. Yaygın kümeleme algoritmaları arasında k-means, hiyerarşik kümeleme ve DBSCAN (Density-Based Spatial Clustering of Applications with Noise) bulunur. K-means algoritması, verileri k sayıda kümeye ayırarak her kümenin merkezini (centroid) belirler ve verileri bu merkezlere en yakın olacak şekilde gruplandırılır. Hiyerarşik kümeleme, verileri ağaç yapısında hiyerarşik olarak gruplar. DBSCAN ise yoğunluk tabanlı bir algoritma olup, verileri yoğunluk bölgelerine göre kümeler ve gürültülü noktaları belirler [20].

Boyut indirgeme, denetimsiz öğrenme algoritmalarının veri setinin boyutunu azaltarak daha yönetilebilir ve anlamlı hale getirmek için kullandığı bir yöntemdir. Boyut indirgeme problemlerinde, modelin amacı, yüksek boyutlu verileri daha düşük boyutlu bir temsile dönüştürmektir. Yaygın boyut indirgeme algoritmaları arasında Ana Bileşen Analizi (PCA), Türetilmiş Bileşen Analizi (ICA) ve t-dağılımlı Stokastik Komşu Yerleştirme (t-SNE) bulunur. Ana Bileşen Analizi (PCA) veri setindeki en yüksek varyansa sahip bileşenleri belirleyerek boyutları indirger. ICA, bağımsız bileşenleri çıkararak verilerin bağımsız kaynaklarını ortaya çıkarır. t-SNE, yüksek boyutlu verileri iki veya üç boyutlu bir temsile dönüştürerek görselleştirmeye yardımcı olur [23].

Denetimsiz öğrenme modellerinin başarısı, çeşitli değerlendirme metrikleri kullanılarak ölçülür. Bu metrikler, modelin veri kümesindeki yapıları veya kalıpları ne kadar iyi keşfettiğini değerlendirmek için kullanılır. Kümeleme problemlerinde yaygın olarak kullanılan metrikler arasında Silhouette Skoru, Davies-Bouldin İndeksi ve Calinski-Harabasz İndeksi bulunur. Silhouette Skoru, veri noktalarının kendi kümelerine ne kadar iyi oturduğunu ve diğer kümelerden ne kadar iyi ayrıldığını ölçer. Davies-Bouldin İndeksi, kümelerin birbirine olan benzerliğini değerlendirir; daha düşük

değerler, daha iyi kümeleme performansını gösterir. Calinski-Harabasz İndeksi, küme içi dağılımın ve küme merkezlerinin birbirine olan uzaklığının oranını hesaplar; daha yüksek değerler, daha iyi kümeleme performansını gösterir. Boyut indirgeme problemlerinde ise, veri kaybının minimumda tutulması ve yeni temsillerin orijinal verilerin önemli özelliklerini koruması önemlidir [40].

Denetimsiz öğrenme, büyük ve karmaşık veri kümelerindeki gizli kalıpları ve yapıları ortaya çıkarmada kritik bir rol oynar. Etiketlenmemiş verilerle çalışarak, denetimsiz öğrenme algoritmaları, veri madenciliği ve keşifsel veri analizi gibi alanlarda değerli içgörüler sağlar. Bu teknikler, geniş bir uygulama yelpazesine sahip olup, veri biliminde önemli bir araç olarak kabul edilir.

2.1.2.3 Makine Öğrenimi Uygulamaları

Makine öğrenmesi, klasik algoritmaların performansının yetersiz olduğu durumlarda kullanılır. Ağ analizleri, spam mail incelemeleri, optik karakter tanıma gibi yüksek hesaplama gücü gerektiren alanlarda uygulanır. Lojistik, tedarik, üretim ve taşımacılık alanlarında; üretim hızını artırmak, robot kollarını eğitmek, endüstriyel araçları otomatikleştirmek ve operasyonları daha verimli hale getirmek için kullanılabilir. Örneğin, Amazon ve Netflix gibi platformlar, kullanıcı taleplerine göre kaynak dağıtımını optimize etmek amacıyla makine öğrenmesi algoritmalarını kullanır. Ayrıca, cihaz bakımı veya arızasını öngörmek için de bu teknolojiye başvurulur.

Günümüzde, makine öğrenmesinin kullanım alanları hızla genişlemektedir. Sağlık sektöründe, hastalık teşhisi ve tedavi planlaması gibi konularda makine öğrenmesi önemli bir rol oynamaktadır. Örneğin, görüntüleme verilerini analiz ederek kanser teşhisi koymak veya hastaların tıbbi geçmişine dayalı olarak kişiselleştirilmiş tedavi önerileri sunmak mümkündür. Yapay zeka destekli robotlar, cerrahi operasyonlarda cerrahlara yardımcı olarak daha hassas ve güvenli müdahaleler yapılmasını sağlar.

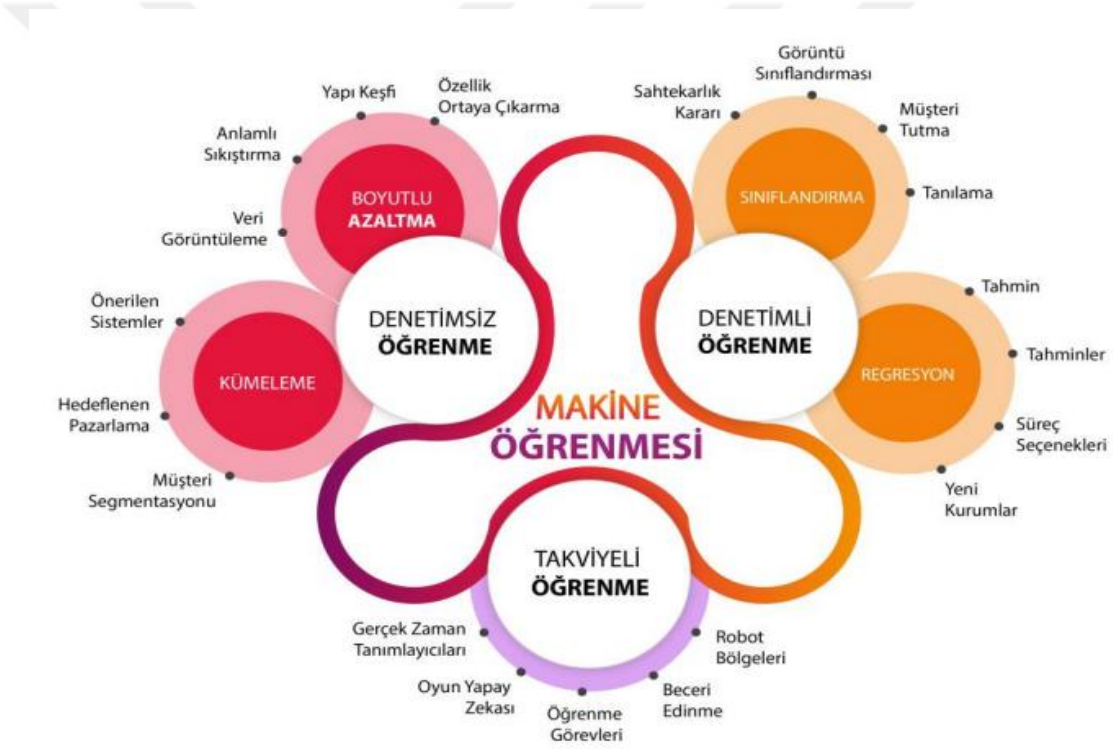
Finans sektöründe, makine öğrenmesi ile dolandırıcılık tespiti, kredi risk analizi ve algoritmik ticaret gibi uygulamalar yaygınlaşmıştır. Bankalar, müşterilerinin harcama alışkanlıklarını analiz ederek kişiselleştirilmiş finansal hizmetler sunmakta ve risk yönetimini daha etkin bir şekilde gerçekleştirmektedir.

Eğitim alanında, makine öğrenmesi öğrenci performansını izlemek, öğrenme materyallerini kişiselleştirmek ve adaptif öğrenme sistemleri geliştirmek için kullanılmaktadır. Bu sayede, öğrencilerin bireysel ihtiyaçlarına uygun eğitim programları oluşturulabilir ve öğrenme süreçleri optimize edilebilir.

Tarım sektöründe, makine öğrenmesi ile mahsul verimliliği artırılmakta, zararlıların erken tespiti sağlanmakta ve sulama sistemleri optimize edilmektedir. İklim verileri ve toprak analizleri kullanılarak, çiftçilere en uygun ekim ve hasat zamanları konusunda öneriler sunulmaktadır.

Otonom araçlar, makine öğrenmesi teknolojileri sayesinde trafik işaretlerini tanıyabilmekte, yol ve hava koşullarını değerlendirebilmekte ve güvenli sürüş sağlayabilmektedir. Tesla ve Waymo gibi şirketler, sürücüsüz araç teknolojilerinde makine öğrenmesini etkin bir şekilde kullanarak geleceğin ulaşım çözümlerini geliştirmektedir.

Sonuç olarak, makine öğrenmesi, pek çok sektörde devrim niteliğinde yenilikler sunmakta ve hayatımızın birçok alanında önemli iyileştirmeler sağlamaktadır. Gelişen teknoloji ile birlikte, makine öğrenmesinin uygulama alanlarının daha da genişlemesi ve yeni fırsatlar sunması beklenmektedir.



Şekil 3: Makine Öğrenmesi yöntemlerinin gösterimi

2.1.3. Derin Öğrenme

Derin öğrenme, makine öğrenmesinin bir alt dalıdır ve yapay sinir ağı algoritmalarını kullanarak karmaşık yapılar ve karmaşık verilerle çalışabilme imkanı sunar. Derin öğrenme yöntemleri sayesinde, görüntü işleme ve görsel nesne tanıma gibi alanlardaki başarı oranı artmıştır. Derin

öğrenme, karmaşık ve çok boyutlu veri setlerini daha iyi analiz eder. Derin öğrenme, çok katmanlı yapay sinir ağlarına dayanır.

Derin öğrenme karmaşık veri yapıları ve büyük veri setlerini işlemek için daha idealdir. Görüntü işleme, ses tanıma, doğal dil işleme ve biyomedikal veri analizi gibi çeşitli alanlarda başarılı sonuçlar elde edilmiştir. Örneğin, derin öğrenme yöntemleri, tıbbi görüntülerin analizinde doktorlara yardımcı olarak hastalıkların erken teşhisinde kullanılmaktadır. Otomotiv sektöründe, otonom araçların çevrelerini algılaması ve güvenli sürüş yapması için derin öğrenme algoritmaları kullanılmaktadır [27].

Finansal hizmetlerde dolandırıcılık tespiti ve müşterilerin davranışlarının analizi gibi uygulamalarda da derin öğrenme yaygın olarak kullanılmaktadır. Perakende sektöründe ise müşteri tercihlerini analiz ederek kişiselleştirilmiş öneriler sunmak ve envanter yönetimini optimize etmek için kullanılmaktadır.

Sonuç olarak, derin öğrenme, yapay zeka teknolojilerinin en güçlü araçlarından biri olarak pek çok alanda devrim niteliğinde yenilikler sunmakta ve çeşitli endüstrilerde önemli iyileştirmeler sağlamaktadır. Gelişen teknoloji ve artan veri miktarı ile birlikte, derin öğrenme uygulamalarının daha da yaygınlaşması ve yeni fırsatlar yaratması beklenmektedir.

2.2. BEYİN VE TÜMÖR İLE İLGİLİ GENEL BİLGİLER

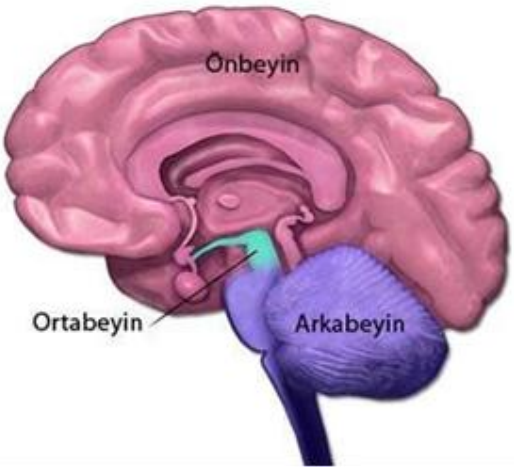
Beyin, merkezi sinir sisteminin karmaşık bir parçası olup, vücudun kontrolünü sağlar ve bilişsel işlevlerin merkezidir. Beyin kanseri, beyin dokusunun içinde başlayan ve çevresindeki sağlıklı hücreleri etkileyen bir tür tümördür. Bu tümörler genellikle primer veya metastatik olarak sınıflandırılır. Primer beyin tümörleri, beyin içindeki dokularda başlar ve genellikle iyi veya kötü huylu olarak değerlendirilir. Metastatik beyin tümörleri ise vücudun başka bir yerinde başlayan kanser hücrelerinin beyne yayılmasıyla ortaya çıkar. Beyin tümörleri genellikle baş ağrısı, bulantı, kusma, nörolojik bozukluklar ve duyusal değişiklikler gibi semptomlarla ortaya çıkar ve teşhis genellikle MRI veya CT taramaları ile konur. Tedavi seçenekleri arasında cerrahi müdahale, radyoterapi, kemoterapi ve hedefe yönelik ilaç tedavileri bulunur, ancak tedavi planı tümörün tipine, yerine ve hastanın genel sağlık durumuna bağlı olarak belirlenir [22].

2.2.1 BEYİN

Beyin, insan vücudundaki en karmaşık ve en büyük yapılardan biridir. Trilyonlarca bağlantıyla iletişim kuran 100 milyardan fazla sinirden ve birbiriyle koordineli çalışan birçok özel yapıdan oluşur.

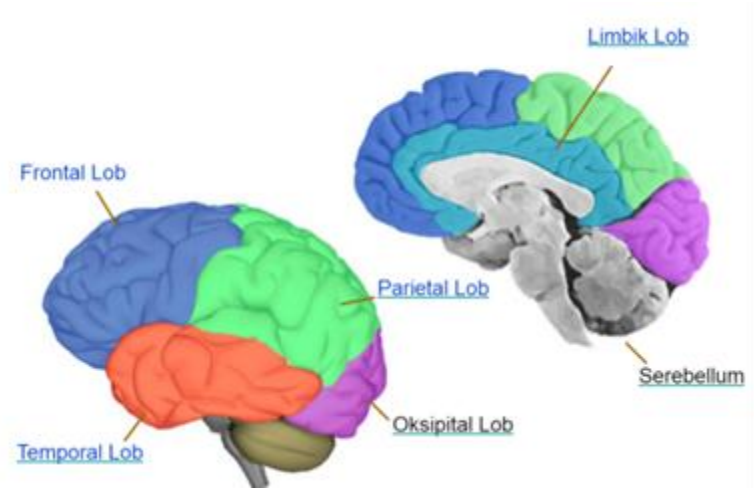
2.2.1.1. Beynin Yapısı ve Bölümleri

Beyin, ön beyin, orta beyin ve arka beyin olmak üzere üç ana kısımdan meydana gelir. Arka beyin, kalp atım hızı, nefes alıp verme gibi hayati önem taşıyan fonksiyonları kontrol eder. Arka beyin omuriliğin üst kısmı, beyin sapı ve serebellum olarak üç ana kısımda incelenebilir. Orta beyin ise beyin sapının üst tarafında yer alır ve bazı refleksif hareketlerin veya istemli hareketlerin kontrolünü sağlar. İnsan beyninin en büyük ve gelişmiş bölümü ise ön beyindir: serebrum ve onun altındaki yapılardan meydana gelir. Beyin görsellerine bakıldığında genellikle görünen serebrumdur. Serebrum derin bir yarık ile iki ayrı yarım küreye ayrılır, bu iki yarım küre, birbirleri ile alt tabakada yer alan sinir lifleri bağlantısıyla iletişim kurarlar.



Şekil 4: Beynin temel bölümleri

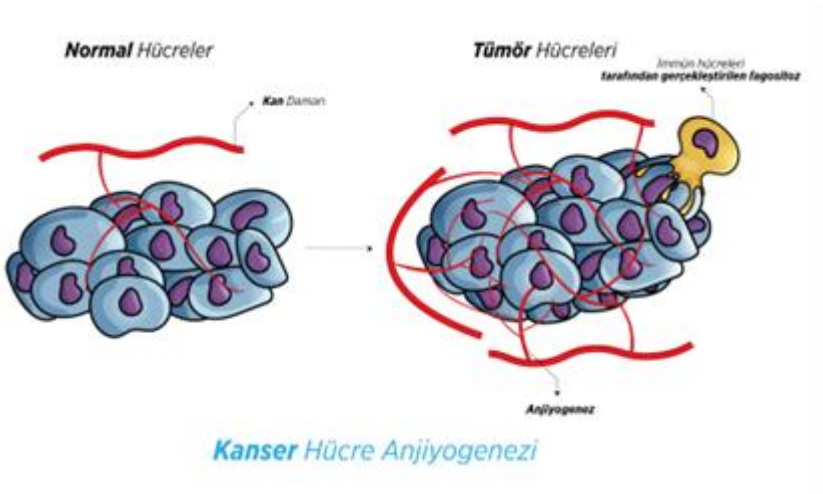
Beyin, vücudun kontrol merkezidir ve kalp, akciğerler gibi tüm organlar ile koordineli bir şekilde çalışmaktadır. Görme, koku alma, dokunma, duyma, tatma duyularımızın hepsi beyine bağlı çalışmaktadır. Örneğin görme duyusu, gözden gelen sinyaller beyine iletildiğinde, beyinde anlamlı bir görüntü oluştuğunda mümkün olmaktadır. Beynin gerçekleştirmesi gereken görevler çeşitli loblara bölünmüştür. Her lob kendi içerisinde belirli bir görev tanımı ile çalışır ve görevleri farklıdır. Frontal lob, vücudun hareketi, konsantrasyon, planlama, kişilik, kelimelerin anlamı, problem çözme yetisi, konuşma, duygusal tepkiler, koku alma gibi işlevlerden sorumludur. Oksipital lob, görme, beyincik, ince kas kontrolü, denge ve koordinasyondan sorumludur. Temporal lob, işitme, yüzleri tanıma, duygu, uzun süreli hafızaya atmadan sorumludur. Limbik lob**,** mutluluk, üzüntü ve aşk gibi duyguları kontrol etmeyi sağlar. Parietal lob ise dokunma ve baskı, damak tadı, vücut bilincinden sorumludur [4].



Şekil 5: Beynin lobları

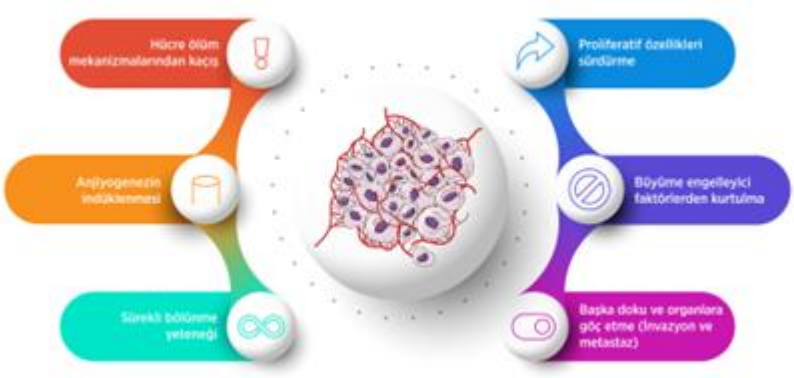
2.2.2 KANSER

Normal durumlarda, vücut hücreleri mitoz olarak bilinen bir sürece göre bölünürler. Ancak hücrelerde aniden bir hata meydana gelirse, kontrolsüz hücre bölünmesi meydana gelir ve trilyonlarca yeni hücre oluşumu başlar. Bu durum kanser veya tümör olarak isimlendirilir. Bu hastalık genellikle kalp gibi bazı organlar hariç vücudun herhangi bir yerinde oluşabilir ve kanser hücrelerinin diğer bölgelere sıçrama ve yayılma özelliği vardır. Bu duruma metastaz denir. Hücreler sinyal almadan anormal şekilde büyür, çoğalırlarsa ve programlanmış hücre ölümü olan apoptozis durdurulursa, tümör olarak adlandırılan bir doku kümesi oluştururlar. Ayrıca, bu hastalık dünya genelinde en yaygın olan hastalıktır ve her 6 ölümden birinin nedeni kanserdir, özellikle kanserli hücrelerin metastazı büyük ölçüde ölüm sayısının artmasına neden olur.



Şekil 6: Normal hücreler ve tümör hücreleri

Kanserin, 2020'de dünya genelindeki ölüm oranı yaklaşık 10 milyon insan kadardır ve her yıl yaklaşık 400.000 çocuk kanser hastalığına yakalanmaktadır. Kanserin ana kaynağı genetikle ilişkilidir, özellikle hücre gelişimi ve bölünmesinden sorumlu olan gen ile ilgilidir. Bu genin anormal şekilde çalışmaya başlaması kansere neden olur. Çevredeki kimyasallar, kirleticiler ve diğer faktörler nedeniyle insanlar etrafındaki pek çok değişiklik ve kanserojen madde, bu genin DNA'sında değişikliklere neden olarak kanser hastalığına yol açar.



Şekil 7: Kanser hücrelerinin özellikleri

2.2.2.1. Kanser Türleri

Vücudun bulunduğu bölgeye ve etkilenen organa bağlı olarak, birçok farklı kanser türü mevcuttur. Bu türler arasında mesane kanseri, meme kanseri, akciğer kanseri, serviks kanseri, boyun kanseri, kolorektal kanser, baş ve boyun kanseri, karaciğer kanseri, jinekolojik kanser, böbrek kanseri, lenfoma, mezotelyoma, miyeloma, over kanseri, prostat kanseri, cilt kanseri, beyin kanseri, tiroid kanseri, uterus kanseri, vajinal ve vulvar kanserler yer almaktadır.

Mesane kanseri, belirli kimyasalların, genetik mutasyonların ve sıklıkla sigara tüketiminin etkisiyle oluşur. Mesane, üreter ve böbreklerdeki ürotelyal hücrelerde meydana gelebilir, ancak genellikle mesane boşluğunda görülür. Erken belirtiler gösterse de tekrarlama riski vardır ve düzenli takip gerektirir. Meme kanseri, meme hücrelerinin kontrolsüz şekilde büyümesiyle oluşan bir tümördür. Bu tümör, lenf düğümleri yoluyla vücudun diğer bölgelerine yayılabilir. Kadınlarda yaygın olup, mamografi ile erken teşhis edilebilir. Serviks kanseri, rahim ile vajina arasındaki bölgede gelişir ve genellikle insan papilloma virüsü (HPV) ve korunmasız cinsel ilişki ile ilişkilidir. HPV aşısı, bu

kanser türüne karşı korunmada etkilidir. Kolorektal kanser, kolon veya rektumda ortaya çıkar ve 45 yaş üzerindeki bireyler için düzenli testler önerilir.

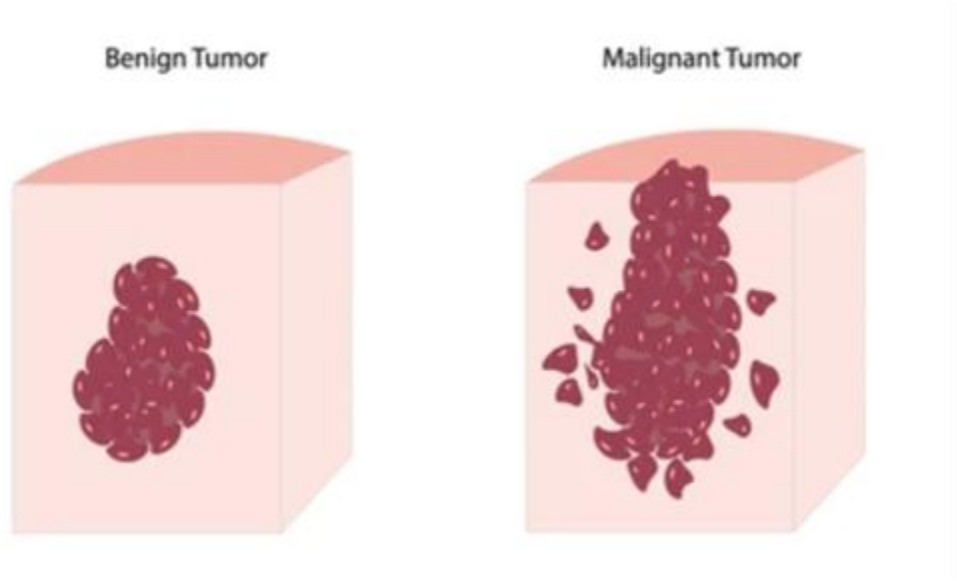
Kadın üreme organlarıyla ilgili olan jinekolojik kanserler, HPV, cinsiyet, aile öyküsü, yaş ve obezite gibi faktörlerle ilişkilidir. Baş ve boyun kanseri, genellikle sigara ve HPV'nin etkisiyle boğaz, ağız ve farinks bölgesinde ortaya çıkar ve erken evrede tespit edilirse tedavi edilebilir. Karaciğer kanseri, karaciğer hücrelerinde, safra kanallarında veya karaciğerin kan damarlarında oluşan tümörlerdir. Bu kanser türüne karşı hepatit B ve C aşıları önerilmektedir. Akciğer kanseri, hava kirliliği, sigara, pasif içicilik ve nargile tüketimi gibi faktörlerle ilişkilidir. İki ana tipi bulunur: küçük hücreli ve küçük hücreli olmayan kanser.

Böbrek kanseri, böbrek hücrelerinde gelişir ve başlıca iki türü vardır: açık hücreli böbrek hücreli karsinom ve açık hücreli olmayan böbrek hücreli karsinom. Lenfoma, lenfosit adı verilen beyaz kan hücrelerinde gelişen bir kan kanseridir. Miyeloma, kemik iliğinde bulunan plazma hücrelerinde ortaya çıkan bir kanser türüdür. Mezotelyoma, akciğerler ve karın boşluğu gibi organların dış sınırlarında gelişen ve asbest liflerinin solunması sonucu oluşan bir kanserdir. Vajinal, over ve vulvar kanserler, pelvik bölgede gelişen kanser türleridir; vajina, yumurtalıklar ve rahim tabakalarında görülür ve kadınlarda yaygındır.

Prostat kanseri, erkeklerde ve bazı trans kadınlarda prostat bezindeki hücrelerin anormal büyümesi sonucu gelişir. Cilt kanseri, uzun süreli ultraviyole ışınlarına maruz kalma sonucu cilt hücrelerinin anormal büyümesi ile ortaya çıkar. Tiroid kanseri, tiroid bezinde gelişir ve beyin kanseri, merkezi sinir sistemi yani beyin veya omurilikte ortaya çıkan tümörlerdir. Bu tür tümörlerin tehlikeli olup olmadığı, kanser hücrelerinin moleküler ve genetik yapısına bağlıdır; malign veya benign olarak sınıflandırılırlar.

Malign tümör tüm vücuda yayılma ve metastaz yapabilme özelliğine sahiptir ve hastalar için en zararlı olanıdır. Karsinom, sarkom ve blastoma gibi birçok türü vardır. Hızlı bir tedavi gerektirir ve yaşam için büyük bir tehdit oluşturur.

Benign tümör ise diğer bölgelere yayılmaz ve acil bir tedavi gerektirmez, ayrıca malign olan kadar zararlı değildir. Ancak çevreleyen alanlara baskı yaparak ağrıya neden olabilir [1].

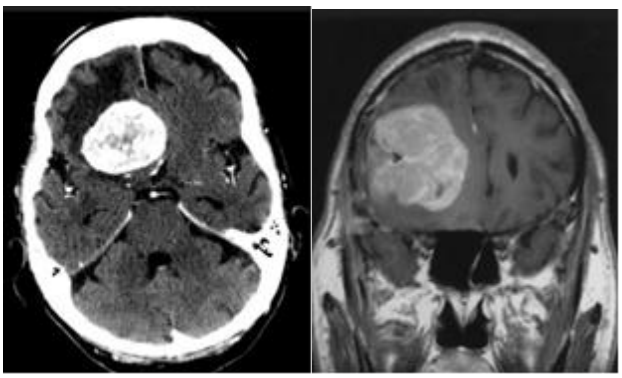


Şekil 8: Benign ve Malign tümör

2.2.3 Beyin Tümörü

Beyin tümörleri, merkezi sinir sistemi (beyin ve omurilik) hücrelerinde meydana gelen anormal hücre büyümelerdir. Bu tümörler, en başta glial hücreler olmak üzere çeşitli hücre tiplerinden oluşabilmektedir.

Glial hücreler, sinir hücrelerine destek veren ve sinir sisteminin işlevlerini yerine getirmesine yardımcı olan hücrelerdir. Beyin tümörleri, histopatolojik özellikleri, büyüme hızları, boyutları ve konumlarına göre sınıflandırılır. Dünya Sağlık Örgütü (WHO) tarafından kullanılan bir sınıflandırma sistemine göre, beyin tümörleri dört temel grupta ele alınır: Gliomlar (glial hücre tümörü), menenjiyomlar (beyin zarı tümörü), nöroepitelyal tümörler (beyin dokusu tümörü) ve hemanjioblastomlar (damarsal orjinli tümör). Her bir grup, belirli tümör tiplerini içerir ve farklı tedavi türleri gerektirmektedir [44].



