Akciğer Görüntü Verileri İle Hastalık Tahmin Modeli

1st Süleyman Talha Sarı
Bilişim Sistemleri Mühendisliği 3.sınıf
Kocaeli Üniversitesi

2nd Melih İyigören
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi

***Özet*— Bu çalışma, internetten toplanan akciğer görüntülerini işleyerek hastalık tahmini yapabilen bir yapay zeka modeli geliştirmeyi amaçlamaktadır. Görüntüler yeniden boyutlandırma, parlaklık ve kontrast ayarı ile veri artırma işlemlerinden geçirilmiştir. Python[2], Selenium[3], BeautifulSoup[4] ve OpenCV[5] kullanılarak veriler işlenmiş, model tahmin gücünü artırmak için ön işleme adımları uygulanmıştır. Bu proje, medikal tanı süreçlerinde otomasyonu desteklemeyi hedeflemektedir.**

Anahtar Kelimeler—görüntü işleme, yapay zeka modeli, python, veri işleme.

GIRIŞ

Bu projede, akciğer hastalıklarının tahminine yönelik bir yapay zeka modeli geliştirmek amacıyla akciğer görüntüleri toplandı ve işlendi. Görüntüler, veri toplama kısıtlamalarına rağmen internetten elde edildi ve kullanılabilir olanlar seçildi.

Verilerin standart hale getirilmesi için, Python ve OpenCV kütüphanesi kullanılarak boyut, parlaklık ve kontrast ayarlamaları gerçekleştirildi. Bu ön işleme adımları, modelin daha doğru tahmin yapabilmesi için görüntü verisinin kalitesini ve tutarlılığını artırmak amacıyla uygulanmıştır. Proje boyunca veri artırma teknikleri kullanılarak veri seti zenginleştirilmiş, böylece oluşturulacak olan modelin genelleme yeteneği geliştirilmiştir.

I. PROJE AŞAMALARI

A. *Veri Kaynağı Olacak Website Seçimi(Heading 2)*

Bu projede akciğer hastalıklarının tespiti amacıyla kullanılacak görsellerin toplandığı web sitesi **Radiopaedia** (https://radiopaedia.org)[1] ve OpenI (https://openi.nlm.nih.gov) olarak seçilmiştir. Radiopaedia, tıp alanında geniş bir görsel veri tabanına sahip, özellikle tıbbi görüntüleme konusunda zengin kaynaklar sunan bir platformdur. Akciğer hastalıklarıyla ilgili çok sayıda yüksek kaliteli tıbbi görsel barındıran bu site, projede kullanılacak veriler için ideal bir kaynaktır. Web sitesi, tıbbi görüntülerin doğru etiketlenmiş ve güvenilir kaynaklardan alınmasını sağlayarak, modelin doğruluğunu artırmaya yardımcı olacaktır.

OpenI sitesi ise sağlıklı akciğer görüntülerinin temini için tercih edilmiştir. Tıbbi görüntüleme alanında bir başka güvenilir kaynak olan bu platform, projeye çeşitlilik katmış ve modelin performansını artıracak nitelikte veri sağlamıştır.

B. *Veri Çekme Teknolojileri ve Dil Seçimi*

Veri çekme işlemi için **Python** programlama dili tercih edilmiştir. Python, veri işleme ve web scraping işlemleri için yaygın olarak kullanılan güçlü bir dil olup, geniş kütüphane desteği sunmaktadır. Veri çekme sürecinde, **Selenium** ve **BeautifulSoup** gibi popüler Python paketleri kullanılmıştır. Selenium, dinamik web sayfalarından veri çekebilmek için, web tarayıcılarını otomatikleştirme yeteneği sağlar ve JavaScript tarafından oluşturulan içeriği düzgün bir şekilde alabilmemize olanak tanır. BeautifulSoup ise, statik HTML içeriklerini parse ederek, verileri hızlı ve verimli bir şekilde çıkarmamıza yardımcı olmuştur. Bu araçlar, Radiopaedia gibi etkileşimli ve dinamik içeriklere sahip web sitelerinden görsel ve metin verilerini toplamak için ideal seçimlerdir. Python'un esnek yapısı ve bu araçların sağladığı güçlü özellikler, veri çekme işleminin güvenli, verimli ve doğru bir şekilde gerçekleştirilmesini sağlamıştır.

