Beyin Tümör Tespiti

Muhammet Enes İnal
Bilgisayar Mühendisliği
İstanbul Topkapı Üniversitesi
22040101023
muhammetenesinal@stu.topkapi.edu.tr

Muhammed Mert Oruç
Bilgisayar Mühendisliği
İstanbul Topkapı Üniversitesi
22040101035
muhammedmertoruc@stu.topkapi.edu.tr

Zekeriya Deniz Uğurlu
Bilgisayar Mühendisliği
İstanbul Topkapı Üniversitesi
22040101034
zekeriyadenizugurlu@stu.topkapi.edu.tr

Anahtar Kelimeler:beyin tümörü tespiti, MLP, sinir ağı, tıbbi görüntü sınıflandırma

I. ÖZET

Bu projede, beyin MR görüntülerinden tümör tespiti yapmak amacıyla bir sınıflandırma modeli geliştirilmiştir. Model, MLP kullanılarak eğitilmiş ve "tumor" ve "notumor" sınıfları arasında ayrım yapmıştır. Sonuçlar, modelin yüksek doğruluk oranları sağladığını göstermektedir. Bu çalışma, beyin tümörlerinin erken teşhisine yardımcı olacak yapay zeka tabanlı bir çözüm sunmayı hedeflemektedir.

II. GİRİŞ

Beyin tümörleri, ciddi sağlık sorunlarına yol açabilen hastalıklardır. Erken teşhis, tedavi sürecini belirleyen en önemli faktörlerden biridir. Bu projede, beyin MR görüntülerini kullanarak tümör tespiti yapan bir MLP tabanlı sınıflandırma modeli geliştirdik. MLP, derin öğrenme yöntemlerine kıyasla daha basit ve hızlı bir yapı sunarak temel sınıflandırma görevlerinde yüksek doğruluk elde etmemizi sağladı. Proje kapsamında, tıbbi görüntülerin analizini kolaylaştıracak yapay zeka tabanlı bir çözüm sunduk ve beyin tümörü tespitinde yeni bir yaklaşım geliştirdik.

III. YÖNTEMLER

A) Veri Toplama ve Hazırlık

Bu projede kullanılan veri seti, Kaggle platformundan alınmıştır. Veri seti, "Training" ve "Testing" olmak üzere iki ana klasöre ayrılmıştır. Her iki klasörde de "tumor" ve "no tumor" adında iki ayrı alt klasör bulunmaktadır. Veri setinde toplamda yaklaşık 3000 görsel yer almaktadır. Görsellerin boyutları 224x224 piksel olarak belirlenmiş ve modelde net sonuçlar elde edebilmek için bu boyutlara çevrilmiştir.

B) Veri Ön İşleme

Tumor" ve "No Tumor" sınıfları, 'LabelEncoder' kullanılarak sayısal değerlere dönüştürülmüştür. Bu işlem sonrasında her bir sınıf, 'one-hot encoding' yöntemiyle kategorik verilere dönüştürülmüştür.

Veriler, 'sklearn.utils.shuffle' fonksiyonu ile karıştırılmıştır. Bu, modelin daha genellenebilir olmasını sağlar.

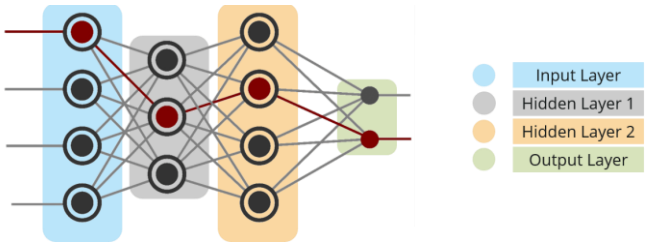
Veriler, eğitim ve test setlerine %80-%20 oranında bölünmüştür. Eğitim verisi 2441 örnek, test verisi ise 611 örnekten oluşmaktadır.

Görsellerin boyutları, modelin daha doğru sonuçlar verebilmesi için 224x224 boyutlarına sabitlenmiştir.