Şekil 9: Beyin tümörlü MRG görüntüsü

2.2.3.1 Gliomlar (Glial Hücre Tümörü)

Gliomlar, glial hücrelerden kaynaklanır, bu hücreler sinir hücrelerinin desteğini sağlar. Glial hücreler arasında astrositler, oligodendrositler ve ependim hücreleri bulunur. Gliomlar, bu hücrelerden birinden veya birkaçından kaynaklanabilir. Gliomlar, hücrelerin hangi kısımlarının etkilendiğine bağlı olarak çeşitli şekillerde sınıflandırılır. Gliomların bazı örnekleri şunlardır:

Astrocytoma: Astrocyte adı verilen glial hücrelerden kaynaklanır. Astrocytomas, iyi huylu (düşük dereceli) veya kötü huylu (yüksek dereceli) olabilir.

Oligodendroglioma: Oligodendrocyte adı verilen glial hücrelerden kaynaklanır. Bu tümörler genellikle yavaş büyür.

Ependimoma: Ependim hücrelerinden kaynaklanır. Bu tür tümörler genellikle beyin ve omurilik sıvısı içeren bölgelerde bulunur.

Gemistositik Astrocytoma: Bu tür tümör, astrocytomasın bir alt türüdür ve gemistosit adı verilen belirli bir hücre tipinden kaynaklanır.

Glioblastoma Multiforme (GBM): Bu en kötü huylu gliom türüdür. Hızla büyür ve beyin dokusunu ciddi şekilde etkileyebilir.

Tedavi seçenekleri, tümörün türüne, büyüklüğüne, konumuna ve hastanın genel sağlık durumuna bağlı olarak değişir. Tedavi seçenekleri arasında cerrahi müdahale, radyoterapi, kemoterapi ve hedeflenmiş ilaç tedavileri yer alabilir [44].

2.2.3.2. Menengiyomlar (Beyin Zarı Tümörü)

Meningiyomlar, beyin ve omuriliği saran zarlar olan meninkslerden (beyin zarları) kaynaklanan tümörlerdir. Bu tümörler genellikle iyi huylu (düşük dereceli) olma eğilimindedir, ancak nadiren kötü huylu (yüksek dereceli) olabilirler.

Meningiyomlar, üç ana meninks tabakasından herhangi birinden kaynaklanabilir:

Dura Mater: Beynin en dış tabakası olan dura materden kaynaklanan meningiyomlar en sık görülenlerdir.
Arachnoid Mater: Orta tabaka olan arachnoid materden kaynaklanan meningiyomlar daha nadirdir.
Pia Mater: En içteki tabaka olan pia materden kaynaklanan meningiyomlar oldukça nadirdir.

Meningiyomlar genellikle belirtiler yapmazlar ve tesadüfen beyin görüntüleme testleri sırasında tespit edilirler. Ancak büyümeleri durumunda baş ağrısı, baş dönmesi, görme problemleri, nörolojik bozukluklar gibi semptomlara neden olabilirler.

Tedavi genellikle cerrahi müdahaleyi içerir. Cerrahi müdahale bazen tümörün tamamen çıkarılmasını sağlar. Ancak tümörün konumu veya boyutu cerrahi müdahaleyi zorlaştırabilir. Bu durumlarda radyoterapi veya diğer tedavi seçenekleri kullanılabilir [44].

2.2.3.3. Nöroepitelyal Tümörler (Beyin Dokusu Tümörü)

Nöroepitelyal tümörler, sinir sisteminin çeşitli bölgelerinde bulunan ve nöroepitelyal hücrelerden kaynaklanan tümörlerdir. Nöroepitelyal hücreler, sinir sisteminin gelişimi sırasında önemli bir rol oynayan ve çeşitli yapıları oluşturan hücrelerdir.

Nöroepitelyal tümörler genellikle çocukluk veya genç yetişkinlik döneminde ortaya çıkarlar. Bu tümörler çeşitli tiplerde olabilir ve bazıları iyi huylu (düşük dereceli) iken bazıları kötü huylu (yüksek dereceli) olabilir. Nöroepitelyal tümörler arasında şunlar bulunabilir:

Medulloblastoma: Genellikle çocukluk döneminde görülen ve sıklıkla beyincikten kaynaklanan bir tür nöroepitelyal tümördür. Kötü huylu bir tümördür ve hızla büyüme eğilimindedir.

Pinealoma (Pineal Tümör): Pineal bezden kaynaklanan tümörlerdir. Bu tür tümörler arasında pinealoblastoma ve pinealositoma gibi çeşitli tipler bulunabilir.

Embriyonel Tümörler: Embriyonel hücrelerden kaynaklanan çeşitli tümörler arasında medulloblastoma, atipik teratom ve teratoid rhabdoid tümör bulunur.

Optik Sinir Gliomları: Optik sinirde (göz siniri) oluşan tümörlerdir. Bunlar genellikle çocukluk döneminde görülür ve nörofibromatozis tip 1 (NF1) gibi genetik sendromlarla ilişkilidir.

Ependimoma: Ependim hücrelerinden kaynaklanan tümörlerdir. Beyin ve omurilikte oluşabilirler.

Nöroepitelyal tümörlerin tedavisi, tümörün tipine, boyutuna, konumuna ve hastanın genel sağlık durumuna bağlı olarak cerrahi, radyoterapi, kemoterapi veya hedeflenmiş tedavileri içerebilir. Tedavi seçenekleri genellikle bir multidisipliner ekibin katılımını gerektirir ve hastanın durumuna özgü olarak belirlenir [44].

2.2.3.4. Hemangioblastom (Damarsal Orjinli Tümör)

Hemangioblastomlar, nadir görülen vasküler tümörlerdir ve genellikle beyin ve omurilikte ortaya çıkarlar. Bu tümörler, vasküler (damar) endotel hücrelerinden ve stromal (destekleyici) hücrelerden oluşur. Hemangioblastomlar genellikle yavaş büyürler ancak nadiren daha agresif bir seyir gösterebilirler.

Hemangioblastomlar, genellikle belirgin bir damar yapısına sahiptir ve çevrelerinde yoğun bir damar ağı oluşturabilirler. Beyin ve omurilikteki tümörler basıya bağlı semptomlara neden olabilirler ve bu semptomlar baş ağrısı, bulantı, kusma, dengesizlik, koordinasyon kaybı ve diğer nörolojik belirtiler olabilir.

Bu tümörler, VHL (Von Hippel-Lindau) sendromu ile ilişkilidir olabilir. VHL sendromu, çeşitli organlarda tümörlerin geliştiği kalıtsal bir durumdur ve hemangioblastomlar bunların arasında yer alır.

Tedavi genellikle cerrahi olarak tümörün çıkarılmasını içerir. Ancak bazı durumlarda tümörün yerleşimi veya boyutu cerrahi müdahaleyi zorlaştırabilir. Bu durumlarda radyoterapi, embolizasyon veya diğer tedavi yöntemleri kullanılabilir. Tedavinin başarısı tümörün boyutuna, yerine, yayılımına ve hastanın genel sağlık durumuna bağlıdır.

Damar kökenli tümörler, vasküler (damar) dokudan kaynaklanan tümörlerdir. Bu tümörler genellikle kan damarları veya lenfatik damarlar gibi vasküler yapılarla ilişkilidir. Damar kökenli tümörler geniş bir yelpazede olabilir ve farklı vasküler yapılar veya hücre tiplerinden kaynaklanabilirler.

Bazı damarsal orijinli tümörler şunları içerebilir:

Hemangioma: Hemangioma, damarların anormal bir şekilde büyümesi sonucu oluşan benign (iyi huylu) bir tümördür. Hemangiomas genellikle deri altında veya iç organlarda bulunabilirler.

Hemangioendotelyoma: Bu tür tümörler, endotel hücrelerinden kaynaklanır ve genellikle kan damarlarının iç yüzeyinde oluşurlar.

Hemangiosarkoma: Hemangiosarkomalar, kötü huylu (malign) damar kökenli tümörlerdir. Bu tümörler hızlı bir şekilde büyüyebilir ve metastaz yapabilirler.

Lenfangioma: Lenfangiomalar, lenfatik damarlardan kaynaklanan benign tümörlerdir. Genellikle lenfatik sistemde (lenf düğümleri veya lenfatik kanallar) bulunurlar.

Damar malformasyonları: Damar malformasyonları, doğuştan gelen anormal vasküler yapılar olarak tanımlanır. Bunlar arteriovenöz malformasyonlar (AVM'ler), venöz malformasyonlar, kavernöz hemanjiyomlar gibi çeşitli tiplerde olabilir.

Damar kökenli tümörlerin belirtileri ve semptomları tümörün türüne ve yerleşimine bağlı olarak değişebilir. Tedavi seçenekleri, tümörün büyüklüğü, yerleşimi ve hastanın genel sağlık durumu göz önüne alınarak belirlenir. Cerrahi çıkarma, radyoterapi, embolizasyon ve ilaç tedavisi gibi yöntemler kullanılabilir.

2.2.3.5. Beyin Tümörlerinin Derecelendirilmesi

Beyinde oluşan tümörler dereceleme sistemi kullanılarak 4 farklı derecede değerlendirilir. Bu sistem, tedavi için tümöre nasıl bir işlem veya bakım yapılacağını belirlemede yardımcı olur. Dünya genelinde şu an için Dünya Sağlık Örgütü'nün (WHO) belirlemiş olduğu sınıflandırma sistemi kullanılmaya devam etmektedir.

Tümör büyümesini ve gelişmesini saptamak için, tümörün işlevine ve özelliklerine bakılarak hareket edilmektedir. Beyin tümörünü belirlemek için kullanılan bazı belirleyiciler şunlardır:
Boyut ve konum, etkilenen doku veya hücrelerin türü, rezektabilite (tümörün bir kısmının veya tümünün ameliyat ile çıkarılma ihtimali), kanserin beyin veya omurilik içerisinde yaygınlaşması, kanserin beyin veya merkezi sinir sisteminin ötesine yayılma ihtimalidir.

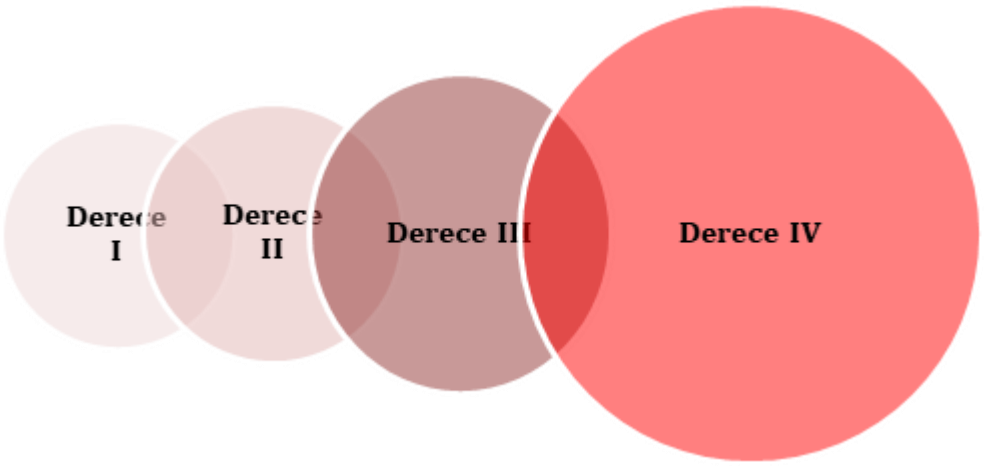
Beyin tümörleri derecelendirme olarak vücutta oluşabilen diğer kanser tiplerine göre farklılık göstermektedir. Diğer kanser tipleri tümörün konumuna, boyutuna ve yayılma olasılığına göre evrelendirilirken beyin tümöründe ise tümör değerlendirilirken tümörlü hücrelerin mikroskop altında ne kadar agresif göründüğüne bakılır. Dünya Sağlık Örgütü (WHO) tarafından belirlenen ve hala tümörün derecelendirilmesinde kullanılan 4 evre şu şekildedir:

Derece I: Tümör, en az kötü huyludur, nadiren yakın dokulara yayılır, yavaş büyür ve tümörü ameliyatla tamamen çıkarmak mümkün olabilir. Ancak birinci derece bir tümör bile tedavi edilmezse yaşamı tehdit edebilir.

Derece II: Tümör, 1. derece tümörlerden daha hızlı büyür ancak yine de yavaş büyür denilebilir.Anormal mikroskobik bir görünüme sahiptir. Bu tümörler çevreleyen normal dokuyu istila edebilir ve 3. derece tümör veya daha ileri derece olarak tekrarlayabilir. Bazı tip 2. derece tümörler (astrositomlar, oligodendroglioma, oligoastrositomalar) daha yüksek maligniteye ilerleme eğilimindedirler.

Derece III: Tümör kötü huyludur ve tümörün yakındaki dokulara yayılma olasılığı yüksektir, hızla büyür. Bu derecede olan tümör hücreleri normal olan diğer hücrelere göre farklı görünür. Tekrarlama şansı yüksektir ve çoğu durumda 3. derece tümörlü hastalar kemoterapi tedavisi alırlar.

Derece IV: Dünya Sağlık Örgütü'ne (WHO) göre en kötü huylu tümördür.Bu evredeki tümör hücreleri normal hücrelere benzemez, hızla büyür ve hızla yayılır. DSÖ bu tümörleri tipik olarak ameliyat öncesi ve sonrası hızlı hastalık gelişimi ve ölümcül bir sonuçla ilişkili aktif ve nekroza eğilimli olarak belirlemiştir.



Şekil 10: Tümör dereceleri şeması

2.2.3.6 Beyin Tümörü Risk Faktörleri

Beyin tümörü risk faktörleri genellikle karmaşık bir etkileşim içinde olan çeşitli faktörlerden kaynaklanır. Bu faktörler aşağıdaki gibi gruplandırılabilir:

Ailede beyin tümörü öyküsü var ise bireyin bu tümörlere yakalanma riskini artırabilir, bazı genetik sendromlar (örneğin, Nörofibromatoz tip 1 ve 2, Tuberöz skleroz kompleksi, Von Hippel-Lindau hastalığı gibi) beyin tümörü riskini artırabilir, özellikle çocuklukta baş ve boyun bölgesine yapılan radyoterapi, ileriki yaşlarda beyin tümörü riskini artırabilir. Genellikle beyin tümörleri ileri yaşlarda daha sık görülür [10]. İmmün yetmezlik durumları (örneğin, HIV/AIDS gibi) beyin tümörü riskini artırabilir. Yapılan bazı çalışmalar beyin tümörü riskinin sigara içenlerde ve aşırı alkol tüketenlerde arttığını göstermiştir.

2.2.3.7 Beyin Tümörü Belirtileri

Beynin farklı bölümleri farklı işlevleri kontrol eder, bu nedenle beyin tümörü semptomları tümörün konumuna bağlı olarak değişecektir. Örneğin başın arkasındaki beyincikte yer alan bir beyin tümörü hareket, yürüme, denge ve koordinasyonda sorunlara neden olabilir. Tümör görmeden sorumlu olan optik yolu etkilerse görme değişiklikleri meydana gelebilir. Tümörün boyutu ve ne kadar hızlı büyüdüğü de kişinin hangi semptomları yaşayacağını etkiler. En yaygın semptomlar şunlardır: Baş ağrısı, nöbetler veya konvülsiyonlar, düşünme, konuşma veya kelime bulma zorluğu, kişilik veya davranış değişiklikleri, vücudun bir kısmında veya bir tarafında zayıflık, uyuşukluk veya felç, denge kaybı, baş dönmesi veya dengesizlik, duyma yeteneğini yitirmek [3].

2.2.3.8 Beyin Tümörü Tedavisi

Beyin tümörlerinin tedavisi, tümörün türüne, konumuna, büyüklüğüne, hastanın genel sağlık durumuna ve diğer bireysel faktörlere bağlı olarak değişiklik gösterir. Genel tedavi yöntemleri şunlardır: Cerrahi müdahale, beyin tümörlerinin tedavisinde yaygın olarak kullanılan bir yöntemdir. Ameliyatın amacı, tümörün mümkün olduğunca çıkarılmasıdır. Bu işlem, hastanın yaşam kalitesini artırmak ve semptomları hafifletmek için yapılır. Cerrahi müdahale sonrası hastaların bazı durumlarda ek tedavilere ihtiyacı olabilir. Radyoterapi, yüksek enerji ışınlarının kullanılarak kanser hücrelerini yok etmeyi amaçlar. Cerrahi müdahale sonrasında geride kalan tümör hücrelerini öldürmek veya tümörün büyümesini durdurmak için kullanılır. Radyoterapi, dış ışın tedavisi (external beam radiation) veya brakiterapi (beyin içine yerleştirilen radyoaktif maddeler) şeklinde uygulanabilir. Kemoterapi, kanser hücrelerini öldürmek veya büyümelerini durdurmak için kullanılan ilaçlarla yapılan tedavidir. Kemoterapi ilaçları genellikle ağız yoluyla alınır veya damar yoluyla verilir. Kemoterapi, tümörün türüne ve yayılma derecesine bağlı olarak tek başına veya diğer tedavi yöntemleriyle birlikte kullanılabilir. Hedefe yönelik tedaviler, tümör hücrelerinin belirli moleküler özelliklerine hedef alarak çalışan ilaçları içerir. Hedefe yönelik tedaviler, daha az yan etkiye sahip olabilir ve belirli beyin tümör türlerinde etkili olabilir. İmmünoterapi, bağışıklık sistemini güçlendirerek kanser hücreleriyle savaşmayı amaçlar. Bu tedavi yöntemi, hastanın bağışıklık sistemini aktive ederek tümör hücrelerini hedef alır ve yok eder. Steroid ilaçlar, beyin tümörlerine bağlı ödemi (şişlik) azaltmak için kullanılır. Antikonvülzanlar ise nöbetleri kontrol altına almak için reçete edilir. Rehabilitasyon, tedavi sonrasında hastaların, fiziksel, konuşma ve mesleki terapi gibi rehabilitasyon hizmetlerine ihtiyacı olabilir. Bu tedaviler, hastaların normal yaşama dönmesine yardımcı olur ve yaşam kalitesini artırır. Her hasta için en uygun tedavi planı, multidisipliner bir ekip tarafından belirlenir. Bu ekip genellikle nörologlar, onkologlar, beyin cerrahları, radyoterapistler, kemoterapi uzmanları ve diğer sağlık profesyonellerinden oluşur. Tedavi planı, hastanın genel sağlık durumu, tümörün özellikleri ve hastanın tercihlerine göre kişiselleştirilir.

2.2.3.8 Tümör Tespitinde Kullanılan Yöntemler

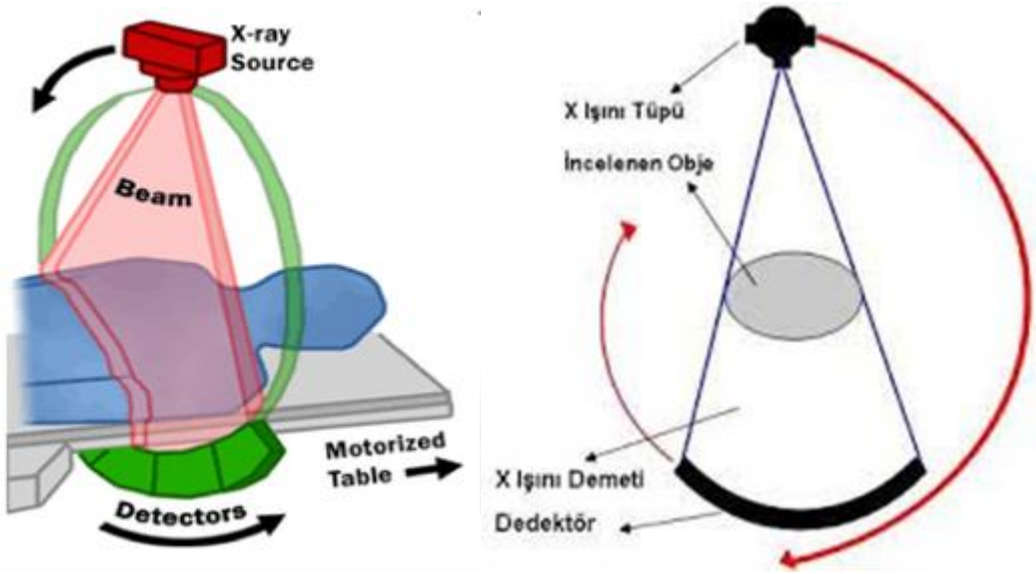
Beyin tümörlerinin klinik belirti ve semptomları jeneralize (eş zamanlı) veya fokal olabilir. Baş ağrısı ve nöbetler gibi jeneralize semptomlar kafa içi basıncının artmasına bağlıdır. Tek taraflı güçsüzlük veya kişilik değişiklikleri gibi odak semptomlar doku hasarı veya özelleşmiş bölgelerin basısına bağlıdır. Düşük dereceli tümörlerde veya hastalığın ilk aşamalarında semptomlar genellikle fokaldir, tümör boyutu büyüdükçe ve yayıldıkça jeneralize semptomlara ilerler.

Tümörle ilişkili baş ağrısının şiddetli, sabahları daha kötü olduğu, bulantı ve kusmayla ortaya çıktığı düşünülür. Bununla birlikte, beyin tümörü olan hastalar daha çok gerilim tipi baş ağrısı bildirirler. Uzun süreli bulantı, kusma, nöbetler, nörolojik semptomlar veya pozisyonel kötüleşme ile birlikte kronik, kalıcı baş ağrısı olan herhangi bir hasta beyin tümörü açısından değerlendirilmelidir. Bilişsel işlev bozukluğu (örneğin, dil, dikkat işlevi) beyin tümörü olan kişilerde yaygındır. Tümörden şüphelenildiğinde öykü ve fizik muayeneye ek olarak nörolojik muayene yapılmalıdır.

Tanı, uygun beyin görüntülemesi ile başlar ve ardından tanıyı doğrulamak için histopatolojik yöntemler kullanılır.

Günümüzde hastalıkların tanı, teşhis ve tedavilerinde biyomedikal görüntüleme yöntemleri kullanılmaktadır. Radyodiagnostik Cihazlar (Radyasyonlu Görüntüleme Cihazları) biyomedikal görüntülemede kullanılan önemli cihazlardır.

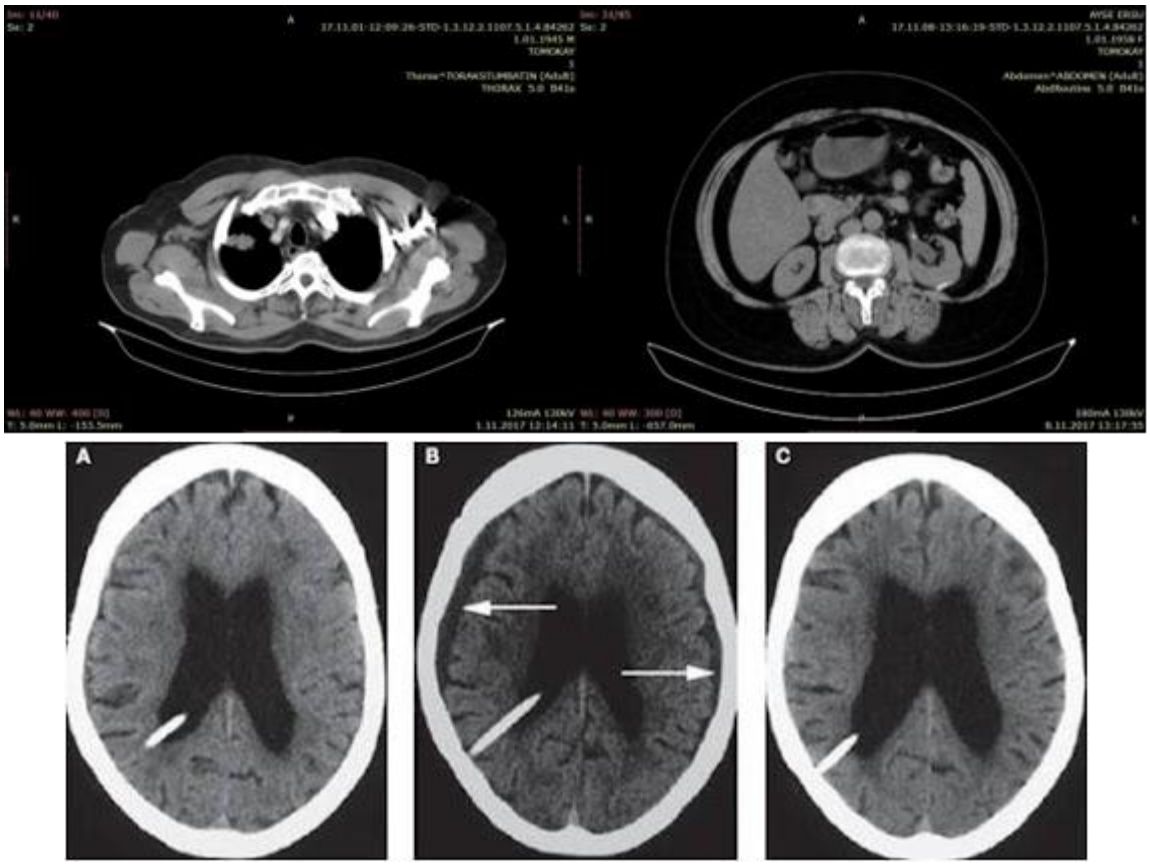
Bilgi edinme amacı için insan vücudunun tamamı ya da bir bölümü görüntülenip incelenebilmektedir. Elektriksel, radyolojik veya sonik yöntemler ile tıbbi görüntüler elde edilmektedir. Vücudun radyolojik görüntülenmesinde X-ışınları, elektromanyetizma, ses dalgaları ve radyoizotoplar kullanılmaktadır. Beyin görüntüleme tekniklerinde ise en yaygın kullanılan cihazlar BT ve MRG'dir.



Şekil 11: BT görüntüleme

2.2.3.8.1. Bilgisayarlı Tomografi

Bilgisayarlı Tomografi (BT), insan vücudunun enine kesit görüntülerini üretmek için x-ışını ekipmanını bir bilgisayar ve bir katot ışın tüpü ekranıyla birleştiren bir tanı teknolojisidir. Radyografik film, x-ışını profilini ölçen bir dedektör ile değiştirilir. Bilgisayar tomografi tarayıcının içinde, bir tarafında x-ışını tüpü, diğer tarafında dedektör monte edilmiş döner bir çerçeve vardır. Dönen bir çerçeve, x-ışını tüpünü ve dedektörü hastanın etrafında döndürüldüğünde bir x-ışını üretir. X-ışını tüpü ve dedektör her tam tur dönüş yaptığında, bir görüntü veya kesit elde edilir. X-ışını tüpü ve dedektör bu turu tamamlarken, dedektör, zayıflatılmış x-ışını demetinin sayısız profilini alır. Her profil, bilgisayar tarafından taranan dilimin iki boyutlu görüntüsü yeniden oluşturur. Üç boyutlu (3B) BT, spiral BT kullanılarak elde edilebilir. Spiral BT, hasta anatomisinin tümü tek bir konumdayken bir veri hacmi elde eder. Bu hacim veri seti daha sonra karmaşık yapıların 3B görüntülerini sağlamak için bilgisayarda yeniden konumlandırılabilir. Elde edilen 3B BT görüntüleri, tümör kitlelerinin 3B olarak görüntülenmesine yardımcı olur. Son zamanlarda, solunum hareketlerinin neden olduğu sorunların üstesinden gelmek için dört boyutlu (4B) BT tanıtılmıştır. Dört boyutlu BT, organ hareketliliği hakkında hem mekansal hem de zamansal bilgi verir.



Şekil 12: BT tarama örnekleri

Bilgisayar tomografi görüntüleme yönteminin avantajları; cerrahi işleme gerek kalmaması, hızlı ve ağrı yapmıyor olması, yüksek çözünürlükte sonuç vermesi, fiziksel yoğunluktaki ufak farkları kolaylıkla ayırt edebiliyor olmasıdır.