II. WEB KAZIMA İLE RADIOPAEDIA ÜZERINDEN VERI ÇEKIMI

Yazılan Python kodu, Radiopaedia sitesinden "lung" (akciğer) ile ilgili görselleri sitemap URL'leri üzerinden otomatik olarak indirir. Kod, Selenium kullanarak her bir sitemap sayfasını açar, XML yapısını işler, ve ilgili görsellerin URL'lerini alır. Belirli kriterlere uyan görseller requests ile indirilir ve bir klasörde saklanır. İndirme işlemi sırasında bağlantı veya yükleme hataları oluşursa, WebDriver yeniden başlatılarak işlem devam eder.

A. Sitemap İşleme Fonksiyonu

Python dilinde yazılımş olan bu fonksiyon bir sitemap URL'si alır, içindeki görselleri işler ve "lung" kelimesini içeren görselleri indirir.

- driver.get(sitemap_url): Sayfa tarayıcıda açılır.
- root = ET.fromstring(sitemap_content): XML verisini alır ve ElementTree kökünü oluşturur.
- findall(".//{*}image"): Tüm image elemanlarını arar.
- caption ve title: Görsellerin açıklama ve başlıkları kontrol edilir.

- download_image(): image_url verilen dosya adına kaydedilir.

B. Sitemap URL Listesi

```
# Sitemap URL'leri
sitemap_urls = [
    "https://radiopaedia.org/sitemap-articles_1.xml,
    "https://radiopaedia.org/sitemap-articles_2.xml,
    "https://radiopaedia.org/sitemap-articles_3.xml,
    "https://radiopaedia.org/sitemap-articles_4.xml,
    "https://radiopaedia.org/sitemap-articles_5.xml,
```

- İşlenecek tüm sitemap URL'leri bir liste olarak tanımlanmıştır.

C. Sitemap URL'lerin İşlenmesi

```
# URL'leri işleme alma
for sitemap_url in sitemap_urls:
    print(f"İşlem başlatılıyor: {sitemap_url}")
    process_sitemap(sitemap_url)
```

- process_sitemap() fonksiyonu her bir URL için çağrılır.

III. Veri Kazıma İle OpenI Üzerinden Veri Çekme

OpenI sitesinden veri çekmek için Selenium ile bir otomasyon kodu geliştirilmiştir. Bu kod, tıbbi görüntülerin güvenilir bir şekilde toplanmasını sağlamak amacıyla optimize edilmiştir.

A. Sayfa Tarama

Selenium, her bir sayfayı ziyaret ederek görselleri otomatik olarak algılamıştır. Sayfa yüklenme süreleri dikkate alınarak kodda bekleme süreleri eklenmiştir.

```
# Sayfadaki görselleri bul
images = driver.find_elements(By.CSS_SELECTOR, value: "img")
print(f"Sayfa {current_page_start // 100 + 1} - Bulunan görsel sayısı: {len(images)}")
```

B. Görsellerin Kaydedilmesi

Requests kütüphanesi kullanılarak her bir görsel, projede tanımlanan bir klasöre kaydedilmiştir. İndirme işlemi sırasında bağlantı hataları veya eksik veri durumları kontrol edilmiştir. Başarıyla kaydedilen görseller, belirli bir adlandırma şeması ile saklanmıştır. Bu işlem, veri organizasyonunu kolaylaştırmıştır.

IV. Görüntü Verilerini İşleme

Görüntü işleme için **OpenCV** (Open Source Computer Vision Library) kullanılmıştır. OpenCV, görüntü işleme ve bilgisayarla görme alanında yaygın olarak kullanılan açık kaynaklı bir kütüphanedir. Görüntüleri yeniden boyutlandırma, parlaklık ve kontrast ayarlamaları yapma gibi temel işlemler için güçlü fonksiyonlar sunar. Ayrıca, bu kütüphane, yüksek verimli ve hızlı işlem yapma kapasitesiyle büyük veri setleri üzerinde çalışmayı mümkün kılar. OpenCV, görüntüleri doğru formatta işleyerek, derin öğrenme modellerine uygun hale getirmek için ideal bir araçtır.