- **Eğitim Seti (X_train):** (2441, 224, 224, 3)
- **Test Seti (X_test):** (611, 224, 224, 3)
- **Eğitim Etiketleri (Y_train):** (2441, 2)
- **Test Etiketleri (Y_test):** (611, 2)

C) Modelin Tanımlanması

MLP MODELİ:

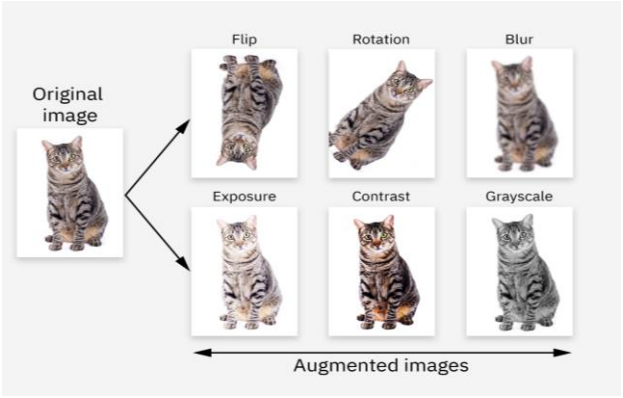


[1]

Model, 'Sequential' API kullanılarak tanımlanmıştır ve aşağıdaki katmanlardan oluşmaktadır.

Data Augmentation:

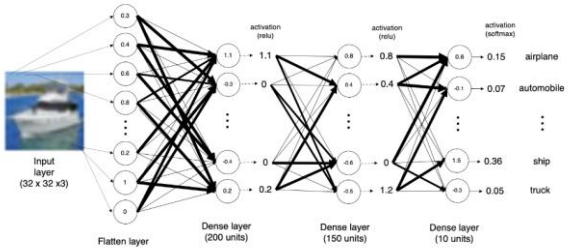
Data Augmentation, modelin genelleme yeteneğini artırmak için eğitim verisini çeşitlendirme işlemidir. Görüntülere rastgele yatay çevrim, döndürme ve yakınlaştırma işlemleri uygular. Bu sayede, modelin farklı veri varyasyonları üzerinde eğitim yaparak daha sağlam ve esnek bir öğrenme sağlanır.



[2]

Flatten Katmanı:

Flatten katmanı, 3D giriş verisini tek boyutlu bir vektöre dönüştürür. Kodumuzda da kullanıldığı gibi , (224 , 224 , 3) boyutundaki bir görüntü, 150,528 elemandan oluşan tek bir vektöre çevrilir. Bu dönüşüm, ardından gelen Dense katmanları için gereklidir çünkü dense katmanlar her bir elemanla tüm diğerleri arasında bağlantılar kurarak sınıflandırma yapar. Yani, Flatten katmanı, görsel özelliklerin birleşiminden modelin karar vermesini sağlar.



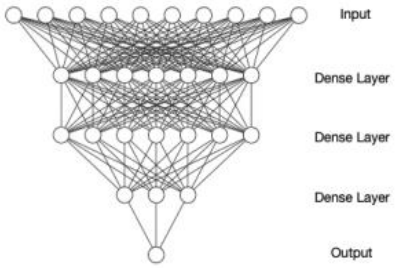
[3]

Dense, BatchNormalization ve Dropout Katmanları:

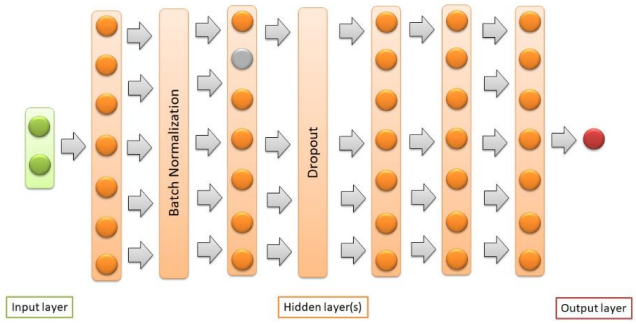
Modeldeki Dense katmanları, ağın öğrenme kapasitesini artırmak için sırasıyla 512, 256, 128, 64 ve 2 nöron sayısına sahip olarak yapılandırılmıştır. Bu yapı, her bir katmanın daha az nöronla daha derinlemesine bir öğrenme yapmasını sağlar. İlk katmanda 512 nöron kullanılması, modelin geniş bir özellik kümesi öğrenebilmesi için yeterli kapasite sunar. Ardından, her katmanda nöron sayısının kademeli olarak azaltılması, modelin daha genel öğrenmesini sağlar. En son olan Dense katmanında 2 nöronun olması sebebi iki adet sınıfa sahip olmasıdır. Modelin karmaşıklığını kontrol altında tutarak aşırı uyumu (overfitting) engellemeye yardımcı olur.

Her Dense katmanından sonra Batch Normalization kullanılması, modelin öğrenme sürecini daha stabil hale getirir ve eğitim hızını artırır. Ayrıca varyans ve standart sapmayı dengede tutar.

Dropout katmanı ise modelin aşırı öğrenmesini engellemek amacıyla, her katmanda %30 oranında rastgele nöronları devre dışı bırakır. Son olarak, çıktıyı iki sınıf arasında('tumor' ve 'notumor') ayıran Softmax katmanı, modelin her sınıf için olasılık tahmini yaparak, en yüksek olasılığa sahip olan sınıfı seçmesini sağlar.



[4]



[5]

D) Modelin Eğitilmesi

Modelin Eğitilmesi: Eğitim verileri (x_train ve y_train) kullanılarak modelin eğitimine başlanır.

.

epochs=30: Modelin eğitim veri seti üzerinde 30 kez (epoch) geçmesi yapılır.

batch_size=32: Eğitim sırasında veriler küçük gruplara (batch) ayrılarak modele sunulur.

verbose=1: Eğitim süreci sırasında, modelin ilerleyişi hakkında bilgi almak amacıyla ayarlanır.

callbacks=[checkpoint, reduce_lr], doğrulama kaybı en düşük olduğunda model ağırlıklarını kaydeder ve öğrenme oranını iyileştirme amacıyla düşürür.

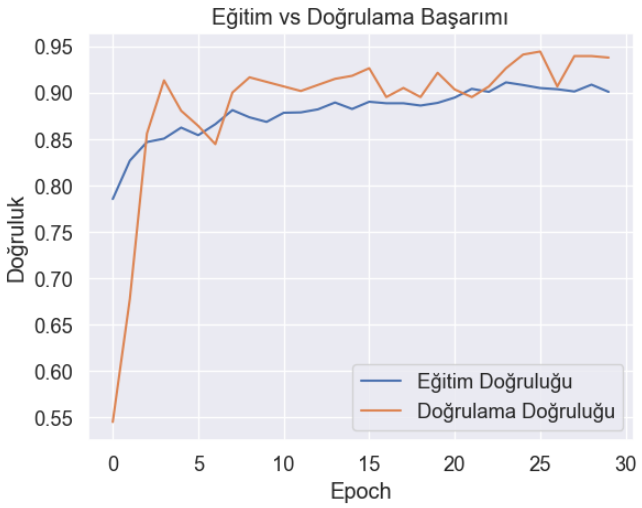
E) Modelin Kaydedilmesi

Modelin yapısı, 'model.to_json()' ile JSON formatında kaydedilirken, ağırlıkları 'model.save_weights("model.weights.h5")' ile HDF5 formatında saklanır. Bu sayede, modelin hem yapısı hem de öğrenilen parametreleri dışa aktarılır ve yeniden kullanılabilir.

F) Modelin Değerlendirilmesi

Modelin değerlendirilmesi, test verisi üzerinde doğruluk (accuracy) ve kayıp (loss) gibi metriklerle yapılır. Doğruluk, modelin doğru tahmin oranını gösterirken, kayıp fonksiyonu modelin tahmin hatasını belirtir. Eğitim sırasında yüksek doğruluk ve düşük kayıp, modelin başarılı olduğunu gösterir. Test verisi kullanılarak yapılan değerlendirme, modelin genelleme yeteneğini ölçer; aşırı uyum (overfitting) durumunda modelin eğitim verisine fazla odaklandığı anlaşılır.