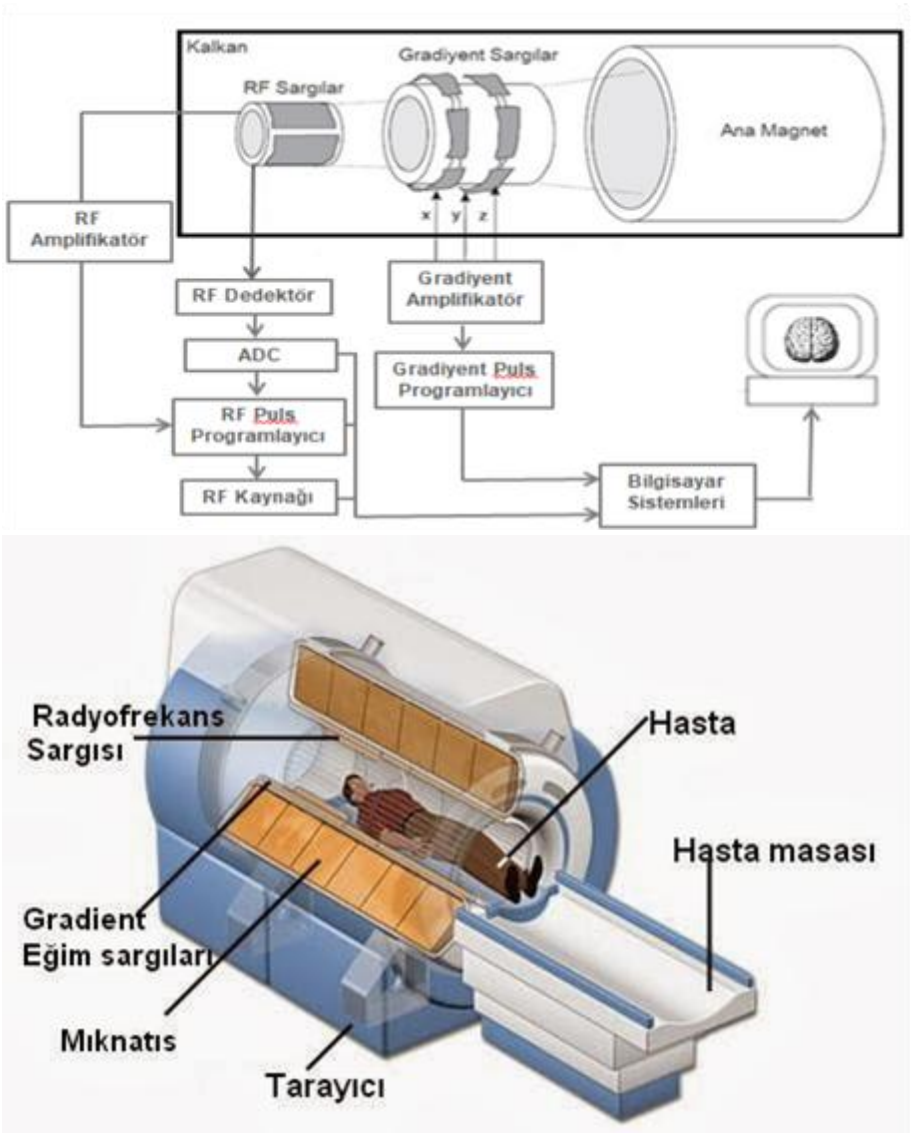
Bilgisayarlı tomografi görüntüleme riskleri; iyonlaştırıcı radyasyona maruz bıraktığı için yaşamın ilerleyen dönemlerinde kanser riskini arttırması, bir organın duvar yapısı içinde bulunan anormallikleri tespit edememesi, kontrast madde kullanılmadan yapılamaması ve kullanılan maddenin alerji ve toksisite oluşturabilmesi, yumuşak doku kontrastının düşük olduğu durumlarda kontrast çözünürlüğü düşük sonuç vermesidir.

Bilgisayar tomografi görüntülemenin yaygın uygulama alanları; insan vücudunun birçok bölümünün incelenmesi (beyin, sinüs, diz, ayak bileği, üriner sistem, yüz kemikleri, diş, eller, bilek, dirsek, omuz, kalça.), hastalık, travma ve anormallik teşhisi, girişimsel veya terapötik prosedürleri planlamak ve yönlendirmek, tedavinin etkinliğinin izlenmesi (kanser tedavisi) olarak söylenebilmektedir [12].

2.2.3.8.2. Manyetik Rezonans Görüntüleme (MRG)

MRG, vücut dokularını görüntülemek ve vücut iç işleyişini incelemek için manyetik ve radyo frekansı dalgalarını kullanan bir tanı yöntemidir. Morfolojik değişiklikleri görselleştirmek için kullanılan MRG, hastalıklı doku tarafından sunulan ortamın karakteristiği olan proton yoğunluğu ve manyetik dönüş gevşeme sürelerindeki değişiklikleri tespit etme yeteneğine dayanır. Manyetik görüntüleme cihazı ana mıknatıs, manyetik alan gradyan sistemi ve radyo frekansı (RF) sistemi olmak üzere üç ana bileşenden oluşur. Ana mıknatıs, manyetik alan oluşturan kalıcı bir mıknatıs olarak tanımlanır. Manyetik alan gradyan sistemi normalde sinyal lokalizasyonu için gerekli olan üç ortogonal gradyan bobinden meydana gelir. Radyo frekansı, bir döndürme sistemini harekete geçirmek için dönen, bir manyetik alan oluşturabilen bir verici bobinden devam eden manyetizasyonu elektrik sinyallerine dönüştüren alıcı bobinden oluşur [12].

Sinyaller MRG cihazı tarafından ölçümlenir ve bilgisayar bu sinyalleri anlamlı hale getirir ve görüntülere dönüştürür. Son zamanlarda, fonksiyonel manyetik rezonans görüntüleme (fMRG) adı verilen beynin aktif bir bölümünde meydana gelen küçük metabolik değişimleri ölçmek için MRG kullanılmaktadır. Beyin aktivitesinin non-invazif ölçümüne olanak sağlayan ileri bir nörogörüntüleme tekniği olan fonksiyonel manyetik rezonans görüntüleme (fMRI) beynin 3 boyutlu ve yüksek mekansal çözünürlüğe sahip aktivasyon haritalarının ve fonksiyonel bağlantısallıkların görüntülenmesine olanak tanır.

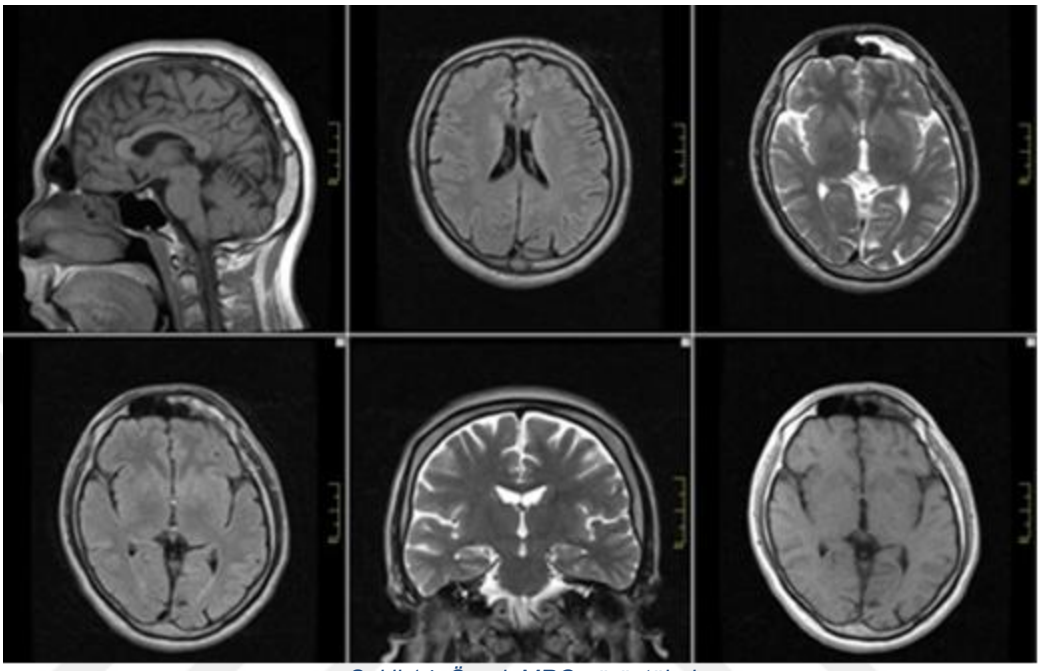


Şekil 13: Manyetik Rezonans sistem çalışma prensibi

Manyetik rezonans görüntüleme yönteminin avantajları; cerrahi işlem gerektirmemektedir ve ağrısızdır , iyonlaştırıcı radyasyon olmadan çekim yapılır, yüksek çözünürlüklü çekimler yapılabilir, kontrast madde kullanmadan çekim yapılabilir, yumuşak doku kontrastı yüksek seviyededir,

Manyetik rezonans görüntüleme riskleri; diğer cihazlara oranla düşük hassasiyet gösterir, tarama ve işlem uzun sürer, bir organın duvar yapısı içinde bulunan anormallikleri tespit edemez, kapalı MR cihazı klostrofobisi (kapalı alan korkusu) olan insanlar için uygun değildir, küçük çocuklarda net sonuç alabilmek adına sedasyon uygulanması gerekebilir, diğer cihazlar ile karşılaştırıldığında maliyeti yüksektir.

Manyetik rezonans görüntülemenin yaygın uygulama alanları; beyin ve omurilik anormalliklerinin incelemesi yapılır, vücudun çeşitli bölgelerindeki tümör, kist ve diğer anormallikler incelenir, eklemlerin yaralanmaları veya anormallikleri incelenir, karaciğer ve diğer iç organların hastalıkları incelenir, kadınlarda meydana gelen pelvik ağrının nedenleri incelenir, yapılan çekime göre ameliyat planı oluşturulabilir.



Şekil 14: Örnek MRG görüntüleri

2.3. Son Teknoloji

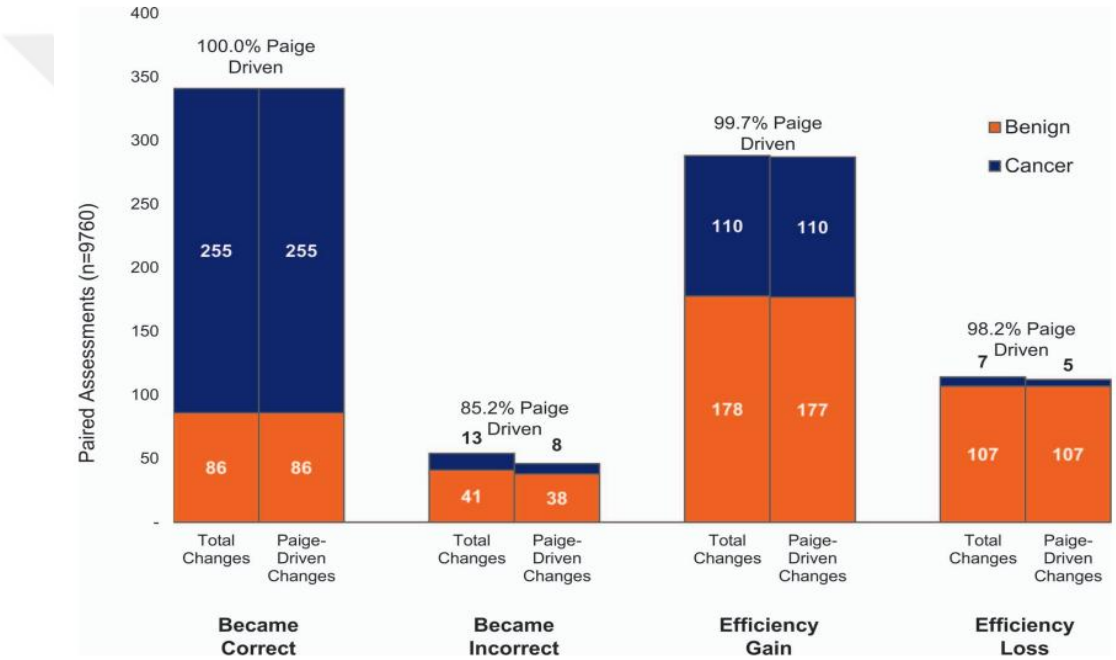
Bu bölümde yapay zeka yöntemlerinin kullanıldığı akademik çalışmalar ve yapay zekanın aktif olarak kullanıldığı örneklerden bahsedilecek olup detaylandırılacaktır.

2.3.1 Page AI

Paige.AI, FDA tarafından onaylanan patolojideki ilk klinik yapay zeka uygulaması olan Paige Prostate'ı geliştirdi. Bu yazılım, patologlara dijital olarak taranmış prostat biyopsi görüntülerinde kanser şüpheli bölgeleri tespit etmede yardımcı olur. Ayrıca, yapılan bir çalışma, Paige Prostate yardımıyla, uzman olmayan kişilerin, yazılım olmadan prostat uzmanları kadar doğru teşhis koyabildiğini gösterdi. Paige Prostate, dünya çapındaki saygın hastaneler ve laboratuvarlar tarafından klinik iş akışlarının bir parçası olarak kullanılmaktadır. Yapay zekanın tıbbi tanıdaki kullanımının, hastalıkların tespitindeki doğruluğu önemli ölçüde artırması, tanı süreçlerinin verimliliğini ve erişilebilirliğini iyileştirmesi beklenmektedir. Bu tür yapay zeka sistemlerinin, patologların ilgili bilgilere daha hızlı ve daha odaklı bir şekilde erişmesini sağlayarak, nihayetinde

daha hassas ve hızlı bir hasta bakımına yol açması amaçlanmaktadır. Patolojide yapay zeka araçlarının kullanımı, tanıların insan uzmanlığı ve ileri teknolojinin birleşimiyle desteklendiği daha kapsamlı ve erişilebilir bir sağlık hizmetine doğru atılmış bir adımdır.

Diğer taraftan, aşırı uyum, yetersiz genelleme yeteneği veya sonuçların yorumlanabilirliğinin eksikliği nedeniyle sorun yaratan yaklaşımlar da vardır. Aşırı uyum, bir yapay zeka modelinin eğitim verilerine çok fazla uyum sağladığında ve yeni, bilinmeyen veriler üzerinde benzer şekilde iyi çalışamadığında ortaya çıkar. Yetersiz genelleme yeteneği, farklı hastalık tabloları, MRI cihazları ve hasta popülasyonları arasında tutarlı bir şekilde çalışabilen modeller geliştirme zorluğunu ifade eder. Yapay zeka kararlarının yorumlanabilirliğinin eksikliği de kritik bir sorundur, çünkü klinisyenlerin yapay zeka tarafından önerilen teşhislerin temelini anlamaları, bu teknolojiye olan güveni artırmak için önemlidir.



Şekil 15: Page AI

Bu çalışma prostatik çekirdek iğne biyopsisi yorumlamasına uygulandığında klinik sınıf YZ araçlarının hasta teşhisi üzerindeki etkisini sağlam bir şekilde ortaya koymaktadır. PaPr'ın etkinliğini ve güvenliğini ortaya koyan bu çalışma, dijital patoloji iş akışı içinde kullanılan sistemin objektif ve sistematik analizini gösteren ve FDA tarafından patolojide bir YZ sistemi için verilen ilk pazarlama izninin temelini oluşturmuştur. FDA onayı rutin klinik çalışmalarda YZ kullanımının önünü açmaktadır; FDA'nın İn Vitro Diagnostik ve Radyolojik Sağlık Ofisi Direktörü Tim Stenzel MD, PhD "Bu YZ tabanlı yazılımın onaylanmasının kanserli doku içeren tanımlanmış prostat biyopsi örneklerinin sayısını artırmaya yardımcı olabileceğini ve sonuçta hayat kurtarabileceğini " belirtilmiştir.

Bu çalışma dünya çapında 218 benzersiz kurumdan gelen vakalardan oluşan zorlu bir veri seti üzerinde PaPr'ın tek başına sağlam performansını göstermektedir; bu da sistemin sağlamlığının H&E boyama ve doku hazırlama değişkenliklerine karşı duyarsızlığının ve genel olarak izlenebilirliğinin bir kanıtıdır. Böylece PaPr'ın yüksek düzeyde performans gösterdiği söylenebilir. Algoritmanın performans özelliklerini korumak için yerinde kalibrasyona ihtiyaç duymadan beklenen performans, diğer benzer araçlardan önemli bir ayırt edici özelliktir. Ayrıca performans hasta yaşı, ırkı ve etnik kökenine göre değerlendirilmiş ve bu değişkenler arasındaki performans farklılıkları anlamlı bulunmamıştır, bu çıktıda bilinmeyen bir yanlılık olmamasını ve bu tür araçların sağlık hizmetlerinde eşitsizliklere katkıda bulunmamasını sağlayan özellikli önemli bir analizdir.

2.3.2. DeepMind

Tümör tespitinde aktif olarak kullanılan yapay zeka (AI) teknolojilerinden biri, Google'ın geliştirdiği "DeepMind" algoritmasıdır. DeepMind, özellikle meme kanseri taramalarında büyük başarı göstermiştir.

DeepMind'in geliştirdiği algoritma, mamogram görüntülerini analiz ederek kanserli hücrelerin tespitinde radyologlara yardımcı oluyor. Bu AI sistemi, büyük veri setleri üzerinde eğitilmiş ve radyologların çalışmalarını hem hızlandırmak hem de doğruluk oranını artırmak amacıyla kullanılıyor. Yapılan testlerde, bu algoritmanın insan radyologlara kıyasla daha yüksek doğruluk oranına sahip olduğu ve yanlış negatif oranlarını azalttığı görülmüştür.

Bu yapay zeka sistemi doktorların daha hızlı ve doğru tanılar koymalarına yardımcı olurken, aynı zamanda hasta bakımını da iyileştirmektedir. Gelişmiş görüntü işleme teknikleri ve makine öğrenimi algoritmaları sayesinde, yapay zeka tıbbi görüntülerdeki anormallikleri tespit etmede giderek daha fazla kullanılmakta ve önem kazanmaktadır.

2.3.2.1 Watson for Oncology

IBM Watson for Oncology, IBM tarafından geliştirilen ve kanser teşhisi ve tedavisinde doktorlara yardımcı olmayı amaçlayan bir yapay zeka platformudur. Bu sistem, geniş bir tıbbi veri yelpazesini analiz ederek kişiselleştirilmiş tedavi önerileri sunar. Watson for Oncology, doktorların karar verme süreçlerini destekleyerek daha hızlı ve doğru teşhisler koymalarına yardımcı olur.

Watson for Oncology, tıbbi literatür, klinik kılavuzlar, tıbbi dergiler ve hasta kayıtları gibi çeşitli veri kaynaklarını analiz ederek, büyük miktarda yapılandırılmış ve yapılandırılmamış veriyi işleyip anlamlı bilgiler elde eder. Bu yapay zeka sistemi, hastanın tıbbi geçmişi, genetik profili ve tümör özellikleri gibi bireysel sağlık verilerini dikkate alarak kişiselleştirilmiş tedavi önerileri sunar ve farklı tedavi seçeneklerinin etkinliği ile olası yan etkileri hakkında bilgi sağlar. Ayrıca, sürekli

güncellenen tıbbi literatür ve klinik araştırmalar sayesinde, doktorlara en güncel ve geçerli bilgileri sunarak kanser tedavisindeki en son gelişmeleri ve yenilikleri takip etme imkanı tanır.

İşeyişi doktorların hastanın sağlık verilerini sisteme girmesiyle başlar; bu veriler arasında hastanın tıbbi geçmişi, genetik test sonuçları, tümör biyopsi sonuçları ve diğer ilgili bilgiler bulunur. Yapay zeka sistemi, bu verileri analiz ederek hastanın durumuna en uygun tedavi seçeneklerini belirler ve farklı tedavi yöntemlerinin avantajları ile potansiyel riskleri hakkında detaylı bilgiler sunar. Doktorlar, Watson'ın önerilerini değerlendirerek hastalar için en uygun tedavi planını oluşturur ve bu sürecin sonunda, Watson'ın sunduğu bilgileri kendi klinik tecrübeleri ve hastanın bireysel durumu ile birleştirerek karar verirler.

Watson for Oncology, özellikle tedavi planlaması, ikinci görüş alma ve eğitim-araştırma alanlarında kullanılır. Tedavi planlamasında, doktorlara hastalara en uygun tedavi planını oluşturmada yardımcı olur. Karmaşık vakalarda, doktorların ikinci bir görüş almasını sağlayarak tedavi süreçlerini destekler. Ayrıca, onkoloji alanında çalışan profesyonellerin eğitiminde ve yeni tedavi yöntemlerinin araştırılmasında da kullanılabilir.

Watson for Oncology, çeşitli sağlık kuruluşlarında başarıyla uygulanmaktadır. Örneğin, Memorial Sloan Kettering Cancer Center ile yapılan iş birliği sayesinde, Watson'ın onkoloji alanındaki bilgisi sürekli güncellenmektedir. Bu iş birliği, Watson'ın kanser tedavisindeki en son bilgileri ve klinik deneyimleri öğrenmesini sağlar ve doktorlara en güncel ve doğru önerileri sunar.

IBM Watson for Oncology, yapay zeka ve büyük veri analizini kullanarak kanser tedavisinde önemli bir yenilik sunar. Doktorların daha hızlı ve doğru kararlar almasına yardımcı olan bu sistem, kanser hastalarının daha iyi tedavi sonuçlarına ulaşmasına katkıda bulunur.

3. MALZEME VE MATERYAL

3.1 Nesne Tanıma ve Algoritmalar

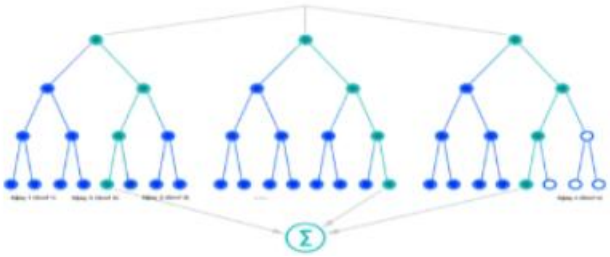
Teknolojinin gelişmesiyle birlikte, kameraların, telefonların ve bilgisayarların kullanımı artmıştır. Bu nedenle, günümüzde video kaydı ve fotoğraf çekimi yaygınlaşmıştır. Dijital ortama aktarılan görüntüler analiz edilmiştir. Bu analizlerin sonuçları, görüntü işleme teknikleri için önemlidir. Kameralar kullanılarak nesneleri birbirinden ayırt etmek için, nesnenin renk, şekil, çevre uzunluğu, köşe sayısı, kenar uzunluğu, ışık yansıması, histogram gibi görsel özellikleri kullanılır. Bu özellikler, görüntü işleme algoritmaları yardımıyla yapay sinir ağı algoritmaları tarafından işlenir. Ayırt edilmesi gereken nesnelerin özellikleri çok farklıysa, yapay sinir ağları kullanılmadan basit seçim algoritmaları ile tanıyabilirler. Ancak, nesne özelliklerinin birbirine yakın olduğu durumlarda, nesnenin özelliklerini değerlendirmek ve yorumlamak için yapay sinir ağlarına ihtiyaç duyulur. Nesnelerin özellikleri birbirine ne kadar benzerse, onları ayırt etmek o kadar zor olur. Yapay sinir ağı teknikleri nesne tanıma ve robotik uygulamalarda yaygın olarak kullanılmaktadır. İnsanlar, nesneleri algılayarak ve bu özellikleri daha önce öğrendikleri bilgilerle karşılaştırarak tanır. Nesne tanıma ve algılama sistemlerinde de benzer yöntemler ve yaklaşımlar kullanılır. İlk olarak tanınacak nesneler için özel veri setleri oluşturulur. Bu veri setleri sisteme öğretilir ve karşılaştırma algoritması ile nesne daha önce öğrenilen bilgilerle karşılaştırılarak tanınır. Bu bölümde, nesne tanıma algoritmaları daha detaylı olarak incelenecektir.

3.1.1 Algoritmalar

Bu bölümde, yapay zeka algoritmaları detaylı bir şekilde tanıtılacak ve her bir algoritmanın çalışma prensipleri ile uygulama alanları açıklanacaktır.

3.1.1.1 RASSAL ORMAN YÖNTEMİ

Rassal Orman algoritması, birden fazla karar ağacının oluşturulması ve bu ağaçların sonuçlarının bir araya getirilerek ortak bir sonuç üretilmesine dayanmaktadır. Şekil 1'de yer alan karar ağaçları, veri kümesindeki özelliklerin belirli bir sıraya göre sıralanarak bir ağaç yapısı oluşturulmasıyla elde edilir. Her bir düğümde belirli bir özellik değeri kontrol edilir ve ağacın dallarına ayrılır. Rassal Orman algoritması, bu işlemi birden fazla karar ağacı ile gerçekleştirir ve her bir ağacın sonucunu bir araya getirerek daha doğru ve stabil bir sonuç üretmeyi hedefler.

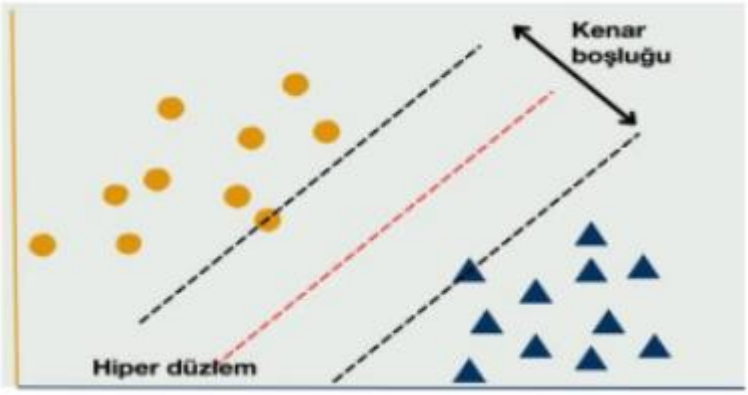


Şekil 16: Rassal Orman Algoritmasının blok gösterimi

Rassal Orman Sınıflandırıcı, Rassal Orman algoritmasını uygulamak için birden fazla karar ağacı oluşturur. Ağaçların sayısı ve diğer parametreler, sınıfın yapıcı (constructor) yöntemi aracılığıyla belirlenir. Oluşturulan ağaçlar, veri kümesindeki özelliklerin belirli bir alt kümesini kullanarak rastgele seçilir, bu sayede her bir ağaç farklı özellikleri kullanarak farklı sonuçlar üretmektedir. [24]

3.1.1.2.DESTEK VEKTÖR MAKİNELERİ

Destek vektör makineleri, sınıflandırma, regresyon ve aykırı değerlerin tespiti için kullanılan bir dizi denetimli öğrenme yöntemidir. Destek vektör makinesi algoritmasının amacı, N boyutlu bir uzayda (N - özellik sayısı) veri noktalarını belirgin bir şekilde sınıflandıran bir hiperdüzlem bulmaktır. Şekil 2'de görüldüğü gibi iki veri noktası sınıfını ayırmak için seçilebilecek birçok olası hiperdüzlem vardır. Amacımız maksimum kenar boşluğuna, yani her iki sınıfın veri noktaları arasındaki maksimum mesafeye sahip bir düzlem bulmaktır. Kenar boşluğu mesafesini maksimuma çıkarmak, gelecekteki veri noktalarının daha güvenle sınıflandırılabilmesi için bir miktar güçlendirme sağlar. [24]



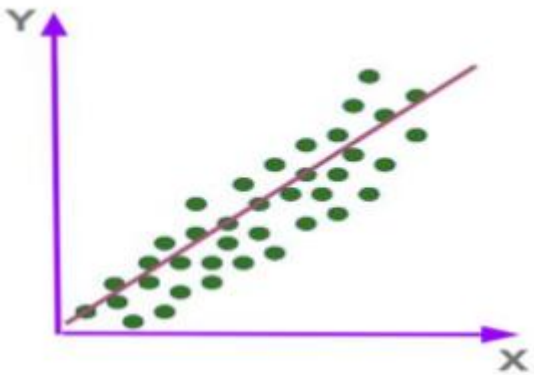
Şekil 17: Destek Vektör Makineleri çalışma şekli

3.1.1.3 DOĞRUSAL - DVM

DVM algoritması, bir sınıflandırma problemi için veri kümesindeki örnekleri bir hiperdüzlemle iki sınıfa ayırmayı hedeflerken Doğrusal-DVM algoritması ise, bu hiperdüzlemi Doğrusal bir fonksiyonla ifade etmektedir. Doğrusal-DVM algoritmasının temelinde yatan optimizasyon problemi olan çift yönlü kararlılık yöntemini kullanır. Bu yöntem, veri kümesindeki örnekleri ikisınıfa ayırmak için bir hiperdüzlem bulurken, bu hiperdüzlemi en iyi şekilde ayıracak olan destek vektörlerini bulmaya çalışır. Bu işlem, hiperdüzlemi tanımlayan ağırlık vektörü ve bias terimini tahmin etmek için kullanılır. Doğrusal-DVM algoritması, bir hiperdüzlemi bulmak için örnekleri iki sınıfa ayırmaya çalışırken, ayrılabilir bir veri kümesi için doğrudan çözüm bulabilir. Ancak, eğer veri kümesi ayrılabilir değilse, DVM algoritması yumuşatmalı sınıflandırma adı verilen bir yöntemle çalışır. Bu yöntem, bazı örneklerin hiperdüzlemin ötesinde yer almasına izin verir, böylece DVM algoritması daha iyi bir sınıflandırma performansı elde eder. Özetle, Doğrusal-DVM algoritması, veri kümesindeki örnekleri iki sınıfa ayırmak için bir hiperdüzlem bulmaya çalışırken DVM algoritmasının dual-formulation yöntemini kullanarak, hiperdüzlemi en iyi şekilde ayıracak olan destek vektörlerini bulmaya çalışmaktadır. [24]

3.1.1.4 DOĞRUSAL REGRESYON-DR

Doğrusal regresyon, bir bağımsız değişken (X) ile bir veya daha fazla bağımlı değişken (Y) arasındaki doğrusal ilişkiyi modellemektedir. Doğrusal regresyon, en uygun bir doğru veya düzlem (eğer çoklu değişkenler varsa) çizerek, verilerin değişkenler arasındaki ilişkisini en iyi şekilde temsil eden Şekil 3'deki gibi bir model oluşturur. Burada yeşil noktalar verilerin konumu kırmızı doğrusal çizgi regresyon çizgisidir.



Şekil 18: Doğrusal regresyon grafiği

Bu çalışmadaki Doğrusal regresyon sınıflandırıcısı, Python programlama kütüphanesinde yer alan "Linear Regression" sınıfı kullanılarak uygulanmıştır. Bu sınıf, doğrusal regresyon için temel özellikleri ve parametreleri içermektedir ve regresyon modeli oluşturmak için veriler üzerinde en küçük kareler yöntemini kullanır. Doğrusal regresyon, veri kümesindeki değişkenler arasındaki ilişkiyi modellerken, hataların en aza indirilmesi prensibine dayanır. Bu nedenle, doğrusal regresyon, en küçük kareler yöntemi olarak da adlandırılmaktadır. Bu model, veri noktaları arasındaki doğrusal ilişkiyi temsil eden bir doğru ile ifade edilir. Bu doğru, eğitim verilerindeki hataların en aza indirilmesi prensibine göre en iyi uyumu sağlar. Eğitim verilerindeki özellikler, Şekil 16'de görüleceği üzere sınıflandırıcının X girdisi olarak kullanılırken, etiketler (pozitif ve negatif tümör durumları) sınıflandırıcının Y çıktısı olarak kullanılmıştır. [24]

3.1.1.5 CHİ-KARE

Chi-Kare (Chi-Square) testi, kategorik verilerin analizinde kullanılan bir istatistiksel testtir. Bu test, iki kategorik değişken arasındaki ilişkinin varlığını veya yokluğunu belirlemek için kullanılmaktadır. Çalışma içinde, Chi-Kare sınıflandırıcısı özellik çıkarımı için Select KBest sınıfı kullanılarak uygulanmıştır. Bu sınıf, verilerden en önemli özellikleri seçmek için kullanılır. Özellik seçimi, sınıflandırıcı için en anlamlı ve önemli olan özelliklerin seçilmesi amacıyla yapılır ve sınıflandırma doğruluğunu artırmayı hedefler.Select KBest sınıfı, veri özelliklerinin önemini, istatistiksel bir test olan Chi-kare testini kullanarak belirler. Bu test, her özelliğin her sınıftaki frekanslarını ve beklenen frekanslarını karşılaştırarak, sınıf etiketleriyle ilişkisini ölçer. Sonuç olarak, özellikler sıralanır ve en önemli olanları seçmek için bir kriter belirlenir. [24]

3.1.1.6 HOG

Yönlendirilmiş Gradyan Histogramı (HOG), bilgisayarlı görüş ve nesne tanıma alanında yaygın olarak kullanılan bir özellik çıkarım yöntemidir. Bu yöntem, özellikle nesne tanıma ve insan algılama gibi uygulamalarda kullanılır. HOG, görüntüleri bloklara böler, her bloktaki kenar ve gradient bilgilerini hesaplar, ardından bu bilgileri bir özellik vektörü olarak birleştirir. Bu çalışmada, HOG özellikleri kullanılarak nesne tanıma yapılacaktır. HOG, görüntülerdeki kenarları ve yapıları yakalamak için kullanılan bir özellik vektörüdür. Bu özellik vektörü, görüntülerin kenar ve gradient bilgilerini temsil eder. HOG özellikleri genellikle nesne tanıma sistemlerinde ve insan algılama uygulamalarında kullanılır. [24]

3.2 Kaggle

Kaggle, bir veri bilimci ve makine öğrenme tutkunları için popüler bir platformdur. Kaggle, 2010 yılında başlatılan ve 2017 yılında Google tarafından satın alınan bir platform olarak, kullanıcıların

veri setlerini keşfetmelerine, paylaşmalarına ve üzerinde çalışma yaparak projeler geliştirmelerine olanak sağlamaktadır. Kaggle, veri bilim yarışmaları, veri setleri ve eğitim kaynakları ve tartışma panoları gibi çok çeşitli özellikler sunar. Bu özellikler, kullanıcıların yeteneklerini geliştirmelerine ve gerçek dünya problemlerini çözmelerine yardımcı olur; Kaggle, kullanıcıların hackathonları dijital veya meta bir yarışma olarak kullandıkları ortak bir platformdur.

Kaggle, çeşitli şirketler ya da araştırma kurumlarınca düzenlenen ve dünyanın birçok yerinde yapılan veri bilimi yarışmalarına; büyük veya küçük ölçekteki katkıları sayesinde ev sahipliği yapmaktadır. Aslında, bunları katılımcılara kendilerini belirli bir veri seti üzerinde en iyi model dahilinde göstermeleri için platformlar şeklinde düşünebilirsiniz. Bu yarışmaların ödülleri genellikle para ve altın kariyer fırsatları olarak protokol edilmektedir. Ayrıca, kullanıcılar onlarca veri seti hakkında bilgilere ve bu veri setlerini projelerinde kullanmalarına ücretsiz erişebilme fırsatına sahiptir. Kaggle kullanıcıların kendi veri setlerini yükleyip paylaşmalarına olanak tanır.

Python ve R gibi dillerde kod yazarak veri analizi yapabilmesine olanak tanıyan çevrimiçi çalışma ortamları, Kaggle aracılığıyla kullanılabilir. Bu not defterleri kullanıcıların analizlerini paylaşmalarına ve diğer kullanıcıların kodlarını incelemelerine olanak tanır. Ayrıca, Kaggle veri bilimi ve makine öğrenimi konularında çeşitli kurslar ve öğreticiler sunmaktadır. Bu kaynakların herkese uygundur ve yeni başlayanlardan ileri düzey kullanıcılara kadar eğitim dersleri sağlar. Kaggle, projelerini ve fikirlerini paylaşabileceğiniz aktif bir topluluğa sahiptir. Ayrıca, Tartışma panoları ve blog yazıları aracılığıyla bilgi alışverişi yapabilirsiniz.

3.3 Google Colab

Google Colab, Python ve R programlama dilleri kullanarak makine öğrenimi ve veri analizi projeleri gerçekleştirmek için kullanılan ücretsiz bir çevrimiçi not defteri ortamıdır. Google tarafından sunulan bu hizmet, kullanıcılara güçlü donanım kaynaklarına (CPU, GPU, TPU) erişim imkanı tanır ve bu sayede karmaşık modelleri eğitmek ve büyük veri setlerini işlemek daha kolay hale gelir.

Google Colab, Jupyter Notebook tabanlıdır ve kullanıcıların kod, metin, görseller ve matematiksel ifadeler içeren interaktif belgeler oluşturmasına olanak tanır. Bu not defterleri, Google Drive'da saklanabilir ve paylaşılabilir. Colab, kullanıcılara ücretsiz olarak GPU ve TPU kaynaklarına erişim imkanı sunar. Bu özellik, makine öğrenimi modellerini eğitmek için büyük bir avantaj sağlar. Ayrıca, Google Colab, Google Drive ve GitHub ile kolayca entegre edilebilir. Kullanıcılar, Drive'daki dosyalarına erişebilir ve GitHub üzerindeki projeleri doğrudan Colab'da açıp düzenleyebilirler.

Google Colab, TensorFlow, Keras, PyTorch, OpenCV gibi birçok popüler makine öğrenimi ve veri bilimi kütüphanesi ile önceden yüklü olarak gelir. Bu, kullanıcıların hızlı bir şekilde projelerine başlamalarını sağlar. Colab not defterleri, tıpkı Google Dokümanlar gibi, başkalarıyla kolayca paylaşılabilir ve aynı anda birden fazla kişi tarafından düzenlenebilir. Bu, ekip çalışmasını ve

işbirliğini kolaylaştırır. Özellikle eğitim kurumları ve öğrenciler için ideal bir araç olan Colab, öğrencilerin not defterlerini öğretmenleriyle paylaşmalarına ve anında geri bildirim almalarına olanak tanır.

Bu iki araç, veri bilimi ve makine öğrenimi alanında çalışan araştırmacılar ve profesyoneller için vazgeçilmez kaynaklar sunar. Kaggle, topluluk desteği ve yarışmaları ile öne çıkarken, Google Colab ise güçlü donanım desteği ve kolay entegrasyon özellikleri ile dikkat çeker.

3.4 Python

Python, veri bilimi, yapay zeka, web geliştirme ve daha birçok alanda yaygın olarak kullanılan güçlü ve esnek bir programlama dilidir. Guido van Rossum tarafından 1991 yılında geliştirilen Python, sadeliği ve okunabilirliği ile tanınmaktadır [15]. Bu özellikler, hem yeni başlayanlar hem de deneyimli programcılar için Python'u ideal bir seçim haline getirir.

Python, geniş ve aktif bir topluluğa sahiptir, bu da dilin sürekli olarak geliştirilmesini ve desteklenmesini sağlar. Python'un standart kütüphanesi, dosya işleme, veri analizi, web hizmetleri ve daha birçok konuda kapsamlı araçlar sunar [30]. Bunun yanı sıra, Python, çeşitli üçüncü taraf kütüphaneleri ve paketleri ile de desteklenmektedir. Bu kütüphaneler, bilimsel hesaplamalar, veri görselleştirme ve makine öğrenimi gibi belirli alanlarda programcıların işini kolaylaştırır ve projelerini hızlı bir şekilde geliştirmelerine olanak tanır [32].

Veri bilimi ve yapay zeka alanında Python, Pandas, NumPy, SciPy, Scikit-Learn, TensorFlow, Keras ve PyTorch gibi güçlü kütüphaneleri ile öne çıkar. Pandas, veri işleme ve analizinde yaygın olarak kullanılırken, NumPy ve SciPy, sayısal hesaplamalar için güçlü araçlar sunar. Scikit-Learn, makine öğrenimi modellerini eğitmek ve değerlendirmek için kapsamlı bir araç seti sağlarken, TensorFlow ve PyTorch, derin öğrenme projelerinde yaygın olarak kullanılır [19].

Sonuç olarak, Python, sadeliği, esnekliği ve geniş kütüphane desteği ile birçok farklı alanda yaygın olarak kullanılan bir programlama dilidir. Veri bilimi, yapay zeka, web geliştirme ve otomasyon gibi çeşitli alanlarda güçlü araçlar sunar ve geniş topluluğu sayesinde sürekli olarak gelişmektedir. Bu özellikler, Python'u hem yeni başlayanlar hem de deneyimli programcılar için ideal bir dil haline getirmektedir [13].

3.4.1 Kütüphaneler

Yapay zeka projelerinde, veri işleme, model oluşturma, eğitim ve görselleştirme gibi adımlarda çeşitli kütüphaneler kullanılmaktadır. Bu kütüphaneler, projelerin etkin ve verimli bir şekilde

yürütülmesini sağlar. Her bir kütüphane, belirli işlevler ve avantajlar sunarak projelerin başarılı bir şekilde tamamlanmasını sağlar. TensorFlow ve Keras, güçlü modelleme yetenekleri ile derin öğrenme projelerinin temel taşlarını oluştururken, Pandas ve NumPy, veri işleme ve analizi süreçlerinde etkin çözümler sunar. Matplotlib ve Seaborn ise, görselleştirme araçları ile model performansının ve veri analizinin görselleştirilmesine katkıda bulunur. Bu kütüphanelerin açıklamaları ve yapay zeka projelerindeki rolleri detaylandırılmıştır.

3.4.1.1 TensorFlow

Google tarafından geliştirilen açık kaynak kodlu bir derin öğrenme kütüphanesidir. TensorFlow, makine öğrenimi modellerini oluşturma, eğitme ve dağıtma için esnek ve kapsamlı bir ekosistem sunar. Yüksek performanslı hesaplamaları destekleyen TensorFlow, derin sinir ağlarının eğitiminde yaygın olarak kullanılır. TensorFlow, büyük veri kümeleri üzerinde paralel hesaplamalar yaparak verimli model eğitimi sağlar ve GPU hızlandırmasını destekler. Ayrıca, TensorFlow Lite ile mobil ve gömülü cihazlarda makine öğrenimi modellerinin dağıtımı da mümkündür.

TensorFlow'un ana bileşenleri arasında TensorFlow Core ve yüksek seviyeli API'ler bulunmaktadır. TensorFlow Core, düşük seviyeli matematiksel işlemler ve hesaplamalar yapmayı sağlar. Bu bileşen, esneklik ve kontrol gerektiren durumlar için idealdir. Yüksek seviyeli API'ler, kullanıcıların daha hızlı ve kolay bir şekilde model geliştirmelerini sağlar. TensorFlow, geniş bir topluluk ve ekosistem tarafından desteklenir ve sürekli olarak güncellenir.

TensorFlow, çeşitli veri türleri ve uygulama alanları için geniş bir araç seti sunar. Örneğin, doğal dil işleme, görüntü işleme, zaman serisi analizi ve tavsiye sistemleri gibi alanlarda kullanılır. TensorFlow, aynı zamanda TensorFlow Extended (TFX) gibi araçlarla uçtan uca makine öğrenimi çözümleri sunar. TFX, veri hazırlama, model eğitimi, değerlendirme ve dağıtım süreçlerini otomatikleştirir ve optimize eder.

3.4.1.2 Keras

Keras, TensorFlow'un üstünde çalışan yüksek seviyeli bir neural network kütüphanesidir. Kullanıcı dostu ve modüler yapısı sayesinde, hızlı prototipleme ve model geliştirme süreçlerinde tercih edilir. Keras, sekansiyel ve fonksiyonel API'leri ile karmaşık modellerin oluşturulmasını kolaylaştırır. Ayrıca, Keras, eğitim, doğrulama ve değerlendirme işlemlerini basitleştirir. TensorFlow ile entegre çalışması sayesinde, Keras kullanıcıları TensorFlow'un tüm avantajlarından faydalanabilirler.

Keras, çeşitli katman türlerini, aktivasyon fonksiyonlarını, kayıp fonksiyonlarını ve optimizasyon algoritmalarını destekler. Bu esneklik, araştırmacıların ve mühendislerin deneysel modelleri hızlı

bir şekilde geliştirmelerine olanak tanır. Keras'ın sekansiyel API'si, katmanları sıralı bir şekilde ekleyerek model oluşturmayı sağlar. Fonksiyonel API ise daha karmaşık yapılar ve yönlendirilmiş asiklik grafikler (DAG) oluşturmak için kullanılır.

Keras, aynı zamanda transfer öğrenme ve ince ayar gibi teknikleri destekler. Transfer öğrenme, önceden eğitilmiş modellerin yeniden kullanılarak yeni bir görev için adapte edilmesini sağlar. Bu, özellikle büyük veri setlerine erişimi olmayan durumlarda büyük bir avantaj sağlar. Keras, ModelCheckpoint ve EarlyStopping gibi callback'ler ile model eğitim sürecini daha verimli hale getirir. Bu callback'ler, modelin performansını izleyerek en iyi modeli kaydeder ve gereksiz eğitim adımlarını önler.

3.4.1.3 Pandas

Veri manipülasyonu ve analizi için en çok kullanılan kütüphanelerden biridir. Pandas, tablo verileri (DataFrame) ve diziler (Series) ile çalışmayı kolaylaştırır. Veri temizleme, dönüştürme, birleştirme ve özetleme gibi işlemleri etkin bir şekilde gerçekleştirmeye olanak tanır. Yapay zeka projelerinde, veri setlerinin ön işlenmesi, Pandas kütüphanesi kullanılarak yapılır. Pandas, verilerin hızlı ve kolay bir şekilde analiz edilmesini sağlayarak, model eğitimi için temiz ve düzenli veri setleri oluşturulmasına yardımcı olur.

3.4.1.4 NumPy

NumPy, büyük, çok boyutlu dizi ve matrislerle çalışmayı kolaylaştıran bir kütüphanedir. NumPy, sayısal hesaplamalar için çeşitli matematiksel fonksiyonlar ve istatistiksel araçlar sunar. Yapay zeka projelerinde, verilerin vektörleştirilmesi, matris işlemleri ve lineer cebir hesaplamaları gibi işlemler NumPy kullanılarak gerçekleştirilir. NumPy'nin yüksek performanslı hesaplamalar yapabilme yeteneği, büyük veri setleri ve karmaşık modellerle çalışırken büyük avantaj sağlar.

3.4.1.5 Matplotlib

Matplotlib, veri görselleştirme için kullanılan geniş kapsamlı bir kütüphanedir. Çizgi grafikleri, çubuk grafikleri, histogramlar ve daha birçok görsel öğeyi oluşturmak için kullanılır. Yapay zeka projelerinde, model performansını değerlendirmek ve analiz etmek için veri görselleştirme kritik öneme sahiptir. Matplotlib, eğitim ve doğrulama kayıplarını, doğruluklarını ve diğer metrikleri görselleştirmek için yaygın olarak kullanılır. Bu sayede, modelin performansı hakkında daha iyi bir anlayış elde edilir ve gerekli iyileştirmeler yapılabilir.

3.4.1.6 Seaborn

Seaborn, Matplotlib tabanlı bir veri görselleştirme kütüphanesidir ve estetik açıdan daha çekici ve bilgilendirici grafikler oluşturmayı amaçlar. Seaborn, özellikle istatistiksel verilerin görselleştirilmesinde güçlüdür ve korelasyon haritaları, dağılım grafikleri, kutu grafikleri gibi karmaşık grafiklerin kolayca oluşturulmasını sağlar. Yapay zeka projelerinde, veri keşfi ve analiz sürecinde Seaborn kullanılarak verilerin görselleştirilmesi, veri setindeki ilişkilerin ve dağılımların daha iyi anlaşılmasına yardımcı olur.

3.5 Görüntü İşleme Metotları

Görüntü işleme, makine öğrenimi ve derin öğrenme modellerinin performansını iyileştirmek ve genelleme yeteneklerini artırmak için kritik öneme sahip bir alandır. Bu teknikler, eğitim verilerini çeşitli dönüşümler ve işlemlerle zenginleştirerek daha geniş ve çeşitli bir veri seti oluşturmayı hedefler. Görüntü işleme, özellikle derin öğrenme tabanlı görüntü sınıflandırma, nesne tanıma ve segmentasyon gibi görevlerde önemli bir rol oynar.

Görüntü işleme metotları, görüntü artırma tekniklerinde yaygın olarak kullanılır. Görüntü artırma, mevcut veri setini zenginleştirerek modelin farklı koşullar ve varyasyonlar altında daha iyi performans göstermesini sağlar. Bu teknikler, modelin overfitting (aşırı öğrenme) yapma riskini azaltır ve modelin daha iyi genelleme yapmasını sağlar. Aşağıda, görüntü işleme metotlarının detaylı bir açıklaması ve bu işlemler için kullanılan Python kütüphaneleri bulunmaktadır.

Görüntülerin yatay ve dikey eksenlerde çevrilmesi, veri setine simetri ekleyerek modelin yönelime duyarlılığını azaltır. Yatay çevirme, görüntüdeki her pikseli sağdan sola doğru yer değiştirirken, dikey çevirme, pikselleri yukarıdan aşağıya doğru taşır. Bu dönüşümler, özellikle görüntüdeki nesnelerin simetrik olduğu durumlarda modelin performansını artırabilir. Örneğin, bir kedi resmi yatay çevrildiğinde, model yine de bu resmi tanıyabilmelidir. Bu işlemler için kullanılan Python kütüphaneleri arasında TensorFlow ve Keras bulunur.

Parlaklık değiştirme, bir görüntünün genel aydınlatma seviyesini değiştirerek yapılır. Bu işlem, her pikselin parlaklık değerine belirli bir miktar ekleyerek veya çıkararak gerçekleştirilir. Bu tür bir dönüşüm, modelin farklı aydınlatma koşulları altında daha sağlam olmasını sağlar. Örneğin, düşük ışıkta çekilmiş bir görüntü, parlaklık artırılarak daha net hale getirilebilir ve modelin bu farklı aydınlatma seviyelerinde doğru tahminler yapması sağlanabilir. Bu işlem genellikle TensorFlow ve Keras kütüphanelerindeki ImageDataGenerator sınıfı ile gerçekleştirilir.

Kontrast değiştirme, bir görüntünün parlaklık değerlerini ortalamadan uzaklaştırarak veya yaklaştırarak gerçekleştirilir. Bu işlem, görüntünün genel görünümünü daha keskin veya daha yumuşak hale getirir. Kontrast artırma, görüntüdeki nesnelerin daha belirgin hale gelmesini

sağlarken, kontrast azaltma daha yumuşak geçişler elde etmeye yardımcı olabilir. Bu tür dönüşümler, modelin farklı kontrast seviyelerinde nesneleri tanıyabilme yeteneğini artırır. Keras ve TensorFlow bu işlemler için sıkça kullanılır.

Doygunluk ayarı, renkli bir görüntünün renk yoğunluğunu artırarak veya azaltarak gerçekleştirilir. Bu işlem, görüntüdeki renklerin daha canlı veya daha soluk görünmesini sağlar. Doygunluk artırma, renklerin daha parlak ve doygun hale gelmesine yardımcı olurken, doygunluk azaltma, pastel tonlar elde etmeyi sağlar. Bu tür dönüşümler, modelin farklı renk yoğunlukları altında performansını iyileştirir. TensorFlow'un tf.image modülü bu tür işlemler için uygundur.

Renk tonu değiştirme, bir görüntünün renk bileşenlerini belirli bir açı etrafında döndürerek yapılır. Bu işlem, görüntünün genel renk tonunu değiştirir. Örneğin, kırmızı renk tonuna sahip bir görüntü, renk tonu değiştirilerek mavi veya yeşil tonlarına kaydırılabilir. Bu dönüşüm, modelin farklı renk varyasyonlarına karşı daha esnek olmasını sağlar ve renklerin farklı tonlarını tanıyabilme yeteneğini artırır. Bu işlemler de TensorFlow ve Keras kütüphaneleri ile gerçekleştirilebilir.

Görüntü döndürme, belirli bir açı etrafında görüntünün tüm piksellerinin yer değiştirmesi işlemidir. Bu dönüşüm, modelin farklı açılardan gelen görüntüleri tanıyabilme yeteneğini artırır. Örneğin, bir nesne 30 derece döndürülse bile modelin bu nesneyi tanıyabilmesi gereklidir. Döndürme işlemi, özellikle nesnelerin farklı açılardan çekilmiş görüntüleriyle çalışırken modelin daha esnek olmasını sağlar. Bu işlem de yine Keras ve TensorFlow kütüphaneleri ile yapılabilir.

Görüntünün belirli bir oranla yakınlaştırılması veya uzaklaştırılması, ölçekleme dönüşümü kullanılarak yapılır. Bu işlem, modelin farklı mesafelerden gelen görüntüleri tanıyabilmesine yardımcı olur. Yakınlaştırma, nesnelerin detaylarını daha belirgin hale getirirken, uzaklaştırma, nesnelerin genel görünümünü koruyarak daha geniş bir perspektif sunar. Bu dönüşümler, modelin hem detayları hem de genel yapıyı öğrenmesini sağlar. TensorFlow ve Keras'ta bu tür işlemler ImageDataGenerator sınıfı ile uygulanabilir.

Görüntülerin yatayda ve/veya dikeyde taşınması, tüm piksellerin belirli bir mesafeyle kaydırılmasıyla yapılır. Bu işlem, nesnelerin farklı pozisyonlarda tanınmasını sağlar. Örneğin, bir nesne görüntünün merkezinden kenarına taşındığında bile modelin bu nesneyi tanıyabilmesi gereklidir. Taşıma işlemi, modelin nesnelerin farklı konumlarını öğrenmesine yardımcı olur. Bu tür dönüşümler, Keras ve TensorFlow'da sıkça kullanılır.

Kesme, bir görüntünün belirli bir yönde kaydırılması işlemidir. Bu işlem, görüntüdeki nesnelerin şeklini bozmadan belirli bir açıda kaydırılmasını sağlar. Kesme, özellikle perspektif değişikliklerini simüle ederken kullanılır ve modelin deformasyonlara karşı daha dayanıklı olmasını sağlar. Bu tür dönüşümler, modelin nesnelerin farklı şekillerde ve açılarda tanınmasını öğrenmesini sağlar. Kesme işlemi için de Keras ve TensorFlow kütüphaneleri kullanılabilir.

3.6 Beyin Tümörü Tespitinde Kullanılan Yapay Zeka Yöntemleri

Bu bölümde, beyin tümörü tespitinde kullanılan yapay zeka yöntemlerini detaylı bir şekilde ele alacağız. Yapay zeka ve derin öğrenme teknikleri, tıbbi görüntülemede devrim niteliğinde gelişmeler sağlamış ve beyin tümörlerinin tespit ve sınıflandırılmasında yaygın olarak kullanılmaya başlanmıştır. Bu bölümde, çeşitli yapay zeka modellerinin performans, doğruluk oranları ve eğitim sürelerini karşılaştıracağız. Ayrıca, bu modellerin klinik uygulamalardaki etkinliklerini ve pratik kullanım avantajlarını değerlendireceğiz. Çeşitli metrikler kullanarak, modellerin birbirlerine göre üstünlüklerini ve zayıf yönlerini ortaya koymayı amaçlıyoruz. Bu analiz, beyin tümörü tespiti için en uygun yapay zeka yöntemlerinin belirlenmesine katkı sağlayacaktır.

3.6.1. Konvolüsyonel Sinir Ağları (CNN)

Konvolüsyonel Sinir Ağları (CNN), görüntü işleme ve özellikle tıbbi görüntü analizi alanında oldukça etkili olan derin öğrenme modelleridir. CNN'ler, beyin tümörlerinin tespit ve sınıflandırılmasında yaygın olarak kullanılmaktadır. Bu ağlar, beyin MRI görüntülerindeki karmaşık özellikleri otomatik olarak öğrenebilir ve yüksek doğruluk oranları ile sınıflandırma yapabilir.

3.6.1.1 CNN'in Yapısı ve Çalışma Prensipleri

CNN'ler, tipik olarak üç ana bileşenden oluşur: konvolüsyonel katmanlar, havuzlama katmanları ve tam bağlantılı katmanlar.

Konvolüsyonel Katmanlar (Convolutional Layers): Bu katmanlar, görüntüdeki özellikleri çıkarmak için filtreler (kernels) kullanır. Her filtre, görüntü üzerinde kaydırılarak belirli özellikleri (örneğin kenarlar, dokular) algılar ve bu özellikleri çıkaran bir özellik haritası (feature map) oluşturur. Konvolüsyonel katmanlar, çok katmanlı yapılarında görüntünün farklı seviyelerdeki özelliklerini öğrenirler.

Havuzlama Katmanları (Pooling Layers): Bu katmanlar, konvolüsyonel katmanlar tarafından çıkarılan özellik haritalarının boyutunu azaltır ve modelin hesaplama maliyetini düşürür. En yaygın kullanılan havuzlama yöntemi max-pooling'dir, bu yöntem her havuzlama penceresindeki maksimum değeri seçer.

Tam Bağlantılı Katmanlar (Fully Connected Layers): Bu katmanlar, geleneksel yapay sinir ağı katmanları gibidir ve sınıflandırma görevlerini gerçekleştirir. Özellik haritaları düzleştirilerek

(flattening) bu katmanlara verilir ve softmax aktivasyon fonksiyonu ile her sınıf için olasılık değerleri hesaplanır.

3.6.1.2 CNN Kullanarak Beyin Tümörü Sınıflandırma

Beyin tümörü sınıflandırmasında CNN'ler, MRI görüntülerindeki tümörleri tanımlayabilir ve sınıflandırabilir. Bu süreç, genellikle şu adımları içerir:

Veri Ön İşleme (Data Preprocessing): MRI görüntüleri, CNN modeline uygun hale getirilmek üzere boyutlandırılır, normalize edilir ve veri artırma (data augmentation) teknikleri ile çeşitlendirilir. Bu adımlar, modelin daha iyi genelleme yapabilmesini sağlar.

Model Eğitimi (Model Training): CNN modeli, eğitim veri seti üzerinde eğitilir. Eğitim süreci boyunca model, doğru sınıflandırma yapabilmesi için ağırlıklarını ayarlar. Eğitim süreci, birçok epoch boyunca devam eder ve her epoch'ta modelin performansı değerlendirilir.

Model Değerlendirmesi (Model Evaluation): Eğitilen model, test veri seti üzerinde değerlendirilir. Performans metrikleri arasında doğruluk (accuracy), kesinlik (precision), geri çağırma (recall) ve F1 skoru yer alır. Bu metrikler, modelin genel performansını ve doğruluğunu değerlendirir. [7][26]

3.6.1.3 CNN'in Avantajları ve Dezavantajları

Avantajları:

- Yüksek Doğruluk: CNN'ler, karmaşık görüntü verilerinde yüksek doğruluk oranları ile sınıflandırma yapabilir.
- Otomatik Özellik Çıkarma: CNN'ler, insan müdahalesi olmadan görüntülerdeki önemli özellikleri otomatik olarak öğrenir.
- Esneklik: Farklı tıbbi görüntüleme modalitelerinde (örneğin MRI, CT) kullanılabilir.

Dezavantajları:

- Hesaplama Maliyeti: CNN'ler, büyük veri setleri üzerinde eğitilirken yüksek hesaplama gücü gerektirir.
- Veri Gereksinimi: Etkili bir şekilde öğrenebilmek için büyük miktarda etiketlenmiş veri gerektirirler.

3.6.2 Tekrarlayan Sinir Ağları (RNN)

Tekrarlayan Sinir Ağları (RNN), zaman serisi verileri ve ardışık veri analizleri için özel olarak tasarlanmış derin öğrenme modelleridir. RNN'ler, özellikle ardışık verilerin önemli olduğu alanlarda, örneğin dil modelleme, ses tanıma ve zaman serisi tahmini gibi uygulamalarda yaygın olarak kullanılmaktadır. Bu bölümde, RNN'lerin beyin tümörü tespiti ve sınıflandırılmasındaki kullanımlarını inceleyeceğiz.

3.6.2.1 RNN'in Yapısı ve Çalışma Prensipleri

RNN'ler, önceki zaman adımlarından gelen bilgiyi kullanarak bir sonraki zaman adımındaki tahmini yapabilen ağlardır. Bu, özellikle ardışık verilerdeki bağımlılıkların modellenmesinde oldukça etkilidir.

Gizli Durum (Hidden State): RNN'ler, bir gizli durum vektörü (hidden state) kullanarak önceki zaman adımlarından gelen bilgiyi taşır. Bu gizli durum, her adımda güncellenir ve sonraki adımda kullanılır.