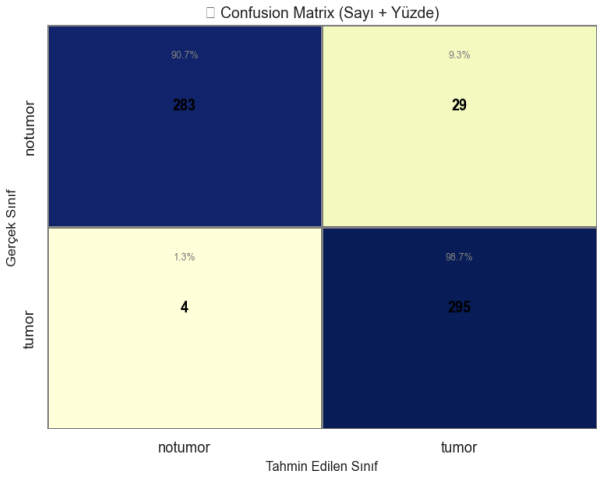
G) Modelde Eğitim Sonuçlarının Görselleştirilmesi



Grafik1:eğitim ve doğrulama doğruluğu

Grafikte, mavi çizgiler eğitim doğruğunu ve turuncu çizgiler ise doğrulama doğruluğunu temsil etmektedir. X ekseni Epoch'u (Eğitim döngü sayısı), Y ekseni doğruluk (accuracy) değerini temsil eder.

1. **Yüksek Doğruluk**: Hem eğitim doğruluğu hem de doğrulama doğruluğu zamanla artarak %90'ın üzerine çıkmış. Son epoch'larda doğrulama doğruluğu %94-95 seviyelerine kadar ulaşmış. Bu, modelin verilere oldukça iyi uyum sağladığını gösteriyor.
2. **Aşırı Uyum (Overfitting)**: Doğrulama doğruluğunda küçük dalgalanmalar var ama çok belirgin bir düşüş veya kopma görülmüyor. Bu durum aşırı uyum riskinin çok düşük olduğunu gösteriyor.
3. **Dengeli Performans**: Eğitim ve doğrulama doğruluğunun benzer seviyelerde seyretmesi modelin dengeli ve tutarlı çalıştığını gösterir.



Grafik2: Confusion Matrix

Grafikte yatay eksen gerçek sınıfları temsil ederken dikey eksen ise tahmin edilen sınıfları temsil eder. Sınıflar ise 'notumor' yani tümörün olmadığı durum ve 'tumor' yani tümörün olduğu durum olarak ikiye ayrılır.

1. **Yüzdelik Dağılım**:

- Gerçekte tümör olmayanlardan %90.7'si doğru şekilde '*notumor*' olarak tahmin edilmiş.
- Gerçekte tümör olmayanların %9.3'ü yanlışlıkla '*tumor'* olarak tahmin edilmiş.
- Gerçekte tümör olanların %98.7'si doğru şekilde '*tumor*' olarak tahmin edilmiş.
- Gerçekte tümör olanların %1.3'ü yanlışlıkla '*notumor'* olarak tahmin edilmiş.

2. **Modelin Güçlü Yönleri**: '*Tumor'* sınıfını tespit etme konusunda çok başarılıdır. Gerçek hasta olup sağlıklı tahmini oldukça düşük olması, sağlık alanında önemli ve başarılı bir sonuçtur.
3. **İyileştirilmesi Gereken Noktalar**: Yanlış tahmin sayısının biraz yüksek olması sıkıntı yaratabilir. Bu durum, bazı sağlıklı bireylerin hatalı şekilde hasta olarak değerlendirilmesine sebep olabilir.

IV. SONUÇ:

Bu projede, beyin tümörlerini tespit etmek için bir derin öğrenme modeli oluşturduk. Model, beyin tümörü ve tümörsüz görüntüleri ayırt etmek için eğitildi. Eğitimde kullandığımız veriler, modelin görüntüdeki önemli özellikleri öğrenmesini sağladı. Veri artırma (data augmentation) teknikleri, modelin daha iyi genelleme yapmasını sağladı ve aşırı uyum (overfitting) olasılığını azalttı.

Modeli eğitirken, doğru hiperparametreler ve optimizasyon algoritmaları kullandık. Adam optimizasyonu sayesinde modelin öğrenme süreci hızlandı ve stabil hale geldi. Eğitim sürecinde elde ettiğimiz doğruluk oranı, modelin doğru sonuçlar verdiğini gösteriyor, ancak yine de bu oranı daha da artırmak mümkün.

Sonuç olarak, bu model, beyin tümörü tespiti gibi önemli bir alanda kullanılabilecek bir temel oluşturuyor. Gelecekte, daha fazla veri eklenerek ve farklı yöntemler kullanılarak doğruluk oranı artırılabilir.

V. REFERENCES

[1] Masoudnickparvar, "Brain Tumor MRI Dataset", Kaggle, 2021. [Available online at: https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset].

[2] Chollet, F. (2015). Keras: The Python Deep Learning library. GitHub. Retrieved from https://github.com/keras-team/keras

[3] Çetin, A., & Yılmaz, H. (2020). *Derin Öğrenme: Temelleri ve Uygulamaları*. İstanbul: Papatya Yayıncılık.

[4] "Yinelenen Sinir Ağları Nedir?" *Datakapital Blog*, [Online]. Available: https://datakapital.com/blog/yinelenen-sinir-aglari-nedir. [Accessed: Jan. 16, 2025]. "

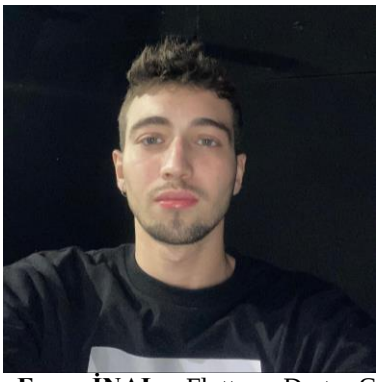
[5] "Recurrent Neural Networks," *GeeksforGeeks*, [Online]. Available: https://media.geeksforgeeks.org/wp-content/uploads/20231204131544/recurrent_neural_networks-768.png. [Accessed: Jan. 16, 2025]."

[6] O'Shea, T., & Nash, A. (2015). *Convolutional Neural Networks: A Brief Review of Architectures and Applications*. International Journal of Computer Science, 10(4), 23-29.

[7] "https://datascience.stackexchange.com/questions/103183/working-of-dense-layer. [Accessed: Jan. 16, 2025]. "

[8] Özdemir, M., & Yıldız, A. (2017). *Yapay Zeka ve Derin Öğrenme Uygulamaları*. İstanbul: Nobel Akademik Yayıncılık.

PROJEDE EMEĞİ GEÇENLER



Muhammet Enes İNAL, Flutter, Dart, C, C#, Java, Javascript, Python, PHP dilleri üstünde çalışmıştır. Önemli projeleri arasında C# ile hastane uygulaması, Html-CSS-PHP ile otel web sitesi yer alır.

Zekeriya Deniz UĞURLU; Flutter, Dart, C, C++, C#, React, Html-CSS-PHP , Java, Python dilleri üzerinde yoğunlaşmış ve kendini geliştirmiştir. Önemli projeleri arasında C# ile sürücü kursu kayıt uygulaması, Flutter ile Spor takvim oluşturma mobil uygulaması ve HTML-CSS ile sürücü kursu web sitesi projeleri yer alır.

Muhammed Mert ORUÇ, C, Flutter, Dart, C++, C#,Java, HTML, CSS, PHP ve Python dillerinde kendini geliştirmiştir. C# ile otopark uygulaması, Html-CSS-PHP ile bankacılık web sitesi yer alır.