Ağ Yapısı: RNN'in her bir katmanı, önceki adımın gizli durumu ve mevcut girdiyi kullanarak yeni bir gizli durum ve çıktı üretir. Bu, zaman adımlarının birbirine bağlı olduğu bir yapı oluşturur.

Geri Yayılım (Backpropagation Through Time - BPTT): RNN'lerin eğitimi, geri yayılım algoritması kullanılarak gerçekleştirilir. BPTT, zaman adımları boyunca hataların geri yayılmasını sağlar. [17]

3.6.2.2 RNN Kullanarak Beyin Tümörü Sınıflandırma

Beyin tümörü sınıflandırmasında RNN'ler, özellikle ardışık MRI görüntülerindeki zaman serisi verilerini analiz edebilir. Bu süreç, genellikle şu adımları içerir:

Veri Ön İşleme (Data Preprocessing): MRI görüntüleri, RNN modeline uygun hale getirilmek üzere zaman serisi verisi olarak düzenlenir. Bu adım, her bir hastanın ardışık MRI taramalarının zaman serisi şeklinde düzenlenmesini içerir.

Model Eğitimi (Model Training): RNN modeli, eğitim veri seti üzerinde eğitilir. Eğitim süreci boyunca model, ardışık verilerdeki kalıpları öğrenir ve doğru sınıflandırma yapabilmek için ağırlıklarını ayarlar.

Model Değerlendirmesi (Model Evaluation): Eğitilen model, test veri seti üzerinde çeşitli metrikler kullanılarak değerlendirilir. Performans metrikleri arasında doğruluk (accuracy), kesinlik (precision), geri çağırma (recall) ve F1 skoru yer alır.[28]

3.6.2.3 RNN'in Avantajları ve Dezavantajları

Avantajları:

- **Zaman Serisi Verileri**: RNN'ler, zaman serisi verilerindeki bağımlılıkları etkili bir şekilde modelleyebilir.
- **Bağlam Bilgisi**: Gizli durum vektörleri sayesinde, önceki adımlardan gelen bilgiyi kullanarak daha doğru tahminler yapabilir.

Dezavantajları:

- **Uzun Vadeli Bağımlılıklar**: RNN'ler, uzun vadeli bağımlılıkları modellemekte zorlanabilir ve bu nedenle LSTM (Long Short-Term Memory) ve GRU (Gated Recurrent Unit) gibi geliştirilmiş RNN türleri geliştirilmiştir.
- **Eğitim Zorlukları**: BPTT algoritması, uzun ardışık verilerdeki gradyan sönümlenmesi ve patlaması problemlerine yol açabilir.

3.6.2.4 RNN Kullanarak Beyin Tümörü Tespiti

RNN'lerin beyin tümörü tespitinde kullanımı, ardışık MRI taramalarının analiz edilmesi ve tümörlerin zaman içindeki gelişiminin izlenmesi için uygundur. RNN modelleri, her bir zaman adımındaki MRI taramasını analiz eder ve tümörün boyutunu, konumunu ve gelişimini tahmin edebilir.

3.6.3 Üretici Çekişmeli Ağlar (GAN)

Üretici Çekişmeli Ağlar (GAN), iki sinir ağının birbirine karşı yarıştığı bir yapay zeka yöntemidir. Bu yöntem, özellikle veri üretimi ve veri artırma konularında oldukça etkili olup, tıbbi görüntülemede de önemli uygulama alanlarına sahiptir. GAN'ler, bir "üretici" (generator) ve bir "ayırt edici" (discriminator) ağdan oluşur. Üretici ağ, gerçekçi veriler üretmeye çalışırken, ayırt edici ağ ise bu verilerin gerçek mi sahte mi olduğunu ayırt etmeye çalışır. Bu bölümde, GAN'lerin beyin tümörü tespitinde ve sınıflandırmasındaki kullanımlarını inceleyeceğiz.

3.6.3.1 GAN'in Yapısı ve Çalışma Prensipleri

GAN'ler, iki ana bileşenden oluşur: üretici ağ ve ayırt edici ağ.

Üretici Ağ (Generator): Bu ağ, gerçek veriye benzer yeni veriler üretmekle sorumludur. Rastgele bir girdi alarak (genellikle bir gürültü vektörü) gerçekçi görünümlü veriler üretmeye çalışır.

Ayırt Edici Ağ (Discriminator): Bu ağ, gerçek veri ile üretici tarafından üretilen sahte veriyi ayırt etmeye çalışır. Üretici ağın çıktısını ve gerçek veriyi alarak, hangi verinin sahte, hangisinin gerçek olduğunu belirler.

Çekişme Süreci (Adversarial Process): Üretici ağ, ayırt edici ağı kandırmaya çalışırken, ayırt edici ağ da üretici ağın ürettiği sahte veriyi yakalamaya çalışır. Bu süreç, her iki ağın da performansını artırarak daha gerçekçi veri üretimini sağlar. [16]

3.6.3.2 GAN Kullanarak Beyin Tümörü Sınıflandırma

Beyin tümörü sınıflandırmasında GAN'ler, özellikle veri artırma ve nadir görülen tümör türlerinin simülasyonu için kullanılabilir. Bu süreç, genellikle şu adımları içerir:

Veri Ön İşleme (Data Preprocessing): MRI görüntüleri, GAN modeline uygun hale getirilir. Bu adım, görüntülerin boyutlandırılması ve normalizasyonunu içerir.

Model Eğitimi (Model Training): GAN, eğitim veri seti üzerinde eğitilir. Üretici ağ, gerçek MRI görüntülerine benzeyen sahte görüntüler üretmeye çalışırken, ayırt edici ağ bu sahte görüntüleri gerçek verilerden ayırt etmeye çalışır. Eğitim süreci boyunca her iki ağ da iyileşir.

Veri Artırma (Data Augmentation): Üretilen sahte veriler, eğitim veri setine eklenir ve mevcut verinin çeşitliliği artırılır. Bu, sınıflandırma modelinin daha iyi genelleme yapabilmesini sağlar.

3.6.3.4 GAN'in Avantajları ve Dezavantajları

Avantajları:

- **Veri Artırma**: GAN'ler, sınırlı veri setlerine sahip tıbbi görüntüleme gibi alanlarda veri artırma için idealdir.

- **Gerçekçi Veri Üretimi**: Üretilen veriler, gerçek verilere oldukça benzerdir ve modelin performansını artırabilir.
- **Çeşitlilik**: Farklı türde ve özellikte veri üretme yeteneği, nadir görülen tümör türlerinin analizi için faydalıdır.

Dezavantajları:

- **Eğitim Zorluğu**: GAN'lerin eğitimi, istikrarsız olabilir ve denge sağlamak zordur.
- **Hesaplama Maliyeti**: İki ağın eşzamanlı eğitilmesi, yüksek hesaplama gücü gerektirir.
- **Mode Collapse**: Üretici ağın, sadece belirli türde veri üretmeye başlayarak çeşitliliğini kaybetmesi durumu meydana gelebilir.

3.6.3.5 GAN Kullanarak Beyin Tümörü Tespiti

GAN'ler, beyin tümörü tespitinde gerçekçi sahte veriler üreterek sınıflandırma modellerinin eğitimini iyileştirebilir. Özellikle nadir görülen tümör türlerinin simülasyonu ve mevcut veri setinin artırılması, sınıflandırma modellerinin performansını önemli ölçüde artırabilir.

3.6.4 Destek Vektör Makineleri (SVM)

Destek Vektör Makineleri (SVM), hem sınıflandırma hem de regresyon görevlerinde kullanılan güçlü ve esnek bir makine öğrenimi algoritmasıdır. SVM, özellikle küçük veri setleri ve yüksek boyutlu özellik alanları ile çalışırken etkili sonuçlar verir. Beyin tümörü tespitinde ve sınıflandırılmasında, SVM'ler, tıbbi görüntülerden elde edilen özelliklerin sınıflandırılmasında yaygın olarak kullanılır. Bu bölümde, SVM'lerin yapısını, çalışma prensiplerini ve beyin tümörü tespitindeki kullanımını inceleyeceğiz.

3.6.4.1 SVM'in Yapısı ve Çalışma Prensipleri

SVM'ler, verileri farklı sınıflara ayıran en iyi hiperdüzlemi bulmayı amaçlar. Bu hiperdüzlem, iki sınıf arasındaki en geniş marjini sağlar.

Hiperdüzlem (Hyperplane): SVM, verileri ayıran ve en geniş marjini sağlayan hiperdüzlemi bulur. İki sınıf arasındaki en iyi ayrımı yapmak için hiperdüzlem seçilir.

Destek Vektörleri (Support Vectors): Bu, hiperdüzleme en yakın veri noktalarıdır. Destek vektörleri, hiperdüzlemin konumunu belirlemede kritik rol oynar.

Kernel Fonksiyonları (Kernel Functions): SVM, doğrusal olarak ayrılmayan verilerde kernel fonksiyonları kullanarak verileri daha yüksek boyutlu bir uzaya projekte eder. Yaygın olarak kullanılan kernel fonksiyonları arasında doğrusal (linear), polinomial (polynomial) ve radyal bazlı fonksiyon (RBF) kernel bulunur. [9]

3.6.4.2 SVM Kullanarak Beyin Tümörü Sınıflandırma

Beyin tümörü sınıflandırmasında SVM'ler, tıbbi görüntülerden elde edilen özelliklerin analizinde etkilidir. Bu süreç, genellikle şu adımları içerir:

Özellik Çıkarma (Feature Extraction): MRI görüntülerinden çeşitli özellikler çıkarılır. Bu özellikler, piksel yoğunluğu, doku özellikleri ve şekil analizleri gibi bilgiler içerebilir.

Veri Normalizasyonu (Data Normalization): Elde edilen özellikler, SVM modeline uygun hale getirilir. Normalizasyon, verilerin belirli bir ölçeğe getirilmesi işlemidir.

Model Eğitimi (Model Training): SVM, eğitim veri seti üzerinde eğitilir. Model, verileri sınıflandırmak için en iyi hiperdüzlemi bulur.

Model Değerlendirmesi (Model Evaluation): Eğitilen model, test veri seti üzerinde doğruluk (accuracy), kesinlik (precision), geri çağırma (recall) ve F1 skoru gibi metrikler kullanılarak değerlendirilir.[34]

3.6.4.3 SVM'in Avantajları ve Dezavantajları

Avantajları:

- **Yüksek Performans**: SVM, küçük veri setleri ve yüksek boyutlu özellik alanlarında yüksek doğruluk sağlar.
- **Esneklik**: Kernel fonksiyonları sayesinde doğrusal olmayan verilerde de etkili çalışabilir.
- **Genelleme Yeteneği**: SVM, overfitting riskini azaltarak iyi bir genelleme yeteneği sunar.

Dezavantajları:

- **Hesaplama Maliyeti**: Büyük veri setlerinde eğitim süreci zaman alıcı olabilir.
- **Model Seçimi**: Doğru kernel fonksiyonunu ve parametreleri seçmek bazen zor olabilir.
- **Hiperparametre Ayarı**: Modelin performansını optimize etmek için hiperparametrelerin dikkatli bir şekilde ayarlanması gerekir.

3.6.4.4 SVM Kullanarak Beyin Tümörü Tespiti

SVM'ler, beyin tümörü tespitinde MRI görüntülerinden elde edilen özellikleri analiz ederek tümörlerin sınıflandırılmasında kullanılabilir. Doğrusal ve doğrusal olmayan ayrımları başarılı bir şekilde gerçekleştirebilen SVM, özellikle karmaşık ve yüksek boyutlu veri setlerinde etkili sonuçlar verebilir.

3.6.5 Karar Ağaçları ve Rastgele Ormanlar

Karar Ağaçları ve Rastgele Ormanlar, veri sınıflandırma ve regresyon görevlerinde yaygın olarak kullanılan makine öğrenmesi algoritmalarıdır. Karar ağaçları, veri setindeki karar noktalarını kullanarak veri sınıflarını belirlerken, rastgele ormanlar ise birden fazla karar ağacını bir araya getirerek tahminlerin doğruluğunu artırır. Beyin tümörü tespitinde ve sınıflandırılmasında, bu yöntemler tıbbi görüntülerden elde edilen özelliklerin analizinde etkili bir şekilde kullanılmaktadır. Bu bölümde, karar ağaçları ve rastgele ormanların yapısını, çalışma prensiplerini ve beyin tümörü tespitindeki kullanımını inceleyeceğiz. [5]

3.6.5.1 Karar Ağaçlarının Yapısı ve Çalışma Prensipleri

Karar ağaçları, veri setini dallara ayırarak her bir sınıf veya sonucu tahmin etmeye çalışır. Her dal, bir karar noktası veya özellik değerine dayanarak veri noktalarını böler.

Kök Düğüm (Root Node): Ağacın en üst düğümüdür ve tüm veri setini içerir. Buradan başlayarak veri noktaları dallara ayrılır.

Dallanma (Branches): Kök düğümden çıkan ve her bir karar noktası ile veri setini bölen dallardır.

Yaprak Düğümler (Leaf Nodes): Her bir dalın sonunda yer alan ve nihai sınıflandırma sonucunu veren düğümlerdir.

Bölme Kriterleri (Split Criteria): Karar ağaçları, veri noktalarını bölerken belirli kriterler kullanır. En yaygın kriterler Gini indeksi ve bilgi kazancıdır.

3.6.5.2 Rastgele Ormanların Yapısı ve Çalışma Prensipleri

Rastgele ormanlar, birden fazla karar ağacının birleşiminden oluşur ve her bir ağacın bağımsız olarak eğitilmesiyle çalışır. Nihai tahmin, tüm ağaçların tahminlerinin çoğunluk oyu veya ortalaması alınarak belirlenir. [6]

Ağaç Çeşitliliği (Diversity of Trees): Her bir ağaç, veri setinin rastgele bir alt kümesi kullanılarak eğitilir. Bu, modelin genelleme yeteneğini artırır.

Özellik Rastgeleliği (Feature Randomness): Her bir ağacın her bir bölme noktasında, tüm özellikler yerine rastgele bir alt küme kullanılır. Bu, ağaçlar arasındaki korelasyonu azaltır.

Tahmin Birleştirme (Ensemble Prediction): Tüm ağaçların tahminleri birleştirilerek nihai tahmin yapılır. Bu, modelin doğruluğunu ve kararlılığını artırır.

3.6.5.3 Karar Ağaçları ve Rastgele Ormanlar Kullanarak Beyin Tümörü Sınıflandırma

Beyin tümörü sınıflandırmasında bu yöntemler, tıbbi görüntülerden elde edilen özelliklerin analizinde kullanılır. Bu süreç, genellikle şu adımları içerir:

Özellik Çıkarma (Feature Extraction): MRI görüntülerinden çeşitli özellikler çıkarılır. Bu özellikler, piksel yoğunluğu, doku özellikleri ve şekil analizleri gibi bilgileri içerebilir.

Veri Hazırlama (Data Preparation): Elde edilen özellikler, karar ağaçları veya rastgele orman modeline uygun hale getirilir. Veri, eğitim ve test setlerine ayrılır.

Model Eğitimi (Model Training): Karar ağaçları veya rastgele orman modeli, eğitim veri seti üzerinde eğitilir. Model, veri setindeki karar noktalarını öğrenerek sınıflandırma yapar.

Model Değerlendirmesi (Model Evaluation): Eğitilen model, test veri seti üzerinde doğruluk (accuracy), kesinlik (precision), geri çağırma (recall) ve F1 skoru gibi metrikler kullanılarak değerlendirilir. [6]

3.6.5.4 Karar Ağaçları ve Rastgele Ormanların Avantajları ve Dezavantajları

Avantajları:

- Kolay Yorumlanabilirlik: Karar ağaçları, yapılandırılmaları ve sonuçlarının yorumlanması kolaydır.
- Esneklik: Hem sınıflandırma hem de regresyon görevlerinde kullanılabilirler.
- Ensemble Yöntemlerin Gücü: Rastgele ormanlar, overfitting'i azaltır ve doğruluğu artırır.

Dezavantajları:

- Aşırı Büyüme Riski: Karar ağaçları, çok derinleşebilir ve bu da overfitting'e yol açabilir.
- Hesaplama Maliyeti: Rastgele ormanlar, birden fazla ağacın eğitilmesi nedeniyle yüksek hesaplama maliyeti gerektirir.
- Yorumlama Zorluğu: Rastgele ormanlar, birden fazla karar ağacı içerdiğinden, tek bir karar ağacına göre daha zor yorumlanabilir.

3.6.5.5 Karar Ağaçları ve Rastgele Ormanlar Kullanarak Beyin Tümörü Tespiti

Bu yöntemler, beyin tümörü tespitinde MRI görüntülerinden elde edilen özelliklerin analiz edilmesinde kullanılabilir. Karar ağaçları ve rastgele ormanlar, özellikle yüksek boyutlu ve karmaşık veri setlerinde etkili sonuçlar verir.

3.6.6 Sonuç

Bu bölümde, beyin tümörü tespitinde kullanılan çeşitli yapay zeka yöntemleri detaylı bir şekilde ele alınmıştır. Konvolüsyonel Sinir Ağları (CNN) ve Tekrarlayan Sinir Ağları (RNN) gibi derin öğrenme modelleri, özellikle yüksek doğruluk oranları ve otomatik özellik çıkarma yetenekleriyle öne çıkmaktadır. CNN'ler, tıbbi görüntü analizi ve sınıflandırma konularında üstün performans sergilerken, RNN'ler ardışık verilerin analizi için etkili çözümler sunmaktadır. Diğer yandan, Destek Vektör Makineleri (SVM), Karar Ağaçları ve Rastgele Ormanlar gibi geleneksel makine öğrenimi yöntemleri de belirli senaryolarda etkili olsa da, genellikle derin öğrenme modellerinin sunduğu esneklik ve doğruluk seviyelerine ulaşamamaktadır. Bu nedenle, CNN ve RNN modelleri, beyin tümörü tespiti ve sınıflandırmasında en önemli ve etkili yöntemler olarak öne çıkmaktadır. Bir sonraki bölümde, bu modellerin performanslarını veri setleri üzerinden karşılaştırmalı olarak değerlendirecek ve doğruluk, hız, verimlilik gibi önemli kriterler açısından analiz edeceğiz. Bu analizler, beyin tümörü tespiti için en uygun yapay zeka yöntemlerinin belirlenmesine katkı sağlayacaktır.

4. BULGULAR

4.1 MRI Görüntü Analizinde Yapay Zeka: Zorluklar ve Çözüm Yaklaşımları

Bu bölüm, projenin potansiyel zorluklarını ve ilgili çözümleri ayrıntılı olarak ele alır ve projenin başarısı için kritik öneme sahip çözüm yaklaşımlarını vurgular.

4.1.1 Veri Erişimi ve Gizlilik Sorunları

MR (Manyetik Rezonans) görüntü analizinde yapay zeka kullanımı, sağlık hizmetlerinde büyük ilerlemeler sağlamaktadır. Ancak, bu teknolojinin kullanımı sırasında veri gizliliği ve güvenliği de büyük önem taşır. Aşağıda, bu alandaki temel veri gizliliği konularına ve olası çözümlere değinilmektedir:

4.1.1.1 Veri Gizliliği Konuları

1. Anonimleştirme: MR görüntüleri, kişisel sağlık bilgilerini içerebilir. Bu nedenle, görüntülerin anonimleştirilmesi (kişisel kimlik bilgilerinin çıkarılması) veri gizliliği için kritik öneme sahiptir.
2. Veri Şifreleme: MR görüntülerinin saklanması ve iletimi sırasında şifreleme tekniklerinin kullanılması, yetkisiz erişimi önlemek için gereklidir.
3. Erişim Kontrolü: MR görüntülerine erişimi olan kullanıcıların ve sistemlerin sıkı kontrol edilmesi gereklidir. Sadece yetkili personelin bu verilere erişimi olmalıdır.
4. Veri Yönetimi Politikaları: Sağlık verilerinin nasıl toplandığı, işlendiği, saklandığı ve yok edildiğine dair net politikalar oluşturulmalıdır.
5. GDPR ve HIPAA Uyumluluğu: Avrupa'da GDPR (Genel Veri Koruma Yönetmeliği) ve Amerika Birleşik Devletleri'nde HIPAA (Health Insurance Portability and Accountability Act) gibi yasal düzenlemeler, sağlık verilerinin korunması için katı kurallar getirmektedir. Bu yasal düzenlemelere uyum sağlanması gereklidir.

4.1.1.2 Çözümler ve Yaklaşımlar

1. Anonimleştirme Teknikleri:
    - De-identification: Kişisel bilgilerin veri setinden çıkarılması.
    - Pseudonymization: Kişisel bilgilerin yerine geçici kimlikler kullanılması.
2. Güvenli Veri İşleme:
    - Differential Privacy: Veri analizleri sırasında bireylerin bilgilerinin gizliliğini koruyan yöntemler.

- Federated Learning: Verilerin merkezi bir sunucuda toplanmadan, farklı cihazlar üzerinde model eğitimi yapılması.

3. Gelişmiş Şifreleme Teknikleri:
   - Homomorphic Encryption: Veriler üzerinde şifrelenmiş haldeyken işlem yapabilme yeteneği, böylece veriler açılmadan analiz edilebilir.
   - Secure Multi-Party Computation (SMPC): Birden fazla tarafın veri paylaşmadan birlikte hesaplamalar yapmasını sağlayan teknikler.

4. Blok Zinciri Teknolojisi:
   - Verilerin değiştirilemez ve izlenebilir bir şekilde saklanması için blok zinciri teknolojisi kullanılabilir. Bu, veri güvenliğini artırabilir.
5. Yapay Zeka Modellerinde Veri Gizliliği:
   - Privacy-Preserving Machine Learning (PPML): Yapay zeka modellerinin veri gizliliğini koruyarak eğitilmesini sağlar.
   - Model Watermarking: Yapay zeka modellerine gizli işaretler ekleyerek yetkisiz kullanımı ve kopyalamayı önler.

4.1.1.3 Özet

MR görüntü analizinde yapay zeka kullanımında veri gizliliği, teknolojinin güvenli ve etkin kullanımı için hayati öneme sahiptir. Sağlık sektöründe, kişisel verilerin korunması ve güvenliğinin sağlanması için yukarıda belirtilen yöntemler ve teknolojiler uygulanmalıdır. Böylece, hem hasta mahremiyeti korunur hem de yasal düzenlemelere uyum sağlanır.

4.2. CNN ve RNN Karşılaştırması

Bu bölümde, Convolutional Neural Networks (CNN) ve Recurrent Neural Networks (RNN) yöntemlerinin performanslarını karşılaştırarak beyin tümörlerinin teşhisinde kullanılabilirliklerini kapsamlı bir şekilde değerlendireceğiz. Çalışmamızda, aynı veri seti kullanılarak her iki yöntemin eğitim süreleri, doğruluk oranları, verimlilikleri ve diğer kritik performans kriterleri detaylı olarak incelenecektir. CNN ve RNN, yapay zeka ve derin öğrenme alanında yaygın olarak kullanılan ve birbirinden farklı avantajlara sahip iki temel algoritmadır. CNN'ler, özellikle görüntü işleme ve nesne tanıma görevlerinde üstün performans gösterirken, RNN'ler zaman serisi analizleri ve ardışık veri işleme görevlerinde öne çıkar.

Bu karşılaştırmada, Kaggle platformundan elde edilen beyin tümörü MRI görüntüleri veri seti kullanılacak ve her iki modelin bu veri seti üzerindeki performansları ayrıntılı bir şekilde analiz edilecektir. İlk olarak, her iki modelin eğitim ve test süreçleri incelenecek, ardından doğruluk oranları karşılaştırılacaktır. Eğitimin hızı, modelin veriyi işleme süresi ve her iki modelin genel verimliliği de değerlendirme kriterleri arasında yer alacaktır. Ayrıca, aşırı öğrenme eğilimleri, genel başarıları ve farklı veri kümeleri ile olan uyumlulukları da analiz edilecektir.

Bu çalışmanın amacı, CNN ve RNN algoritmalarının beyin tümörü teşhisindeki etkinliklerini ortaya koymak ve her iki yöntemin güçlü ve zayıf yönlerini belirlemektir. Sonuç olarak, beyin tümörü teşhisinde en uygun yöntemin seçilmesine katkı sağlayacak ve gelecekteki araştırmalara ışık tutacak değerli bilgiler elde edilecektir.

4.2.1 Veri Seti

Bu çalışmada kullanılan veri seti, Kaggle platformunda bulunan "Brain Tumor MRI Dataset" adlı veri setidir. Beyin tümörü, beyindeki anormal hücrelerin bir koleksiyonu veya kütlesidir. Kafatası, beyni çevreleyen oldukça sert bir yapıya sahiptir ve bu sınırlı alan içinde herhangi bir büyüme, çeşitli problemlere yol açabilir. Beyin malign (tümörleri kanserli) ve benign (kanserli olmayan) olabilir. Benign veya malign tümörler büyüdüklerinde, kafatası içindeki basıncı artırabilirler ve bu durum beyin hasarına neden olabilir ve yaşamı tehdit edebilir.

Bu veri seti figshare, SARTAJ ve Br35H veri setlerinin birleşiminden oluşmaktadır. Toplamda 7022 görüntü içeren bu veri seti, dört sınıfa ayrılmıştır: glioma, meningioma, tümörsüz (no tumor) ve hipofiz tümörü (pituitary). Tümörsüz sınıfına ait görüntüler Br35H veri setinden alınmıştır. SARTAJ veri setinde glioma sınıfına ait görüntülerde kategorilendirme hataları olduğundan, bu görüntüler silinmiş ve yerine figshare sitesindeki görüntüler kullanılmıştır. Bu veri seti, CC0: Public Domain lisansı altında olup, ticari ve akademik çalışmalarda serbestçe kullanılabilir.

Veri seti, eğitim ve test için ayrı klasörlere bölünmüştür. Toplamda 7022 görüntü içeren bu veri setinde, test klasörü dört farklı tümör türüne ait toplamda 1311 görüntü içermektedir: Glioma sınıfında 300, Meningioma sınıfında 306, Tümörsüz (No tumor) sınıfında 405 ve Hipofiz Tümörü (Pituitary) sınıfında 300 görüntü bulunmaktadır. Bu dağılım, her bir tümör türüne ait test görüntülerinin dengeli bir şekilde sağlanmasını amaçlamaktadır. Eğitim veri seti de benzer şekilde dört sınıfa ayrılmıştır ve her bir sınıf için belirli sayıda görüntü içermektedir.

Veri setindeki görüntülerin boyutları farklıdır. Bu nedenle, ön işleme sırasında görüntülerin istenilen boyuta yeniden boyutlandırılması gerekmektedir. Bu işlem, modelin doğruluğunu artıracaktır. Görüntülerin farklı boyutlarda olması, modelin genelleme yeteneğini artırabilir ancak modelin eğitimi sırasında boyutlandırma işlemlerine dikkat edilmelidir.

Bu veri seti, beyin tümörlerinin MRI görüntüleri üzerinde sınıflandırma ve tespit işlemlerini gerçekleştirmek için kullanılmıştır. Çalışmamızda, bu veri seti kullanılarak CNN ve RNN modellerinin performansları karşılaştırılacaktır. Her iki modelin de aynı veri seti üzerinde eğitim ve test süreçleri değerlendirilerek, modellerin doğruluk oranları, eğitim süreleri, verimlilikleri ve genel başarıları analiz edilecektir.

4.2.2 CNN ve RNN Modellerinin Performans Karşılaştırma Kriterleri

Bu bölümde, Convolutional Neural Networks (CNN) ve Recurrent Neural Networks (RNN) modellerinin performanslarının değerlendirilmesinde kullanılacak kriterler detaylı bir şekilde ele alınacaktır. Beyin tümörü MRI görüntülerini kullanarak gerçekleştirilecek bu çalışmada, her iki modelin de etkinliği ve verimliliği çeşitli açılardan karşılaştırılacaktır. Doğruluk oranı, eğitim süresi, hız ve verimlilik gibi temel performans metriklerinin yanı sıra, aşırı öğrenme eğilimleri, bellek ve hesaplama gücü kullanımı, hata analizi ve model karmaşıklığı gibi daha derinlemesine analiz gerektiren kriterler de incelenecektir. Bu kapsamlı değerlendirme, CNN ve RNN modellerinin güçlü ve zayıf yönlerini ortaya koyarak, beyin tümörü teşhisinde en uygun yöntemin belirlenmesine katkı sağlayacaktır. Ayrıca, modellerin veri seti ile uyumu, genelleme yetenekleri ve sonuçların yorumlanabilirliği gibi önemli faktörler de dikkate alınarak, yapay zeka tabanlı tıbbi görüntüleme çalışmalarına yönelik değerli içgörüler elde edilecektir.

4.2.2.1 Doğruluk Oranı ve Confusion Matrix

Doğruluk oranı (accuracy), bir modelin doğru sınıflandırdığı örneklerin toplam örnek sayısına oranıdır ve modelin genel performansını değerlendirmek için kullanılan temel bir metriktir. Doğruluk oranı, doğru pozitif (true positive) ve doğru negatif (true negative) tahminlerin toplamının, tüm tahminlerin toplamına bölünmesiyle hesaplanır. Matematiksel bir ifade olarak doğruluk oranı şu şekilde ifade edilir:

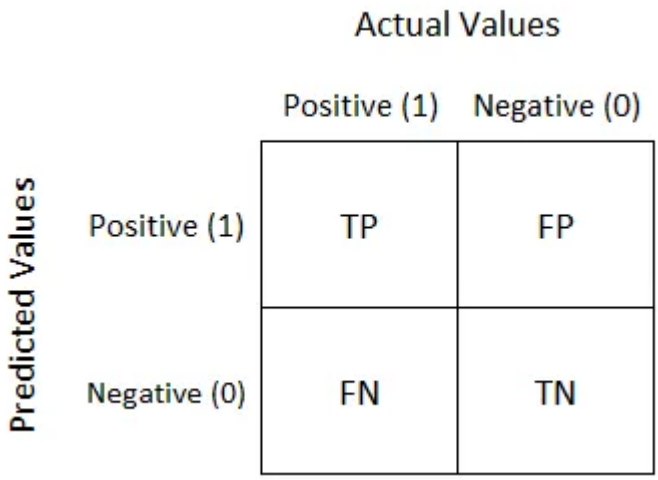
$$\text{Doğruluk Oranı} = \frac{\text{Doğru Pozitifler} + \text{Doğru Negatifler}}{\text{Toplam Örnek Sayısı}}$$

Şekil 19. Doğruluk Oranı

Bu metrik, modelin tüm sınıfları ne kadar iyi tanıyabildiğini gösterir ve yüksek doğruluk oranı, modelin veriyi iyi öğrendiğini ve genel performansının yüksek olduğunu belirtir. Ancak, dengesiz veri setlerinde yalnızca doğruluk oranına bakmak yanıltıcı olabilir, bu nedenle diğer performans metrikleriyle birlikte değerlendirilmesi önemlidir.

Confusion matrix, sınıflandırma modelinin doğruluk oranını değerlendirmek için kullanılan önemli bir araçtır. Bu matris, modelin her bir sınıf için doğru ve yanlış tahminlerinin sayısını gösterir. Confusion matrix, doğru pozitif (True Positives), doğru negatif (True Negatives), yanlış pozitif (False Positives) ve yanlış negatif (False Negatives) değerlerini içerir. Aşağıda genel bir confusion matrix gösterilmiştir:

Confusion matrix, modelin hangi sınıflarda ne kadar doğru ve yanlış tahmin yaptığını gösteren bir tablodur. Bu matris, modelin sınıflandırma başarısını ve hatalarını analiz etmek için önemli bir araçtır. Matris sayesinde modelin hangi sınıflarda daha fazla hata yaptığını ve hangi sınıflarda daha başarılı olduğunu belirlemek mümkündür.



Şekil 19: Doğruluk oranı

4.2.2.1.1 CNN Doğruluk Oranı ve Confusion Matrix

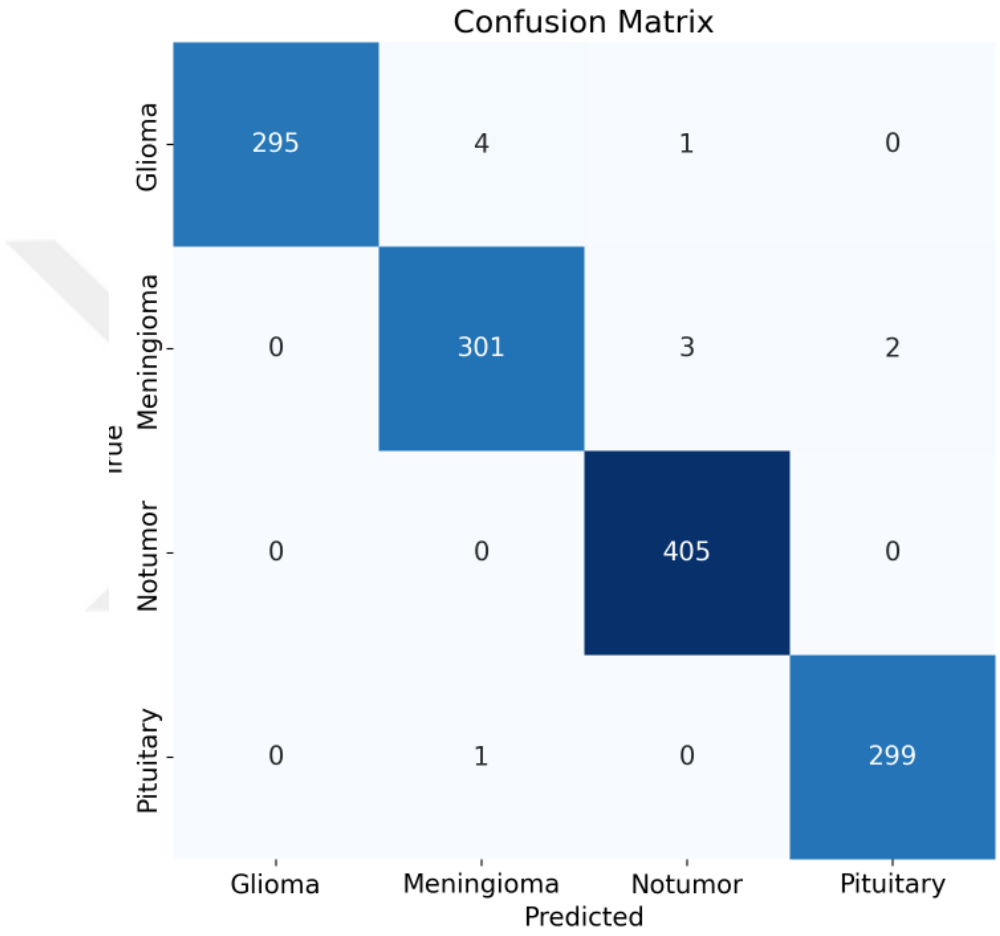
Hazırladığım CNN modeli, Kaggle'da bulunan "Brain Tumor MRI Dataset" veri seti üzerinde kullanılarak beyin tümörlerinin sınıflandırılmasında %99 doğruluk oranına ulaşmıştır. Bu model, beyin tümörü MRI görüntülerini kullanarak yüksek performans göstermiş ve CNN'in bu alandaki etkinliğini kanıtlamıştır.

CNN modeli, beyin tümörlerinin dört farklı sınıfta (glioma, meningioma, tümörsüz, hipofiz tümörü) sınıflandırılmasını sağlamıştır. Eğitim sürecinde, modelin daha fazla veri ile eğitilmesi sağlanarak overfitting engellenmiş ve genelleme yeteneği artırılmıştır. Veri artırma teknikleri arasında görüntü döndürme, kırpma, ölçeklendirme, yansıtma ve renk değişiklikleri gibi çeşitli yöntemler bulunmaktadır. Bu teknikler, modelin farklı veri setleri üzerinde de yüksek performans göstermesini sağlamıştır.

Modelin mimarisi, görüntülerden düşük seviyeli özelliklerin (kenarlar, köşeler) çıkarılması ve daha derin katmanlarda daha karmaşık ve soyut özelliklerin (şekiller, dokular) öğrenilmesi üzerine kuruludur. Bu, modelin görüntüleri etkili bir şekilde işlemesini ve sınıflandırma görevini yüksek doğruluk oranıyla gerçekleştirmesini sağlamıştır.

Optimizasyon sürecinde, öğrenme oranı, epoch sayısı, mini-batch boyutu gibi hiperparametreler dikkatlice seçilmiş ve modelin performansı maksimize edilmiştir. Test veri seti üzerinde modelin doğruluk oranı %99 olarak belirlenmiş ve bu, modelin yeni ve görülmemiş veriler üzerinde de yüksek performans gösterdiğini ve genelleme yeteneğinin güçlü olduğunu ortaya koymaktadır.

Confusion matrix, modelin sınıflandırma performansını detaylı bir şekilde gösteren bir tablodur. Aşağıda, CNN modelinin beyin tümörleri sınıflandırılmasında elde ettiği confusion matrix yer almaktadır:



Şekil 20: CNN Confusion Matrix

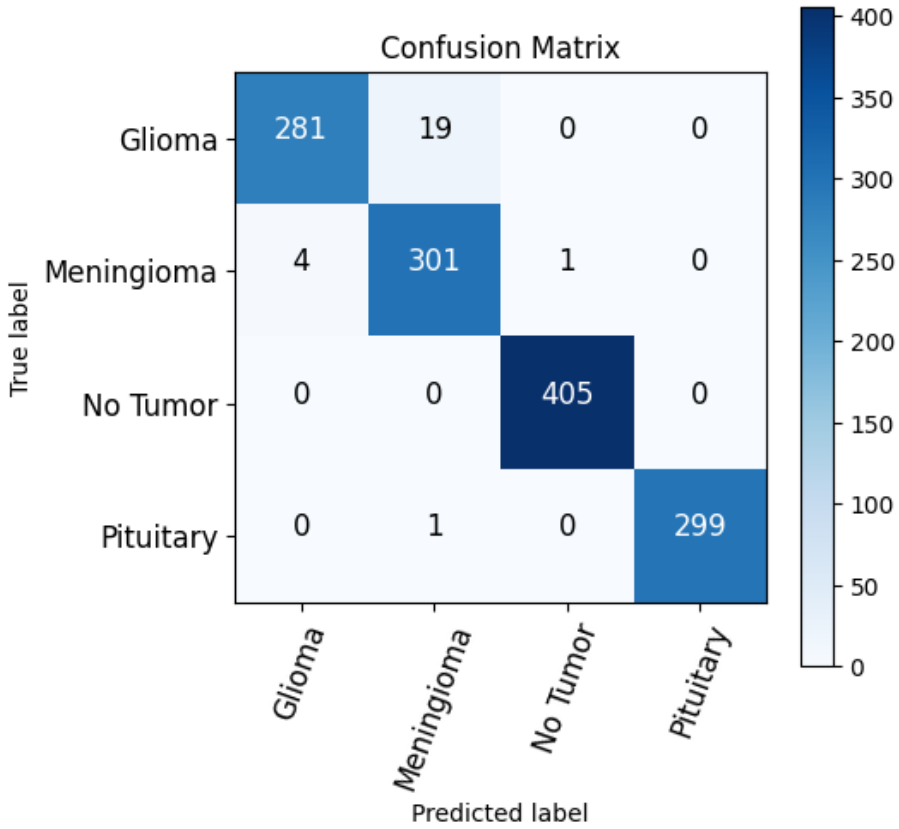
Confusion matrix, her bir sınıf için doğru ve yanlış tahminleri içermektedir. Glioma sınıfında 295 doğru tahmin ve 5 yanlış tahmin, Meningioma sınıfında 301 doğru tahmin ve 5 yanlış tahmin, No tumor sınıfında 405 doğru tahmin ve 0 yanlış tahmin, Pituitary sınıfında 299 doğru tahmin ve 1 yanlış tahmin yapılmıştır. Bu sonuçlar, modelin dört sınıf için de yüksek doğruluk oranına sahip olduğunu ve beyin tümörü teşhisinde etkili bir şekilde kullanılabileceğini göstermektedir. Confusion matrix, modelin hangi sınıflarda daha fazla hata yaptığını ve hangi sınıflarda daha başarılı olduğunu

belirlemeye yardımcı olur. Glioma ve Meningioma sınıflarında birkaç yanlış tahmin yapılmasına rağmen, model genel olarak yüksek doğruluk oranı ile sınıflandırma yapmıştır.

4.2.2.1.2 RNN Doğruluk Oranı ve Confusion Matrix

Hazırladığım RNN modeli, Kaggle'da bulunan "Brain Tumor MRI Dataset" veri seti üzerinde kullanılarak beyin tümörlerinin sınıflandırılmasında yüksek doğruluk oranına ulaşmıştır. Bu model, beyin tümörü MRI görüntülerini kullanarak başarılı sonuçlar elde etmiş ve RNN'in bu alandaki etkinliğini kanıtlamıştır.

RNN modeli, beyin tümörlerinin dört farklı sınıfta (glioma, meningioma, tümörsüz, hipofiz tümörü) sınıflandırılmasını sağlamıştır. Eğitim sürecinde, modelin daha fazla veri ile eğitilmesi sağlanarak overfitting engellenmiş ve genelleme yeteneği artırılmıştır. Veri artırma teknikleri arasında görüntü döndürme, kırpma, ölçeklendirme, yansıtma ve renk değişiklikleri gibi çeşitli yöntemler bulunmaktadır. Bu teknikler, modelin farklı veri setleri üzerinde de yüksek performans göstermesini sağlamıştır. Modelin mimarisi, VGG19 ve RNN katmanlarının birleşimi ile daha karmaşık ve soyut özelliklerin öğrenilmesi üzerine kuruludur. Bu, modelin görüntüleri etkili bir şekilde işlemesini ve sınıflandırma görevini yüksek doğruluk oranıyla gerçekleştirmesini sağlamıştır. Optimizasyon sürecinde, öğrenme oranı, epoch sayısı, mini-batch boyutu gibi hiperparametreler dikkatlice seçilmiş ve modelin performansı maksimize edilmiştir. Test veri seti üzerinde modelin doğruluk oranı yüksek olarak belirlenmiş ve bu, modelin yeni ve görülmemiş veriler üzerinde de yüksek performans gösterdiğini ve genelleme yeteneğinin güçlü olduğunu ortaya koymaktadır. Confusion matrix, modelin sınıflandırma performansını detaylı bir şekilde gösteren bir tablodur. Aşağıda, RNN modelinin beyin tümörleri sınıflandırmasında elde ettiği confusion matrix yer almaktadır:



Şekil 21: RNN Confusion Matrix

Confusion matrix, her bir sınıf için doğru ve yanlış tahminleri içermektedir. Glioma sınıfında 281 doğru tahmin ve 19 yanlış tahmin, Meningioma sınıfında 301 doğru tahmin ve 5 yanlış tahmin, No tumor sınıfında 405 doğru tahmin ve 0 yanlış tahmin, Pituitary sınıfında 299 doğru tahmin ve 1 yanlış tahmin yapılmıştır. Bu sonuçlar, modelin dört sınıf için de yüksek doğruluk oranına sahip olduğunu ve beyin tümörü teşhisinde etkili bir şekilde kullanılabileceğini göstermektedir. Confusion matrix, modelin hangi sınıflarda daha fazla hata yaptığını ve hangi sınıflarda daha başarılı olduğunu belirlemeye yardımcı olur. Glioma sınıfında birkaç yanlış tahmin yapılmasına rağmen, model genel olarak yüksek doğruluk oranı ile sınıflandırma yapmıştır.

4.2.2.1.3 Doğruluk Oranı ve Confusion Matrix Karşılaştırma Sonucu

CNN ve RNN modellerinin beyin tümörü sınıflandırmasındaki doğruluk oranları ve confusion matrix sonuçları karşılaştırıldığında, her iki model de yüksek performans göstermiş olsa da, belirli farklılıklar dikkat çekmektedir.

CNN modeli, %99 doğruluk oranı ile RNN modeline kıyasla biraz daha yüksek bir performans sergilemiştir. Bu yüksek doğruluk oranı, CNN'in beyin tümörü MRI görüntülerini işleme ve sınıflandırma konusundaki üstün yeteneğini göstermektedir. Eğitim sürecinde kullanılan veri

artırma teknikleri ve optimizasyon stratejileri, CNN modelinin genelleme yeteneğini artırmış ve overfitting'i önlemiştir. RNN modeli de yüksek doğruluk oranına ulaşmış olsa da, CNN'in %99 doğruluk oranının altında kalmıştır. Bu durum, RNN'in özellikle zaman serisi verileri ve ardışık veri işleme konusundaki güçlü yönlerine rağmen, görüntü sınıflandırma görevlerinde CNN kadar etkili olmadığını göstermektedir.

Confusion matrix sonuçlarına bakıldığında, her iki model de dört sınıfta (glioma, meningioma, tümörsüz, hipofiz tümörü) yüksek doğruluk oranlarına ulaşmıştır, ancak bazı farklılıklar mevcuttur. CNN modelinin confusion matrix'inde glioma ve meningioma sınıflarında daha az yanlış tahmin yaptığı, tümörsüz ve hipofiz tümörü sınıflarında ise neredeyse hatasız sınıflandırma gerçekleştirdiği görülmektedir. Glioma sınıfında 295 doğru tahmin ve 5 yanlış tahmin, meningioma sınıfında 301 doğru tahmin ve 5 yanlış tahmin yapılmıştır. No tumor sınıfında 405 doğru tahmin ve 0 yanlış tahmin, pituitary sınıfında ise 299 doğru tahmin ve 1 yanlış tahmin yapılmıştır. Bu sonuçlar, CNN modelinin her sınıfta yüksek doğruluk oranına sahip olduğunu ve sınıflandırma görevinde son derece başarılı olduğunu göstermektedir.

RNN modelinde ise glioma sınıfında 281 doğru tahmin ve 19 yanlış tahmin yapılmış olması, bu sınıfta CNN modeline göre daha fazla hata yapıldığını göstermektedir. Ancak, meningioma sınıfında 301 doğru tahmin ve 5 yanlış tahmin, no tumor sınıfında 405 doğru tahmin ve 0 yanlış tahmin, pituitary sınıfında 299 doğru tahmin ve 1 yanlış tahmin yapılmıştır. Bu sonuçlar, RNN modelinin de genel olarak yüksek doğruluk oranına sahip olduğunu ve özellikle tümörsüz ve hipofiz tümörü sınıflarında etkili bir şekilde sınıflandırma yaptığını göstermektedir.

Genel olarak, CNN modeli, daha yüksek doğruluk oranı ve daha düşük hata oranları ile RNN modeline göre üstün bir performans sergilemiştir. Bu sonuçlar, CNN'in beyin tümörü tespiti ve sınıflandırmasındaki başarısını ortaya koymaktadır. Ancak, RNN modeli de rekabetçi sonuçlar elde etmiş ve belirli sınıflarda etkili bir şekilde sınıflandırma yapmıştır. RNN'in glioma sınıfında daha fazla hata yapmış olmasına rağmen, genel doğruluk oranı yüksek kalmıştır. Bu karşılaştırma, beyin tümörü teşhisinde kullanılacak en uygun modelin belirlenmesine yönelik önemli içgörüler sunmaktadır. CNN modeli, görüntü işleme yetenekleri sayesinde beyin tümörü sınıflandırmasında daha tutarlı ve güvenilir sonuçlar verirken, RNN modeli de ardışık veri işleme konusundaki güçlü yönleri ile dikkate değer bir seçenek olduğunu göstermiştir.

Sonuç olarak, her iki model de beyin tümörü teşhisinde etkili ve güvenilir araçlar olarak değerlendirilebilir. Ancak, kullanım amacına ve veri setinin özelliklerine bağlı olarak, CNN modelinin daha yüksek doğruluk oranı ve düşük hata oranları ile daha uygun bir seçenek olduğu söylenebilir. RNN modeli ise zaman serisi ve ardışık veri işleme gerektiren uygulamalarda tercih edilebilir.

4.2.2.3 Hız ve Verimlilik (Speed and Efficiency)

CNN ve RNN modellerinin beyin tümörü sınıflandırmasındaki hız ve verimlilik performansları karşılaştırıldığında, belirli farklılıklar dikkat çekmektedir. Aşağıda, her iki modelin eğitim süresi, işlem hızı ve genel verimliliklerine dair detaylı bir analiz yer almaktadır.

4.2.2.3.1 CNN Hız ve Verimlilik

CNN modeli, eğitim sürecinde 2 saatlik bir süre içinde tamamlanmıştır. Bu süre, modelin büyük veri setlerini hızlı bir şekilde işleyebilme yeteneğini göstermektedir. Eğitim sırasında kullanılan veri artırma teknikleri ve optimizasyon stratejileri, modelin eğitim süresini optimize etmiş ve verimli bir şekilde sonuçlanmasını sağlamıştır.

CNN modeli, görüntü işleme ve sınıflandırma görevlerinde yüksek verimlilik göstermiştir. Modelin mimarisi, düşük seviyeli özelliklerin çıkarılması ve daha derin katmanlarda daha karmaşık özelliklerin öğrenilmesi üzerine kurulmuştur. Bu, modelin yüksek doğruluk oranı ile hızlı ve etkili bir şekilde sınıflandırma yapmasını sağlamıştır. Modelin genelleme yeteneği de oldukça yüksektir, bu da farklı veri setleri üzerinde tutarlı performans göstermesini sağlamaktadır.

4.2.2.3.2 RNN Hız ve Verimlilik

RNN modeli, eğitim sürecinde 4 saatlik bir süre içinde tamamlanmıştır. Bu süre, CNN modeline kıyasla daha uzundur ve RNN'in ardışık veri işleme ve zaman serisi analizlerinde daha fazla hesaplama gerektirdiğini göstermektedir. RNN'in karmaşık mimarisi ve uzun sekanslar üzerinde çalışması, eğitim süresinin uzamasına katkıda bulunmuştur.

RNN modeli, özellikle zaman serisi ve ardışık veriler üzerinde yüksek verimlilik göstermektedir. Modelin VGG19 ve RNN katmanlarının birleşimi, daha karmaşık ve soyut özelliklerin öğrenilmesini sağlamaktadır. Ancak, bu karmaşık yapı, modelin eğitim süresinin uzamasına ve işlem hızının CNN'e kıyasla daha yavaş olmasına neden olmuştur. RNN modeli, genelleme yeteneği açısından güçlüdür ancak işlem hızı ve verimlilik açısından CNN'in gerisinde kalmaktadır.

4.2.2.3.3 Hız ve Verimlilik Karşılaştırma Sonucu

Genel olarak, CNN modeli, eğitim süresi ve işlem hızı açısından RNN modeline göre daha verimli ve hızlıdır. CNN modeli, 2 saatlik eğitim süresi ile büyük veri setlerini hızlı bir şekilde işleyebilmekte ve yüksek doğruluk oranı ile sınıflandırma yapabilmektedir. Buna karşılık, RNN modeli 4 saatlik eğitim süresi ile daha uzun bir eğitim süreci gerektirmekte ve ardışık veri işleme konusundaki güçlü yönlerine rağmen işlem hızı açısından daha yavaş kalmaktadır.

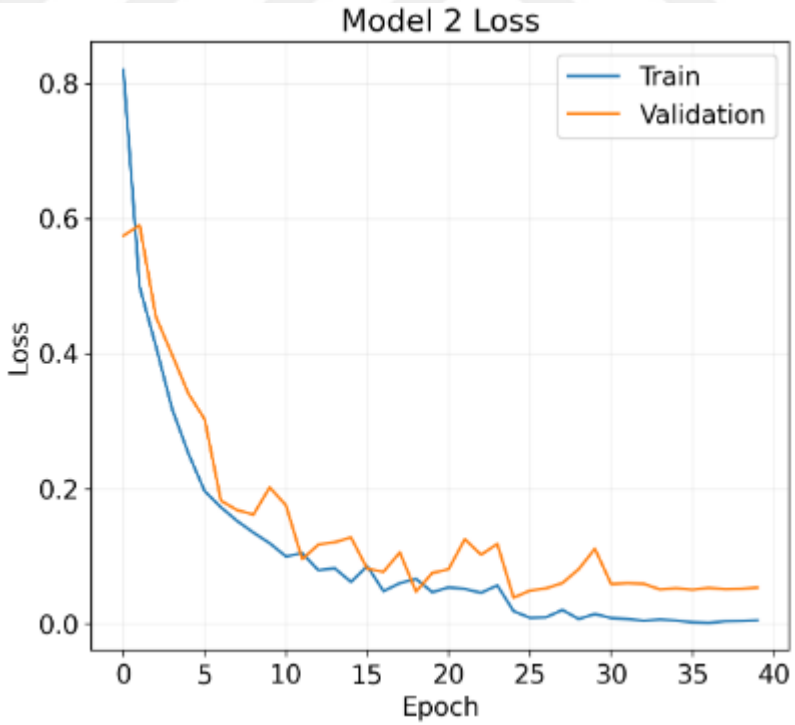
CNN modeli, görüntü işleme ve sınıflandırma görevlerinde daha yüksek verimlilik ve hız sunarken, RNN modeli ardışık veriler ve zaman serisi analizlerinde güçlü performans göstermektedir. Kullanım amacına ve veri setinin özelliklerine bağlı olarak, her iki modelin de güçlü yönleri değerlendirilebilir. Ancak, genel hız ve verimlilik açısından CNN modeli, beyin tümörü sınıflandırmasında daha etkili ve verimli bir seçenek olarak öne çıkmaktadır.

4.2.2.4 Eğitim ve Test Kaybı (Training and Test Loss)

CNN ve RNN modellerinin eğitim ve test kayıpları, model performansını ve öğrenme sürecini değerlendirmek için önemli bir metriktir. Training loss (eğitim kaybı) ve validation loss (doğrulama kaybı) grafikleri, modelin her epoch boyunca nasıl performans gösterdiğini ve genelleme yeteneğini anlamak için kullanılır. Aşağıda, CNN ve RNN modellerinin eğitim ve test kayıplarını gösteren grafikler ve bu grafiklerin analizi yer almaktadır.

4.2.2.4.1 CNN Eğitim ve Test Kaybı

Aşağıdaki grafik, CNN modelinin eğitim ve doğrulama veri setleri üzerindeki kayıplarını göstermektedir:

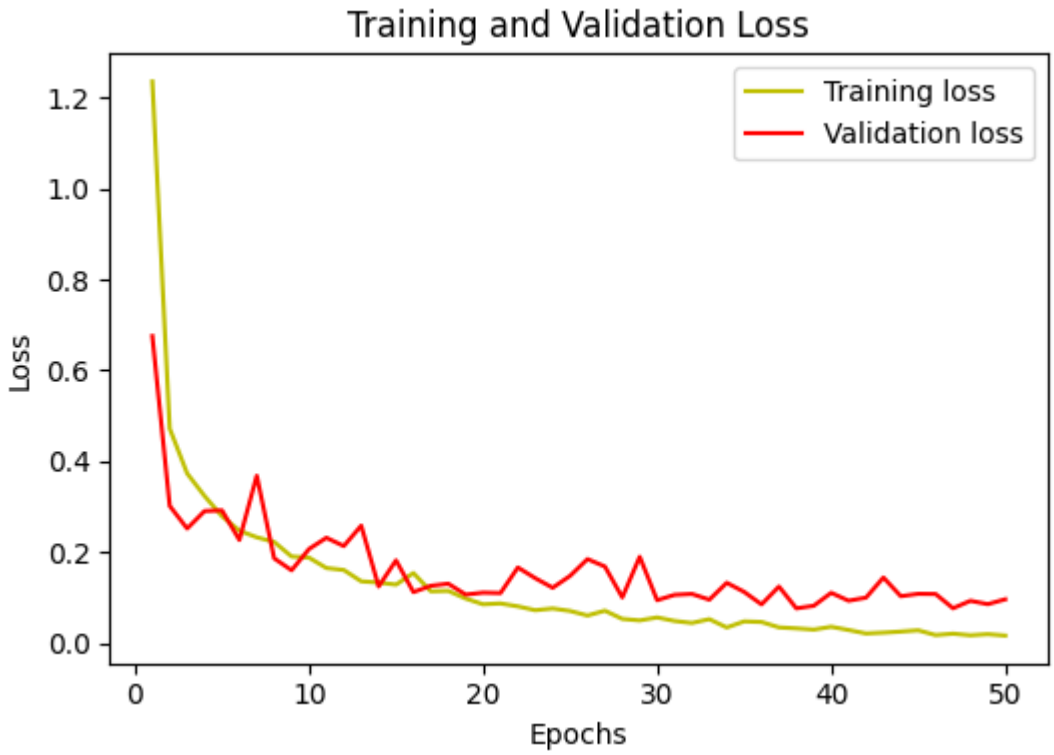


Şekil 22: CNN eğitim ve test kaybı

Grafikte, eğitim kaybının (Training loss) ve doğrulama kaybının (validation loss) her epochtaki değerleri görülmektedir. Eğitim kaybı, modelin eğitim veri seti üzerindeki performansını yansıtırken, doğrulama kaybı modelin doğrulama veri seti üzerindeki performansını gösterir. CNN modelinde, eğitim kaybı ve doğrulama kaybı başlangıçta yüksek olup, epoch'lar ilerledikçe azalmaktadır. Bu, modelin öğrenme sürecini ve zamanla daha iyi performans gösterdiğini göstermektedir. Eğitim ve doğrulama kayıplarının birbirine yakın olması, modelin overfitting yapmadığını ve genelleme yeteneğinin yüksek olduğunu göstermektedir.

4.2.2.4.2 RNN Eğitim ve Test Kaybı

Aşağıdaki grafik, RNN modelinin eğitim ve doğrulama veri setleri üzerindeki kayıplarını göstermektedir:



Şekil 23: RNN eğitim ve test kaybı

Bu grafikte, eğitim kaybının ve doğrulama kaybının epoch'lar boyunca nasıl değiştiği gösterilmektedir. RNN modelinde, eğitim kaybı ve doğrulama kaybı başlangıçta yüksek olup, epoch'lar ilerledikçe azalmaktadır. Ancak, doğrulama kaybının eğitim kaybına göre daha dalgalı olduğu ve bazı epoch'larda arttığı görülmektedir. Bu, modelin belirli epoch'larda overfitting

yapabileceğini göstermektedir. Ancak genel olarak, eğitim ve doğrulama kayıpları azalmaktadır, bu da modelin zamanla daha iyi performans gösterdiğini göstermektedir.

4.2.2.4.3 Eğitim ve Test Kaybı Karşılaştırma Sonucu

CNN ve RNN modellerinin eğitim ve test kayıpları incelendiğinde, her iki modelin de eğitim sürecinde kayıplarını azalttığı ve öğrenme sürecinde ilerleme kaydettiği görülmektedir. Ancak, CNN modelinin eğitim ve doğrulama kayıpları daha stabildir ve birbirine daha yakındır, bu da modelin genelleme yeteneğinin daha yüksek olduğunu ve overfitting yapmadığını göstermektedir. RNN modelinde ise doğrulama kaybının daha dalgalı olduğu ve bazı epoch'larda arttığı görülmektedir, bu da modelin overfitting yapabileceğini ve genelleme yeteneğinin CNN modeline göre daha düşük olabileceğini göstermektedir.

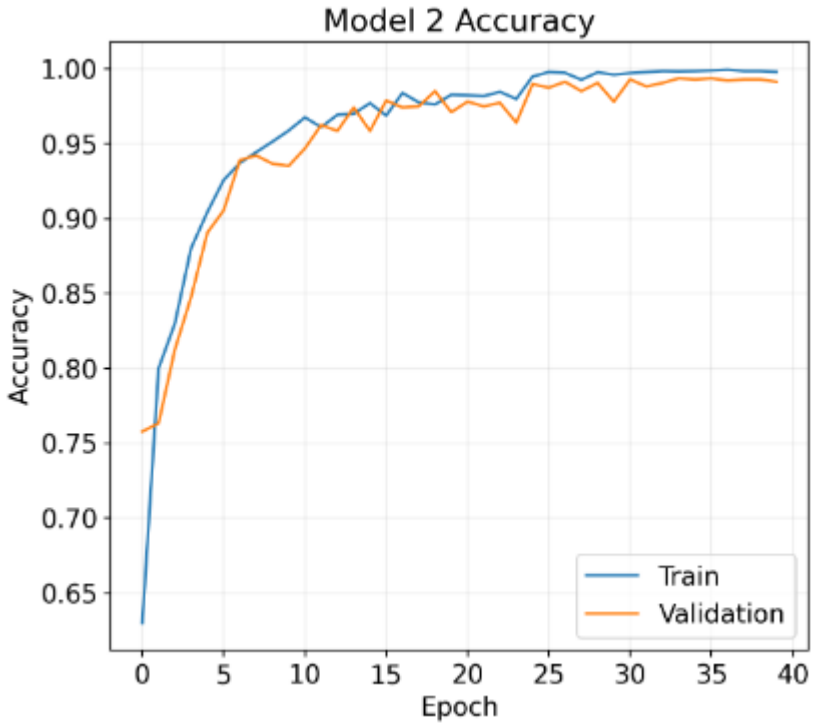
Genel olarak, CNN modeli eğitim ve test kayıpları açısından daha tutarlı ve stabildir, bu da modelin daha iyi bir genelleme yeteneğine sahip olduğunu göstermektedir. RNN modeli de etkili bir performans sergilemekte, ancak doğrulama kayıplarındaki dalgalanmalar modelin genelleme yeteneğini olumsuz etkileyebilir. Bu analiz, eğitim ve test kayıplarının model performansını değerlendirmede önemli bir rol oynadığını ve CNN modelinin bu açıdan daha avantajlı olduğunu göstermektedir.

4.2.2.5 Aşırı Öğrenme (Overfitting) Eğilimleri

Aşırı öğrenme (overfitting), makine öğrenmesi modellerinin eğitim veri setine çok iyi uyum sağlaması ancak yeni ve görülmemiş veriler üzerinde yetersiz performans göstermesi durumudur. Overfitting, modelin eğitim veri setindeki gürültü ve rastlantısal desenleri öğrenmesiyle ortaya çıkar, bu da genelleme yeteneğini olumsuz etkiler.

4.2.2.5.1 CNN Aşırı Öğrenme (Overfitting) Eğilimleri

CNN modeli, eğitim ve doğrulama kayıplarının yakından takip edilmesi sayesinde overfitting eğilimleri açısından analiz edilmiştir. Aşağıdaki grafik, CNN modelinin eğitim ve doğrulama kayıplarını göstermektedir:



Şekil 24: CNN aşırı öğrenme

CNN modelinde, eğitim ve doğrulama kayıplarının birbirine yakın olduğu ve epoch'lar ilerledikçe her ikisinin de azaldığı gözlemlenmektedir. Bu durum, modelin eğitim veri setine aşırı uyum sağlamadan genelleme yeteneğinin yüksek olduğunu göstermektedir. Veri artırma teknikleri, eğitim sürecinde overfitting'in önlenmesine yardımcı olmuş ve modelin doğrulama veri seti üzerinde de yüksek performans göstermesini sağlamıştır.

4.2.2.5.2 RNN Aşırı Öğrenme (Overfitting) Eğilimleri

RNN modeli ise eğitim sürecinde overfitting eğilimleri açısından incelenmiştir. Aşağıdaki grafik, RNN modelinin eğitim ve doğrulama kayıplarını göstermektedir:

Şekil 25: RNN aşırı öğrenme

RNN modelinde, eğitim kaybının doğrulama kaybına göre daha hızlı azaldığı ve doğrulama kaybının bazı epoch'larda arttığı gözlemlenmektedir. Bu dalgalanmalar, modelin overfitting yapma eğiliminde olduğunu ve doğrulama veri seti üzerinde genelleme yeteneğinin CNN modeline göre daha düşük olduğunu göstermektedir. RNN modelinin karmaşık yapısı ve ardışık veri işleme yetenekleri, eğitim sürecinde overfitting'in önlenmesi için daha fazla dikkat gerektirmektedir.

4.2.2.5.3 Aşırı Öğrenme (Overfitting) Eğilimleri Karşılaştırma Sonucu

Aşırı öğrenme eğilimleri açısından yapılan analiz, CNN modelinin eğitim ve doğrulama kayıplarının stabil ve birbirine yakın olduğunu, bu nedenle overfitting eğiliminin düşük olduğunu göstermektedir. RNN modelinde ise doğrulama kaybının dalgalanması ve bazı epoch'larda artması, overfitting eğiliminin daha yüksek olduğunu göstermektedir. Bu sonuçlar, CNN modelinin genelleme yeteneğinin daha güçlü olduğunu ve overfitting'e karşı daha dirençli olduğunu ortaya koymaktadır. RNN modeli ise belirli önlemler alınarak ve daha dikkatli eğitim süreçleriyle optimize edilmelidir.

Bu analiz, modellerin overfitting eğilimlerini değerlendirirken eğitim ve doğrulama kayıplarının incelenmesinin önemini vurgulamaktadır. CNN modeli, genelleme yeteneği ve overfitting'e karşı

dayanıklılık açısından daha avantajlıdır, ancak RNN modelinin de belirli senaryolarda güçlü performans gösterebileceği unutulmamalıdır.

4.2.2.6 Kullanılan Bellek ve Hesaplama Gücü (Memory and Computational Power)

CNN ve RNN modelleri, beyin tümörü sınıflandırmasında kullanılan bellek ve hesaplama gücü açısından belirgin farklılıklar göstermektedir. Bu bölümde, her iki modelin bellek ve hesaplama gücü gereksinimleri karşılaştırılacaktır.

CNN (Convolutional Neural Network) modelleri, genellikle görüntü işleme ve sınıflandırma görevlerinde kullanılır ve yüksek performans sunar. CNN konvolüsyonel katmanlar ve tam bağlantılı katmanları içeren bir mimariden oluşmaktadır. Bu yapısı sayesinde, model büyük miktarda veriyi paralel olarak işleyebilir. CNN modelleri, hesaplama gücü açısından oldukça verimlidir ve modern GPU'lar üzerinde hızlı bir şekilde çalıştırılabilir. Eğitim sürecinde kullanılan bellek miktarı, modelin derinliğine ve katman sayısına bağlıdır, ancak genellikle RNN modellerine göre daha düşüktür. CNN'ler, veri artırma teknikleriyle optimize edildiğinde, bellek ve hesaplama gücü gereksinimlerini minimize ederek yüksek doğruluk oranlarına ulaşabilir.

RNN (Recurrent Neural Network) modelleri, özellikle ardışık verilerin işlenmesi ve zaman serisi analizlerinde kullanılır. RNN'lerin mimarisi, zaman içinde sıralı veri noktalarını işlemek için tekrar eden katmanlar içerir. Bu yapı, RNN'lerin geçmiş bilgiye dayalı tahminler yapmasını sağlar, ancak aynı zamanda hesaplama gücü ve bellek gereksinimlerini artırır. RNN modelleri, özellikle uzun sekanslar üzerinde çalışırken, daha fazla bellek ve hesaplama gücüne ihtiyaç duyar. Eğitim süreci, CNN'lere kıyasla daha yavaş olabilir ve modern GPU'lar üzerinde bile daha uzun sürebilir. RNN'lerin hesaplama gücü gereksinimleri, modelin derinliği, katman sayısı ve kullanılan ardışık veri miktarına bağlı olarak artar.

CNN ve RNN modelleri, bellek ve hesaplama gücü gereksinimleri açısından belirgin farklılıklar göstermektedir. CNN modelleri, paralel veri işleme yetenekleri sayesinde genellikle daha verimli ve hızlıdır. Eğitim sürecinde daha az bellek kullanır ve modern GPU'lar üzerinde daha hızlı çalıştırılabilir. Buna karşılık, RNN modelleri ardışık veri işleme ve zaman serisi analizlerinde güçlü performans göstermesine rağmen, daha fazla bellek ve hesaplama gücüne ihtiyaç duyar ve eğitim süreci daha uzun sürebilir. Bu nedenle, model seçimi yapılırken kullanılacak veri türü, uygulama alanı ve mevcut hesaplama kaynakları dikkate alınmalıdır. CNN modelleri, görüntü işleme ve sınıflandırma görevlerinde daha verimli ve hızlı bir seçenek sunarken, RNN modelleri ardışık veriler ve zaman serisi analizleri için uygun bir alternatif olabilir.

4.2.2.7 Model Karmaşıklığı (Model Complexity) ve Veri Seti İle Uyum (Compatibility with Dataset)

CNN ve RNN modellerinin karmaşıklık düzeyleri, model mimarileri ve hesaplama gereksinimleri açısından karşılaştırıldığında belirgin farklılıklar göstermektedir. CNN modelleri, konvolüsyonel katmanlar ve tam bağlantılı katmanlar içerir. Bu yapı, görüntü işleme ve sınıflandırma görevlerinde yüksek performans sağlar, ancak modelin derinliği ve katman sayısı arttıkça karmaşıklığı da artar. CNN modelleri, hesaplama gücü açısından verimli olmalarına rağmen, derin modellerin eğitimi ve optimize edilmesi daha zorlu olabilir.

RNN modelleri ise ardışık veri işleme ve zaman serisi analizleri için özel olarak tasarlanmıştır. Bu modeller, zaman içinde sıralı veri noktalarını işlemek için tekrar eden katmanlar içerir ve bu nedenle daha karmaşık bir yapıdadır. RNN'ler, özellikle uzun sekanslar üzerinde çalışırken, hesaplama gücü ve bellek gereksinimleri açısından daha yüksek maliyetlere sahiptir. Ayrıca, RNN modellerinin eğitimi, geri yayılım algoritması ve zaman içindeki bağımlılıkların işlenmesi nedeniyle daha zordur ve bu da modeli daha karmaşık hale getirir.

Modellerin veri seti ile olan uyumu değerlendirildiğinde, CNN modelleri özellikle görüntü verileri ile yüksek uyum göstermektedir. CNN'in konvolüsyonel katmanları, görüntülerdeki mekansal ilişkileri ve özellikleri etkili bir şekilde çıkarır ve sınıflandırır. Beyin tümörü MRI görüntüleri gibi veri setlerinde, CNN modelleri genellikle yüksek doğruluk oranları ve güçlü performans sergiler. Bu uyum, CNN modellerinin tıbbi görüntü analizi ve diğer benzer uygulamalarda tercih edilmesini sağlar.

RNN modelleri ise ardışık veriler ve zaman serisi analizleri ile yüksek uyum göstermektedir. Bu modeller, zaman içinde değişen verileri ve ardışık bağımlılıkları etkili bir şekilde işler. Ancak, RNN'lerin görüntü verileri ile olan uyumu, CNN'ler kadar güçlü değildir. Beyin tümörü MRI görüntüleri gibi statik veri setlerinde, RNN modelleri CNN'lere kıyasla daha düşük performans gösterebilir. Bununla birlikte, RNN'lerin ardışık veri gerektiren uygulamalarda güçlü performans göstermesi, zaman serisi verilerinde ve doğal dil işleme gibi alanlarda tercih edilmesini sağlar.

4.2.2.8 Modelin Genelleme Yeteneği (Generalization Ability) ve İyileştirme ve Optimizasyon Teknikleri (Improvement and Optimization Techniques)

CNN ve RNN modellerinin genelleme yetenekleri, farklı veri setleri ve yeni veriler üzerindeki performanslarının karşılaştırılmasıyla değerlendirilebilir. CNN modelleri, özellikle görüntü verileri üzerinde yüksek genelleme yeteneğine sahiptir. Eğitim sürecinde kullanılan veri artırma teknikleri ve düzenlileme stratejileri, CNN modellerinin yeni ve görülmemiş veri setlerinde de yüksek doğruluk oranı ile performans göstermesini sağlar. CNN modelleri, geniş veri setlerinde ve çeşitli görüntü işleme görevlerinde başarılı bir şekilde uygulanabilir.

RNN modelleri ise ardışık veriler ve zaman serisi analizlerinde yüksek genelleme yeteneği göstermektedir. Bu modeller, geçmiş bilgiye dayalı tahminler yapma yetenekleri sayesinde, zaman içinde değişen verileri ve ardışık bağımlılıkları etkili bir şekilde işler. Ancak, RNN'lerin görüntü verileri üzerindeki genelleme yeteneği, CNN modellerine kıyasla daha düşüktür. RNN'ler, dil işleme ve zaman serisi analizleri gibi alanlarda güçlü genelleme yetenekleri sunar.

Modellerin performansını artırmak için kullanılan iyileştirme ve optimizasyon teknikleri de karşılaştırılabilir. CNN modellerinde, veri artırma teknikleri, düzenlileme (regularization) stratejileri ve hiperparametre optimizasyonu gibi yöntemler yaygın olarak kullanılır. Veri artırma teknikleri, modelin overfitting yapmasını engeller ve genelleme yeteneğini artırır. Ayrıca, öğrenme oranı, epoch sayısı ve mini-batch boyutu gibi hiperparametrelerin dikkatlice seçilmesi, modelin performansını maksimize eder.

RNN modellerinde ise, genelleme yeteneğini artırmak ve overfitting'i önlemek için dropout, L2 düzenlileme ve erken durdurma (early stopping) gibi teknikler kullanılır. Ayrıca, ardışık verilerin işlenmesi için dikkat mekanizmaları (attention mechanisms) ve uzun kısa süreli bellek (LSTM) hücreleri gibi yapılar da RNN modellerinin performansını artırmak için kullanılabilir. Hiperparametre optimizasyonu, RNN modellerinde de kritik bir rol oynar ve modelin öğrenme sürecini optimize eder.

Sonuç olarak, CNN ve RNN modelleri, genelleme yetenekleri ve iyileştirme teknikleri açısından farklılıklar göstermektedir. CNN modelleri, görüntü işleme görevlerinde yüksek genelleme yeteneği ve performans sunarken, RNN modelleri ardışık veri işleme ve zaman serisi analizlerinde güçlü performans sergiler. Her iki modelin de performansını artırmak için çeşitli iyileştirme ve optimizasyon teknikleri kullanılabilir ve bu tekniklerin seçimi, modelin uygulanacağı veri seti ve görev türüne bağlıdır.

4.2.2.9 Görselleştirme ve Sonuçların Yorumlanabilirliği (Visualization and Interpretability of Results) ve Uygulama Alanları ve Esneklik (Application Areas and Flexibility)

CNN ve RNN modellerinin sonuçlarının görselleştirilmesi ve yorumlanabilirliği, modellerin kullanım kolaylığı ve analiz edilebilirliği açısından önemlidir. CNN modelleri, özellikle görüntü verileri üzerinde çalıştığı için sonuçlarının görselleştirilmesi daha doğrudandır. Örneğin, konvolüsyonel katmanlarda öğrenilen filtreler, aktivasyon haritaları ve sınıflandırma sonuçları kolayca görselleştirilebilir. Bu görselleştirmeler, modelin hangi özellikleri öğrendiğini ve nasıl karar verdiğini anlamak için kullanılır. Görselleştirme teknikleri, modelin iç işleyişini anlamayı ve sonuçların yorumlanabilirliğini artırır.

RNN modelleri ise zaman serisi verileri ve ardışık veri işleme görevlerinde çalışır. Bu nedenle, sonuçların görselleştirilmesi ve yorumlanabilirliği, zaman içindeki değişikliklerin ve ardışık bağımlılıkların analizi ile ilgilidir. RNN modellerinde, dikkat mekanizmaları (attention mechanisms) ve bellek hücrelerinin (LSTM veya GRU) görselleştirilmesi, modelin hangi veri noktalarına odaklandığını ve nasıl karar verdiğini anlamak için kullanılır. Ancak, RNN modellerinin iç işleyişi ve karar süreçleri, CNN modellerine göre daha karmaşıktır ve görselleştirme teknikleri daha az belirgin olabilir.

Modellerin farklı uygulama alanlarındaki esneklikleri ve uyumları karşılaştırıldığında, CNN ve RNN modelleri belirgin farklılıklar gösterir. CNN modelleri, görüntü işleme, nesne tanıma, yüz tanıma ve tıbbi görüntü analizi gibi çeşitli alanlarda yaygın olarak kullanılır. Görüntü verilerini işleme yetenekleri sayesinde, CNN modelleri birçok uygulama alanında yüksek performans sergiler ve esneklik sağlar. Ayrıca, transfer öğrenme (transfer learning) teknikleri ile önceden eğitilmiş modellerin farklı görevlerde yeniden kullanılması mümkündür.

RNN modelleri dil işleme, zaman serisi analizleri, konuşma tanıma ve makine çevirisi gibi ardışık ve karmaşık veri işlemeler de yaygın olarak tercih edilir.. RNN'lerin ardışık bağımlılıkları öğrenme yetenekleri, dil modellemesi ve zaman serisi tahminleri gibi görevlerde üstün performans sergilemelerini sağlar. RNN modelleri, LSTM ve GRU gibi özel hücre yapıları sayesinde uzun süreli bağımlılıkları da öğrenebilir. Ancak, RNN'lerin karmaşık yapısı ve hesaplama gereksinimleri, belirli uygulama alanlarında esnekliklerini sınırlayabilir.

Sonuç olarak, CNN ve RNN modelleri, görselleştirme ve yorumlanabilirlik ile farklı uygulama alanlarında esneklik açısından belirgin farklılıklar göstermektedir. CNN modelleri, görüntü işleme ve sınıflandırma görevlerinde yüksek performans ve esneklik sunarken, RNN modelleri ardışık veri işleme ve zaman serisi analizlerinde güçlü performans sergiler. Her iki modelin de güçlü yönleri, kullanım amacına ve veri türüne bağlı olarak değerlendirilmeli ve optimize edilmelidir.

4.2.2.10 Veri Ön İşleme Gereksinimleri (Data Preprocessing Requirements)

CNN ve RNN modellerinin veri ön işleme gereksinimleri, model performansını doğrudan etkileyen kritik adımları içerir ve bu adımlar, modelin türüne göre değişiklik gösterir.

CNN modelleri, özellikle görüntü verileri üzerinde çalışırken belirli veri ön işleme tekniklerine ihtiyaç duyar. Bu teknikler arasında görüntü boyutlandırma, normalizasyon, veri artırma (augmentation) ve gürültü temizleme gibi işlemler bulunur. Görüntülerin boyutlandırılması, modelin sabit boyutlu girişleri kabul etmesini sağlar. Normalizasyon ile piksel değerlerinin belirli bir aralığa getirilir. Bu sayede model daha hızlı ve stabil bir şekilde öğrenir. Veri artırma teknikleri, eğitim veri setinin çeşitliliğini artırarak modelin genelleme yeteneğini geliştirir ve overfitting'i önler.

RNN modelleri ise ardışık veri ve zaman serisi analizlerinde belirli veri ön işleme gereksinimlerine sahiptir. Bu modeller için veri normalizasyonu, ardışık veri segmentasyonu ve zaman damgalarının işlenmesi önemlidir. Verilerin normalizasyonu, modelin hızlı ve stabil bir şekilde öğrenmesini sağlar. Ardışık veri segmentasyonu, verilerin zaman içindeki bağımlılıklarını öğrenebilmesi için önemlidir. Ayrıca, RNN modelleri genellikle eksik veri noktaları ve düzensiz zaman serileriyle çalışırken, bu verilerin düzgün bir şekilde işlenmesi ve doldurulması gerekmektedir.

Veri ön işleme hem CNN hem de RNN modellerinin performansını arttırmak ve dengelemek için kritiktir. CNN modelleri için görüntü işleme teknikleri, modelin giriş verilerini standart hale getirmek ve çeşitliliği artırmak için kullanılır. RNN modelleri için ise zaman serisi ve ardışık verilerin doğru bir şekilde segmentasyonu ve normalizasyonu, modelin ardışık bağımlılıkları etkili bir şekilde öğrenmesini sağlar. Her iki modelde de uygun veri ön işleme adımlarının uygulanması, modelin genel performansını ve genelleme yeteneğini artırarak daha güvenilir ve doğru sonuçlar elde edilmesini sağlar.

5. TARTŞMA VE SONUÇ

Bu çalışmada, Convolutional Neural Networks (CNN) ve Recurrent Neural Networks (RNN) modellerinin beyin tümörlerinin teşhisindeki performansları karşılaştırılarak değerlendirildi. Aynı veri seti kullanılarak her iki modelin eğitim süreleri, doğruluk oranları, verimlilikleri, hızları ve diğer performans kriterleri ayrıntılı bir şekilde incelendi. Elde edilen sonuçlar, her iki modelin de belirli güçlü yönlere sahip olduğunu, ancak farklı uygulama alanlarında daha uygun olabileceklerini göstermektedir.

CNN modeli, özellikle görüntü işleme görevlerinde üstün performans sergilemiştir. %99 doğruluk oranına ulaşan CNN modeli, eğitim süresinde de verimli olup, büyük veri setlerini hızlı bir şekilde işleyebilme yeteneğini göstermiştir. Modelin mimarisi, görüntülerdeki mekansal ilişkileri ve özellikleri etkili bir şekilde çıkararak yüksek doğruluk oranlarına ulaşmasını sağlamıştır. Eğitim sürecinde kullanılan veri artırma teknikleri ve optimizasyon stratejileri, modelin genelleme yeteneğini artırmış ve overfitting'i önlemiştir. CNN modelinin bellek ve hesaplama gücü gereksinimleri, paralel veri işleme yetenekleri sayesinde genellikle daha düşüktür ve modern GPU'lar üzerinde hızlı çalışabilmektedir.

RNN modeli ise ardışık veri işleme ve zaman serisi analizlerinde güçlü performans sergilemiştir. Eğitim sürecinde yüksek doğruluk oranlarına ulaşan RNN modeli, zaman içindeki bağımlılıkları etkili bir şekilde işleyebilme yeteneği ile dikkat çekmiştir. Ancak, RNN'in eğitim süresi daha uzun olup, hesaplama gücü ve bellek gereksinimleri de daha yüksektir. Modelin karmaşık yapısı ve ardışık veri işleme yetenekleri, belirli senaryolarda güçlü performans göstermesini sağlar, ancak eğitim süreci ve optimize edilmesi daha fazla dikkat gerektirir.

Her iki modelin eğitim ve test kayıpları incelendiğinde, CNN modelinin eğitim ve doğrulama kayıplarının daha stabil ve birbirine yakın olduğu, bu nedenle overfitting eğiliminin düşük olduğu gözlemlenmiştir. RNN modelinde ise doğrulama kaybının dalgalanması ve bazı epoch'larda artması, overfitting eğiliminin daha yüksek olduğunu göstermektedir. Bu sonuçlar, CNN modelinin genelleme yeteneğinin daha güçlü olduğunu ve overfitting'e karşı daha dirençli olduğunu ortaya koymaktadır. Ancak, RNN modeli de belirli önlemler alınarak ve daha dikkatli eğitim süreçleriyle optimize edildiğinde etkili performans sergileyebilir.

Modellerin sonuçlarının görselleştirilmesi ve yorumlanabilirliği açısından, CNN modelleri özellikle görüntü verileri üzerinde çalıştığı için daha doğrudan ve anlaşılır sonuçlar sunmaktadır. RNN modellerinin sonuçlarının görselleştirilmesi ve yorumlanabilirliği ise zaman içindeki değişikliklerin ve ardışık bağımlılıkların analizi ile ilgilidir ve daha karmaşık olabilir. Uygulama alanları açısından, CNN modelleri görüntü işleme ve sınıflandırma görevlerinde yüksek performans ve esneklik sunarken, RNN modelleri ardışık veri işleme ve zaman serisi analizlerinde güçlü performans sergiler.

Sonuç olarak, CNN ve RNN modelleri, beyin tümörü teşhisinde etkili ve güvenilir araçlar olarak değerlendirilebilir. Kullanım amacına, veri setinin özelliklerine ve mevcut hesaplama kaynaklarına bağlı olarak, her iki modelin de güçlü yönleri bulunmaktadır. CNN modelleri, görüntü işleme yetenekleri sayesinde beyin tümörü sınıflandırmasında daha tutarlı ve güvenilir sonuçlar verirken, RNN modelleri de ardışık veri işleme konusundaki güçlü yönleri ile dikkate değer bir seçenek olarak öne çıkmaktadır. Bu çalışmanın sonuçları, beyin tümörü teşhisinde en uygun yöntemin seçilmesine katkı sağlayacak ve gelecekteki araştırmalara ışık tutacak değerli bilgiler sunmaktadır.

EKLER

```
pip install --upgrade tensorflow tensorflow-io

# Genel İçe Aktarımlar
import tensorflow as tf
import pandas as pd
import numpy as np
import random
import os

# Görselleştirme
import matplotlib.pyplot as plt
import seaborn as sns

# Model Oluşturma
from keras.utils import plot_model
from tensorflow.keras import models
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.optimizers import legacy

# Training Model
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.callbacks import ReduceLROnPlateau
from tensorflow.keras.callbacks import ModelCheckpoint

# Veri İşleme
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import array_to_img
from tensorflow.keras.preprocessing.image import load_img

# Python uyumluluğunu sağlamak için IPython sihirli komutları yorum satırı haline getirildi.
```

```
# Global değişkenler
SAVE = False
SEED = 111

# Tutarlı sonuçlar için seed (tohum) ayarı yapılıyor
tf.keras.utils.set_random_seed(SEED)
tf.random.set_seed(SEED)
np.random.seed(SEED)

# Veri görselleştirme güncellemeleri
# %config InlineBackend.figure_format = 'retina'
plt.rcParams["figure.figsize"] = (16, 10)
plt.rcParams.update({'font.size': 14})

# Veri Sınıflandırmaları
CLASS_TYPES = ['pituitary', 'notumor', 'meningioma', 'glioma']
N_TYPES = len(CLASS_TYPES)

# Verileri içe aktarma fonksiyonu
def get_data_labels(directory, shuffle=True, random_state=0):
    """
    Function used for going into the main training directory
    whose directory has sub-class-types.
    """
    from sklearn.utils import shuffle
    import os

    # Verileri ve etiketleri depolamak için listeler
    data_path = []
    data_labels = []

    for label in os.listdir(directory):
        label_dir = os.path.join(directory, label)

        # MacOS'un yol bilgisi kaydetmesini önlemek için
        if not os.path.isdir(label_dir):
            continue

        # Her klasöre girip resim yolunu alma
```

```
        for image in os.listdir(label_dir):
            image_path = os.path.join(label_dir, image)
            data_path.append(image_path)
            data_labels.append(label)

    if shuffle:
        data_path, data_labels = shuffle(data_path, data_labels, random_state=random_state)

    return data_path, data_labels

from google.colab import files
files.upload()

!pip install kaggle
import os
os.environ['KAGGLE_CONFIG_DIR'] = '/content/.kaggle'
!kaggle datasets download -d masoudnickparvar/brain-tumor-mri-dataset
!unzip brain-tumor-mri-dataset.zip

# Eğitim ve test için dosya yollarını ayarlama
# USER_PATH = r"/input/brain-tumor-mri-dataset"
USER_PATH = r""
train_dir = USER_PATH + r'Training/'
test_dir = USER_PATH + r'Testing/'

# Yukarıdaki fonksiyonu kullanarak verileri alma
train_paths, train_labels = get_data_labels(train_dir)
test_paths, test_labels = get_data_labels(test_dir)

# Eğitim ve test örneklerinin boyutlarını yazdırma


print('Training')
print(f'Number of Paths: {len(train_paths)}')
print(f'Number of Labels: {len(train_labels)}')
print('\nTesting')
print(f'Number of Paths: {len(test_paths)}')
print(f'Number of Labels: {len(test_labels)}')
```

```
_, ax = plt.subplots(ncols=3, figsize=(20, 14))

# Eğitim veri türlerini görselleştirme
class_counts = [len([x for x in train_labels if x == label]) for label in CLASS_TYPES]
print('Training Counts')
print(dict(zip(CLASS_TYPES, class_counts)))

ax[0].set_title('Training Data')
ax[0].pie(
    class_counts,
    labels=[label.title() for label in CLASS_TYPES],
    colors=['#FAC500','#0BFA00', '#0066FA','#FA0000'],
    autopct=lambda p: '{:.2f}%\n{:,.0f}'.format(p, p * sum(class_counts) / 100),
    explode=tuple(0.01 for i in range(N_TYPES)),
    textprops={'fontsize': 20}
)

# Eğitim ve test bölünmesinin dağılımını görselleştirme
ax[1].set_title('Train Test Split')
ax[1].pie(
    [len(train_labels), len(test_labels)],
    labels=['Train','Test'],
    colors=['darkcyan', 'orange'],
    autopct=lambda p: '{:.2f}%\n{:,.0f}'.format(p, p * sum([len(train_labels), len(test_labels)]) / 100),
    explode=(0.1, 0),
    startangle=85,
    textprops={'fontsize': 20}
)

# Test veri türlerini görselleştirme
class_counts = [len([x for x in test_labels if x == label]) for label in CLASS_TYPES]
print('\nTesting Counts')
print(dict(zip(CLASS_TYPES, class_counts)))

ax[2].set_title('Testing Data')
ax[2].pie(
    class_counts,
    labels=[label.title() for label in CLASS_TYPES],
    colors=['#FAC500', '#0BFA00', '#0066FA', '#FA0000'],
```

```
    autopct=lambda p: '{:.2f}%\n{:,.0f}'.format(p, p * sum(class_counts) / 100),
    explode=tuple(0.01 for i in range(N_TYPES)),  # Daha iyi görselleştirme için dilimleri hafifçe ayırma
    textprops={'fontsize': 20}  # Pasta grafikteki metin için yazı tipi boyutunu ayarlama
)


plt.show()

# Çıktıyı test etmek için görüntü alma
im = load_img(train_paths[3], target_size=(150, 150))
im = img_to_array(im)

# Bunu (1, 150, 150, 3) şeklinde yeniden boyutlandır
im = np.expand_dims(im, axis=0)
print(f'x reshaped: {im.shape}')

# Normalizasyon tensörü
im /= np.max(im) # ~ np.max(img_tensor)

# Diziyi yeniden görüntü formatına dönüştürme

im = array_to_img(im[0])
display(im)

# Verilen indekslere göre bir dizi resmi görüntülemek için fonksiyon
def show_images(paths, label_paths, index_list=range(10), im_size=250, figsize=(12, 8), save=False):
    """
    Show images from a given path based on the inputted
    list indices related to the desired images one wishes
    to see.
    """

    num_images = len(index_list)
    num_rows = (num_images + 3) // 4

    _, ax = plt.subplots(nrows=num_rows, ncols=4, figsize=figsize)
    ax = ax.flatten()

    for i, index in enumerate(index_list):
```

```
        if i >= num_images:
            break

        image = load_img(paths[index], target_size=(im_size, im_size))
        ax[i].imshow(image)
        ax[i].set_title(f'{index}: {label_paths[index]}')
        ax[i].axis('off')

    plt.tight_layout()

    if save:
        plt.savefig('show_image.pdf')
    else:
        plt.show()

# Üç farklı açıdan dört farklı veri sınıflandırma resmi (resimler birbirinden bağımsız)
show_images(train_paths, train_labels, im_size=350, figsize=(13,10),
            index_list=[0, 94, 235, 17,
                        61, 324, 55, 45,
                        374, 65, 391, 488])

# Görüntü boyutu
image_size = (150, 150)

# Eğitim batch boyutu
batch_size = 32

# Veri artırma ve ön işleme
train_datagen = ImageDataGenerator(rescale=1./255,
                                   rotation_range=10,
                                   brightness_range=(0.85, 1.15),
                                   width_shift_range=0.002,
                                   height_shift_range=0.002,
                                   shear_range=12.5,
                                   zoom_range=0,
                                   horizontal_flip=True,
                                   vertical_flip=False,
                                   fill_mode="nearest")
```

```
# Eğitim verilerine sabit seed ile jeneratör uygulama
train_generator = train_datagen.flow_from_directory(train_dir,
                                                    target_size=image_size,
                                                    batch_size=batch_size,
                                                    class_mode="categorical",
                                                    seed=SEED)

# Test verilerinde artırma yok, sadece yeniden boyutlandırma
test_datagen = ImageDataGenerator(rescale=1./255)

# Test verilerine sabit seed ile jeneratör uygulama
test_generator = test_datagen.flow_from_directory(test_dir,
                                                  target_size=image_size,
                                                  batch_size=batch_size,
                                                  class_mode="categorical",
                                                  shuffle=False,
                                                  seed=SEED)

# Eğitim veri jeneratörü için sınıf indekslerine erişme
class_indices_train = train_generator.class_indices
class_indices_train_list = list(train_generator.class_indices.keys())


# Kategorik türleri gösterme
print("Categorical types for the training data:")
print(class_indices_train)

def show_ImageDataGenerator(data_generator, num_samples=5, figsize=(12, 12), save=False):
    """
    Function to visualize how the ImageDataGenerator augments the data
    """

    # Artırılmış örnekler oluşturma
    augmented_samples = next(data_generator)

    # Parti içerisinden resimleri çıkarma
    images = augmented_samples[0][:num_samples]
```

```
    # Artırılmış resimleri gösterme
    fig, axes = plt.subplots(1, num_samples, figsize=figsize)

    for i, ax in enumerate(axes):
        ax.imshow(images[i])
        ax.axis('off')

    plt.tight_layout()

    if save:
        plt.savefig('show_ImageDataGenerator.pdf')

    plt.show()

SAVE = False
show_ImageDataGenerator(train_generator, num_samples=5, figsize=(12.5, 8), save=SAVE)

# Görüntü boyutu: yükseklik, genişlik, RBG
image_shape = (image_size[0], image_size[1], 3)

# Eğitim epoch'ları (dönemleri)
epochs = 40

# Epoch başına adım sayısı
steps_per_epoch = train_generator.samples // batch_size

# Doğrulama adımları
validation_steps = test_generator.samples // batch_size

print(f'Image shape: {image_shape}')
print(f'Epochs: {epochs}')
print(f'Batch size: {batch_size}')
print(f'Steps Per Epoch: {steps_per_epoch}')
print(f'Validation steps: {validation_steps}')

def plot_sample_predictions(model, test_generator, categories, test_dir, num_samples=9, figsize=(12, 8)):
    """
    Nice display of prediction samples to see CNN predictions
    for classification.
```

```
    """
    # Test veri seti üzerinde tahminler yapma
    predictions = model.predict(test_generator)
    predicted_categories = np.argmax(predictions, axis=1)
    true_categories = test_generator.classes

    # Rastgele test resimleri seçme
    test_images = np.array(test_generator.filepaths)
    sample_indices = np.random.choice(len(test_images), size=num_samples, replace=False)
    sample_images = test_images[sample_indices]
    sample_predictions = [categories[predicted_categories[i]] for i in sample_indices]
    sample_true_labels = [categories[true_categories[i]] for i in sample_indices]

    # Tahmin edilen ve gerçek etiketleriyle örnek resimleri görselleştirme
    plt.figure(figsize=figsize)

    # Örnekler üzerinde döngü oluşturma
    for i, image_path in enumerate(sample_images):
        # Alt grafik oluşturma ve çizme
        plt.subplot(3, 3, i + 1)
        img = plt.imread(image_path)
        plt.imshow(img)
        plt.axis("off")

        # Doğru tahmine bağlı olarak eksen etiket rengini ayarlama
        prediction_color = 'green' if sample_predictions[i] == sample_true_labels[i] else 'red'
        plt.title(f"Predicted: {sample_predictions[i]}\nTrue: {sample_true_labels[i]}", color=prediction_color)

    plt.tight_layout()
    plt.show()


def CM(CNN_model, test_generator, categories):
    """
    Function to return the confusion matrix of a given CNN model.
    """
    from sklearn.metrics import confusion_matrix
    # Test veri seti üzerindeki tahminler
    predictions = CNN_model.predict(test_generator)
```

```
    predicted_categories = np.argmax(predictions, axis=1)
    true_categories = test_generator.classes

    # Bir karışıklık matrisi oluşturma
    confusion_matrix_array = confusion_matrix(true_categories, predicted_categories)

    return confusion_matrix_array


def calculate_metrics(confusion_matrix, categories):
    """
    Function to calculate important metrics for multi-classification problems.
    """
    # Dört farklı metriği hesaplama
    precision = np.diag(confusion_matrix) / np.sum(confusion_matrix, axis=0)
    recall = np.diag(confusion_matrix) / np.sum(confusion_matrix, axis=1)
    f1_score = 2 * (precision * recall) / (precision + recall)
    accuracy = np.sum(np.diag(confusion_matrix)) / np.sum(confusion_matrix)

    # Her kategoriye göre sonuçları yazdırma
    for i, category in enumerate(categories):
        print(f"Class: {category.title()}")
        print(f"Precision: {precision[i]:.3f}")
        print(f"Recall: {recall[i]:.3f}")
        print(f"F1-Score: {f1_score[i]:.3f}\n")

    # Modelin toplam doğruluğunu gösterme
    print(f"\nAccuracy: {accuracy:.3f}")

# Model mimarisini tanımlama
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
def build_model(input_shape, num_classes):
    model = Sequential([
        # Convolutional layer 1
        Conv2D(32, (4, 4), activation="relu", input_shape=input_shape),
        MaxPooling2D(pool_size=(3, 3)),
```

```
        # Convolutional layer 2
        Conv2D(64, (4, 4), activation="relu"),
        MaxPooling2D(pool_size=(3, 3)),

        # Convolutional layer 3
        Conv2D(128, (4, 4), activation="relu"),
        MaxPooling2D(pool_size=(3, 3)),

        # Convolutional layer 4
        Conv2D(128, (4, 4), activation="relu"),
        Flatten(),

        # Tam bağlı katmanlar
        Dense(512, activation="relu"),
        Dropout(0.5),
        Dense(num_classes, activation="softmax")
    ])
    return model

image_shape = (150, 150, 3)  # Örnek giriş boyutu, kendi verinize göre ayarlayın
num_classes = 4  # Örnek sınıf sayısı, kendi verinize göre ayarlayın
model = build_model(image_shape, num_classes)

model.summary()

optimizer = Adam(learning_rate=0.001, beta_1=0.869, beta_2=0.995)
model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['accuracy'])

!pip install visualkeras

from visualkeras import layered_view

# Modeli görselleştir
layered_view(model, legend=True, max_xy=300)

# model_visual = models.Model(inputs=model.input, outputs=model.output)

# # Save model architecture to a file
# plot_model(model_visual, show_dtype=True, to_file='model_architecture.png', show_shapes=True)
```

```
# # Display model architecture in the notebook

# from IPython.display import Image
# Image(retina=True, filename='model_architecture.png')

# Kayıp azalmaya devam etmezse eğitimi durdur
model_es = EarlyStopping(monitor='loss', min_delta=1e-9, patience=8, verbose=True)
model_rlr = ReduceLROnPlateau(monitor='val_loss', factor=0.3, patience=5, verbose=True)

# Modeli eğitme
history = model.fit(train_generator,
                    steps_per_epoch=steps_per_epoch,
                    epochs=epochs,
                    validation_data=test_generator,
                    validation_steps=validation_steps,
                    callbacks=[model_es, model_rlr])

# Modeli değerlendirme
loss, accuracy = model.evaluate(test_generator, steps=test_generator.samples//batch_size)
print(f"Test Loss: {loss:0.5f}")
print(f"Test Accuracy: {accuracy:0.5f}")

_, ax = plt.subplots(ncols=2, figsize=(15, 6))

# Eğitim ve doğrulama doğruluğunu epoch'lar boyunca görselleştirme
ax[0].plot(history.history['accuracy'])
ax[0].plot(history.history['val_accuracy'])
ax[0].set_title('Model 2 Accuracy')
ax[0].set_xlabel('Epoch')
ax[0].set_ylabel('Accuracy')
ax[0].legend(['Train', 'Validation'])
ax[0].grid(alpha=0.2)

# Eğitim ve doğrulama kaybını epoch'lar boyunca görselleştirme
ax[1].plot(history.history['loss'])
ax[1].plot(history.history['val_loss'])
ax[1].set_title('Model 2 Loss')
ax[1].set_xlabel('Epoch')
```

```
ax[1].set_ylabel('Loss')
ax[1].legend(['Train', 'Validation'])
ax[1].grid(alpha=0.2)

plt.show()

# Karışıklık matrisini görselleştirme
confusion_matrix = CM(CNN_model=model, test_generator=test_generator,
categories=class_indices_train_list)

plt.figure(figsize=(8,8))
sns.heatmap(confusion_matrix, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("True")
plt.xticks(ticks=np.arange(N_TYPES) + 0.5,
        labels=[name.title() for name in class_indices_train_list], ha='center')
plt.yticks(ticks=np.arange(N_TYPES) + 0.5,
        labels=[name.title() for name in class_indices_train_list], va='center')
plt.show()

# Metrikleri gösterme
calculate_metrics(confusion_matrix, categories=class_indices_train_list)

# Sonuçları göstermek için 6.1'deki fonksiyonları kullanma
plot_sample_predictions(model=model,
                    test_generator=test_generator,
                    categories=class_indices_train_list,
                    test_dir=test_dir,
                    num_samples=9,
                    figsize=(13, 12))

# Kanal haritası grafiği
def plot_channel_activation_maps(model, image, images_per_row=16, N=8, save=False):
    """
    Function to visualize how the first N layers of the model observe the input image.

    Parameters:
        model (tensorflow.keras.models.Model): The Keras model for which to visualize the activation maps.
```

```
    image (numpy.ndarray): The input image for which to generate activation maps.
    images_per_row (int): Number of activation maps to display per row in the grid.
    N (int): Number of layers to visualize.
    save (bool): If True, save the plots as PDF files.

Returns:
    None
"""
from tensorflow.keras.models import Model

# İlk N katman için aktivasyonları çıktı olarak veren bir alt model oluştur
activation_model = Model(inputs=model.input, outputs=[layer.output for layer in model.layers[:N]])
activations = activation_model.predict(image)

# Grafikleri etiketlemek için katmanların isimlerini al
layer_names = [layer.name for layer in model.layers[:N]]

# Her katman için özellik haritalarını görselleştir
for layer_name, layer_activation in zip(layer_names, activations):
    # Bu, özellik haritasındaki özelliklerin sayısıdır
    n_features = layer_activation.shape[-1]
     Özellik haritası (1, boyut, boyut, n_özellik) şekline sahiptir
    size = layer_activation.shape[1]
    # Aktivasyon kanallarını bu matris içinde döşeyeceğiz
    n_cols = n_features // images_per_row
    display_grid = np.zeros((size * n_cols, images_per_row * size))

    # Her filtreyi bu büyük yatay ızgaraya döşeyeceğiz
    for col in range(n_cols):
        for row in range(images_per_row):
            channel_image = layer_activation[0, :, :, col * images_per_row + row]
            # Özelliği görsel olarak daha anlaşılır hale getirmek için son işlemden geçir
            channel_image -= channel_image.mean()
            epsilon = 1e-8  # Sıfıra bölmeyi önlemek için küçük bir epsilon değeri
            channel_std = channel_image.std() + epsilon
            channel_image /= channel_std
            channel_image *= 64
            channel_image += 128
            channel_image = np.clip(channel_image, 0, 255).astype('uint8')
```

```
            display_grid[col * size: (col + 1) * size,
                         row * size: (row + 1) * size] = channel_image

        # Izgarayı göster
        scale = 1. / size
        plt.figure(figsize=(scale * display_grid.shape[1],
                            scale * display_grid.shape[0]))
        plt.title(layer_name)
        plt.grid(False)
        plt.axis('off')
        plt.imshow(display_grid, aspect='auto', cmap='viridis')

        if save:
            plt.savefig(f'plot_channel_activation_maps_{layer_name}.pdf')

    plt.show()

# Test jeneratöründen bir sonraki partiyi al
batch_images, batch_labels = next(test_generator)

# İlk resmi partiden çıkar
image, label = batch_images[0], batch_labels[0]
image_tensor = np.expand_dims(image, axis=0)

# Test jeneratöründen sınıf indekslerini al
class_indices = test_generator.class_indices

# Convert the one-hot encoded label to the class name
label_name = [k for k, v in class_indices.items() if np.argmax(label) == v][0]

# Display the class name
print(f"Class name of the first image: {label_name}")
print(f'Shape {image_tensor.shape}')
array_to_img(image_tensor[0])

plot_channel_activation_maps(model=model, image=image_tensor, N=5, save=SAVE)

# Yanlış sınıflandırılmış görüntülerin görselleştirilmesi
```

```python
def visualize_misclassified_images(model, test_generator, class_indices):
    """
    Visualize misclassified images from the test set alongside their predicted and true labels.

    Parameters:
        model (tensorflow.keras.models.Model): The trained Keras model.
        test_generator (tensorflow.keras.preprocessing.image.DirectoryIterator): The test data generator.
        class_indices (dict): Dictionary mapping class names to their corresponding integer labels.

    Returns:
        None
    """

    from tensorflow.keras.preprocessing.image import array_to_img

    misclassified_images = []
    misclassified_labels_true = []
    misclassified_labels_pred = []

    for i in range(len(test_generator)):
        batch_images, batch_labels = next(test_generator)
        batch_predictions = model.predict(batch_images, verbose=False)
        predicted_labels = [list(class_indices.keys())[np.argmax(pred)] for pred in batch_predictions]
        true_labels = [list(class_indices.keys())[np.argmax(label)] for label in batch_labels]

        for j in range(len(batch_images)):
            if predicted_labels[j] != true_labels[j]:
                misclassified_images.append(batch_images[j])
                misclassified_labels_true.append(true_labels[j])
                misclassified_labels_pred.append(predicted_labels[j])

    # Yanlış sınıflandırılmış görüntüleri, gerçek ve tahmin edilen etiketlerle birlikte göster
    num_misclassified = len(misclassified_images)
    num_rows = int(np.ceil(num_misclassified / 4))
    plt.figure(figsize=(12, 3 * num_rows))

    for i in range(num_misclassified):
        plt.subplot(num_rows, 4, i + 1)
        plt.title(f"True: {misclassified_labels_true[i]}\nPred: {misclassified_labels_pred[i]}", color='red')
```

```
        plt.imshow(array_to_img(misclassified_images[i]))
        plt.axis('off')

    plt.tight_layout()
    plt.show()

# Yukarıdaki fonksiyonu kullanarak
visualize_misclassified_images(model, test_generator, test_generator.class_indices)
```

```
!pip install tensorflow==2.15.0
!pip install tensorflow-addons

import tensorflow as tf
import gc
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout, GlobalAveragePooling2D, Activation, BatchNormalization,
LSTM, ConvLSTM2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.applications import VGG19
from tensorflow.keras.layers import Input, Conv2D, SeparableConv2D, MaxPool2D, LeakyReLU, Lambda,
Reshape, Bidirectional
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.optimizers import Adam, RMSprop
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint, TensorBoard,
TerminateOnNaN, LearningRateScheduler, CSVLogger
from tensorflow.keras.losses import binary_crossentropy
from tensorflow.keras.layers import DepthwiseConv2D, ZeroPadding2D, Add, MaxPooling2D, Concatenate,
GlobalMaxPooling2D, GlobalAveragePooling2D
from tensorflow.keras.regularizers import l2
from tensorflow.keras import layers
from tensorflow.keras import backend as K
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split, StratifiedKFold, RepeatedStratifiedKFold
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix, roc_auc_score, roc_curve,
auc
import matplotlib.pyplot as plt
```

```
from tensorflow.keras.utils import plot_model
from tensorflow.keras import preprocessing
from keras.applications import imagenet_utils
import pandas as pd
import numpy as np
import random
import keras
import shutil
import pathlib
import itertools
import cv2
import os
import matplotlib.image as mpimg
import seaborn as sns

from google.colab import files
files.upload()

!pip install kaggle
import os
os.environ['KAGGLE_CONFIG_DIR'] = '/content/.kaggle'
!kaggle datasets download -d masoudnickparvar/brain-tumor-mri-dataset
!unzip brain-tumor-mri-dataset.zip

train_dir = 'Training/'
test_dir = 'Testing/'

for dirpath, dirnames, filenames in os.walk(train_dir):
  print(f"There are {len(dirnames)} directories and {len(filenames)} images in '{dirpath}'.")

for dirpath, dirnames, filenames in os.walk(test_dir):
  print(f"There are {len(dirnames)} directories and {len(filenames)} images in '{dirpath}'.")

def view_random_image(target_dir, target_class):
  target_folder = target_dir+target_class
  random_image = random.sample(os.listdir(target_folder), 1)
  img = mpimg.imread(target_folder + "/" + random_image[0])
  plt.imshow(img)
  plt.title(target_class)
```

```
  plt.axis("off");
  print(f"Image shape: {img.shape}")
  return img

img = view_random_image(target_dir=test_dir,
                        target_class="glioma")

EPOCHS = 50
from math import floor
N_FOLDS = 5
INIT_LR = 1e-3
T_BS = 16
V_BS = 16
decay_rate = 0.95
decay_step = 1
IMAGE_SIZE = [224,224]

!pip install tensorflow-addons
import tensorflow_addons as tfa

def augment_image(image, label):
    # Artırma dönüşümlerini uygulama
    image = tf.image.random_flip_left_right(image)
    image = tf.image.random_flip_up_down(image)
    image = tf.image.random_brightness(image, max_delta=0.1)
    image = tf.image.random_contrast(image, lower=0.8, upper=1.2)
    image = tf.image.random_saturation(image, lower=0.8, upper=1.2)
    image = tf.image.random_hue(image, max_delta=0.1)

    # Genişlik ve yükseklik kaydırmalarını uygulama
    width_shift = tf.random.uniform([], -0.2, 0.2) * tf.cast(tf.shape(image)[1], tf.float32)
    height_shift = tf.random.uniform([], -0.2, 0.2) * tf.cast(tf.shape(image)[0], tf.float32)
    image = tfa.image.translate(image, [width_shift, height_shift])

    return image, label

train_data = tf.keras.preprocessing.image_dataset_from_directory(train_dir,
                                                 label_mode="categorical",
                                                 batch_size=32,
```

```
                                                image_size=IMAGE_SIZE)

test_data = tf.keras.preprocessing.image_dataset_from_directory(test_dir,
                                                label_mode="categorical",
                                                image_size=IMAGE_SIZE,
                                                shuffle=False)

# Map fonksiyonunu kullanarak eğitim veri kümesine artırma uygulama
train_dataset_augmented = train_data.map(augment_image)

# def Combined_model():
#     # Input layer
#     input_layer = Input(shape=(224, 224, 3))

#     # Base VGG19 model as a feature extractor
#     baseModel = VGG19(weights=None, include_top=False, input_tensor=input_layer)

#     # Load the weights from the local file (specify the path)
#     baseModel.load_weights('/kaggle/input/vgg19-
weight/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5')

#     # Freeze the layers of the VGG19 model
#     for layer in baseModel.layers:
#         layer.trainable = False

#     x = baseModel.output

#     # LSTM layer
#     x = Reshape((49, 512))(x)
#     x = LSTM(512, activation="relu", return_sequences=True, trainable=False)(x)
#     x = BatchNormalization()(x)

#     # FC layer
#     x = Flatten(name="flatten")(x)

#     # fc1 layer
#     x = Dense(units=4096, activation='relu')(x)
#     x = BatchNormalization()(x)
```

```
#     # fc2 layer
#     x = Dense(units=4096, activation='relu')(x)
#     x = BatchNormalization()(x)

#     # Output layer
#     output = Dense(units=4, activation='softmax')(x)

#     model = Model(inputs=input_layer, outputs=output)
#     opt = Adam(lr=1e3)
#     model.compile(loss='categorical_crossentropy', optimizer=opt, metrics=["accuracy"])

#     return model

# # Create the model
# model = Combined_model()

# # Print the model summary
# model.summary()

def Combined_model():
    # Input layer
    input_layer = Input(shape=(224, 224, 3))

    # Base VGG19 model as a feature extractor with pre-trained weights
    baseModel = VGG19(weights='imagenet', include_top=False, input_tensor=input_layer)

    # Freeze the layers of the VGG19 model
    for layer in baseModel.layers:
        layer.trainable = False

    x = baseModel.output

    # LSTM layer
    x = Reshape((49, 512))(x)
    x = LSTM(512, activation="relu", return_sequences=True, trainable=False)(x)
    x = BatchNormalization()(x)

    # FC layer
    x = Flatten(name="flatten")(x)
```

```
    # fc1 layer
    x = Dense(units=4096, activation='relu')(x)
    x = BatchNormalization()(x)

    # fc2 layer
    x = Dense(units=4096, activation='relu')(x)
    x = BatchNormalization()(x)

    # Output layer
    output = Dense(units=4, activation='softmax')(x)

    model = Model(inputs=input_layer, outputs=output)
    opt = Adam(learning_rate=1e-3)
    model.compile(loss='categorical_crossentropy', optimizer=opt, metrics=["accuracy"])

    return model

# Create the model
model = Combined_model()

# Print the model summary
model.summary()

checkpoint = [ModelCheckpoint(filepath='best_model.h5',
monitor='val_accuracy',mode='max',verbose=1,save_best_only=True,save_weights_only=True),
            LearningRateScheduler(lambda epoch : INIT_LR * pow(decay_rate, floor(epoch / decay_step)))]
earlystop = EarlyStopping(monitor='accuracy', min_delta=0, patience=15, verbose=1, mode='max')

history = model.fit(train_dataset_augmented,
                    epochs=50,
                    steps_per_epoch=len(train_dataset_augmented),
                    validation_data = test_data,
                    callbacks=[checkpoint])

model.load_weights('best_model.h5')
_, accuracy = model.evaluate(test_data)
print(f"Validation accuracy: {round(accuracy * 100, 2)}%")
```

```
pred_probs = model.predict(test_data, verbose=1)

pred_classes = pred_probs.argmax(axis=1)

y_labels = []
for images, labels in test_data.unbatch():
  y_labels.append(labels.numpy().argmax())

target_names = ['Glioma', 'Meningioma', 'No Tumor','Pituitary']
print(classification_report(y_labels,
                            pred_classes,
                            target_names=target_names, digits=4))

cm = confusion_matrix(y_labels,pred_classes)

TP = cm[0, 0]
TN = cm[1:, 1:].sum()
FP = cm[0, 1:].sum()
FN = cm[1:, 0].sum()

Population = TN+FN+TP+FP
spesificity = TN / (TN + FP)
sensitivity = TP / (TP + FN)

print("True Positives:", TP)
print("False Positives:", FP)
print("True Negatives:", TN)
print("False Negatives:", FN)
print("Specificity:", spesificity)
print("Sensitivity:", sensitivity)

def make_confusion_matrix(y_true, y_pred, classes=None, figsize=(20, 20), text_size=15, norm=False,
savefig=False):

  cm = confusion_matrix(y_true, y_pred)
  cm_norm = cm.astype("float") / cm.sum(axis=1)[:, np.newaxis]
  n_classes = cm.shape[0]

  fig, ax = plt.subplots(figsize=figsize)
```

```
  cax = ax.matshow(cm, cmap=plt.cm.Blues)
  fig.colorbar(cax)

  if classes:
    labels = classes
  else:
    labels = np.arange(cm.shape[0])

  ax.grid(False)

  ax.set(title="Confusion Matrix",
         xlabel="Predicted label",
         ylabel="True label",
         xticks=np.arange(n_classes),
         yticks=np.arange(n_classes),
         xticklabels=labels,
         yticklabels=labels)

  ax.xaxis.set_label_position("bottom")
  ax.xaxis.tick_bottom()

  plt.xticks(rotation=70, fontsize=text_size)
  plt.yticks(fontsize=text_size)

  threshold = (cm.max() + cm.min()) / 2.

  for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
    if norm:
      plt.text(j, i, f"{cm[i, j]} ({cm_norm[i, j]*100:.1f}%)",
               horizontalalignment="center",
               color="white" if cm[i, j] > threshold else "black",
               size=text_size)
    else:
      plt.text(j, i, f"{cm[i, j]}",
               horizontalalignment="center",
               color="white" if cm[i, j] > threshold else "black",
               size=text_size)

make_confusion_matrix(y_true=y_labels,
```

```
                    y_pred=pred_classes,
                    classes=target_names,
                    figsize=(5, 5),
                    text_size=12,
                    norm=False,
                    savefig=True)

loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(loss) + 1)

plt.figure(figsize=(5.5, 4))

plt.plot(epochs, loss, 'y', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and Validation Loss', fontsize=12)
plt.xlabel('Epochs', fontsize=10)
plt.ylabel('Loss', fontsize=10)
plt.legend(fontsize=10)

plt.xticks(fontsize=10)
plt.yticks(fontsize=10)

plt.grid(False)

plt.tight_layout()
plt.show()

loss = history.history['accuracy']
val_loss = history.history['val_accuracy']
epochs = range(1, len(loss) + 1)

plt.figure(figsize=(5.5, 4))

plt.plot(epochs, loss, 'y', label='Training acc')
plt.plot(epochs, val_loss, 'r', label='Validation acc')
plt.title('Training and Validation Loss', fontsize=12)
plt.xlabel('Epochs', fontsize=10)
plt.ylabel('Loss', fontsize=10)
```

```
plt.legend(fontsize=10)

plt.xticks(fontsize=10)
plt.yticks(fontsize=10)

plt.grid(False)

plt.tight_layout()
plt.show()
```

ÖZGEÇMİŞ

Adı Soyadı	: Gizem Akgönül
Doğum Yeri	:
Doğum Tarihi	:
e-posta adresi	:
Adresi	:

Eğitim Bilgileri

Lise	: Bahçelievler Anadolu Lisesi (İng)
Lisans	: Pamukkale Üniversitesi - Biyomedikal Mühendisliği
Yabancı Dil ve Düzeyi	Türkçe - Anadili İngilizce - İyi