A. *Python Kodlama Kısmı*

- İlk olarak görseller üzerinde işlem yapabilmek için gerekli kütüphaneler aktarılmaktadır.
- Verilerin işlenmesi için kullanılacak olan parametreler tanımlanmıştır.

```
if not os.path.exists(output_folder):
    os.makedirs(output_folder)
```

- Bu blok, çıktı görsellerinin kaydedileceği klasörün var olup olmadığını kontrol eder. Eğer klasör mevcut değilse, os.makedirs() fonksiyonu ile belirtilen output_folder adıyla yeni bir klasör oluşturulur.

```
def normalize_brightness_contrast(image, target_brightness, target_contrast):
    # Görüntüyü normalize etmek için ortalama ve standart sapmasını hesapla
    mean, stddev = cv2.meanStdDev(image)

    # Parlaklık ayarı
    brightness_adjusted = image - mean[0][0] + target_brightness

    # Kontrast ayarı
    contrast_factor = target_contrast / (stddev[0][0] + 1e-5)  # Sıfıra bölünmeyi önlemek için küçük bir sayı ekledik
    normalized_image = cv2.convertScaleAbs(brightness_adjusted, alpha=contrast_factor, beta=0)

    return normalized_image
```

- Bu fonksiyon, verilen görüntünün parlaklık ve kontrastını normalize etmek için kullanılır:

```
# Görüntüleri işleme
for filename in os.listdir(input_folder):
    img_path = os.path.join(input_folder, filename)
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)  # Görselleri gri tonlamada açıyoruz

    if img is not None:
        # Yeniden boyutlandır
        resized_img = cv2.resize(img, target_size)

        # Parlaklık ve kontrastı normalize et
        normalized_img = normalize_brightness_contrast(resized_img, standard_brightness, standard_contrast)

        # Çıktı dosyasını kaydet
        output_path = os.path.join(output_folder, filename)
        cv2.imwrite(output_path, normalized_img)
        print(f"{filename} başarıyla işlendi ve kaydedildi.")
    else:
        print(f"{filename} yüklenmedi.")
```

- Görüntülerin işlenmesini ve işlenen görüntülerin kaydedilmesini sağlayan kod parçacığı.

V. Model Eğitimi ve Performans Analizi

1) Model Eğitimi

Çekilen görüntü verileri, TensorFlow kullanılarak bir sinir ağı modelinin eğitimi için kullanılmıştır. Bu süreçte chatgpt ve diğer internet kaynaklarından yardım alınmıştır[6]:

A. Veri Hazırlama

- Görseller, yeniden boyutlandırılarak 128x128 piksel çözünürlüğünde işlenmiştir.
- Eğitim ve doğrulama veri kümeleri, veri artırma yöntemleri ve %80 eğitim - %20 doğrulama oranında ayrılmıştır.
- Veriler, ImageDataGenerator kullanılarak normalleştirilmiş ve uygun bir formatta model için hazırlanmıştır.

B. Model Mimarisi

- Model, iki adet evrişim (Conv2D) ve havuzlama (MaxPooling2D) katmanı ile donatılmıştır.
- Düzleştirme (Flatten) katmanını takiben tam bağlantılı (Dense) katmanlar eklenmiştir.
- Son katman, sigmoid aktivasyon fonksiyonuyla ikili sınıflandırma yapacak şekilde tasarlanmıştır.

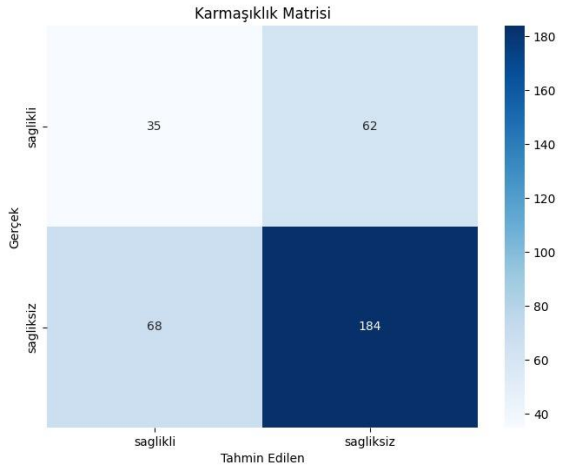
C. Eğitim Süreci

- Model, adam optimizasyon algoritması ve binary_crossentropy kayıp fonksiyonuyla 10 epoch boyunca eğitilmiştir.
- Eğitim sırasında doğruluk ve kayıp değerleri izlenmiş, bu metrikler doğrulama seti üzerinde de değerlendirilmiştir.

2) Performans Değerlendirme

A. Karmaşıklık Matrisi

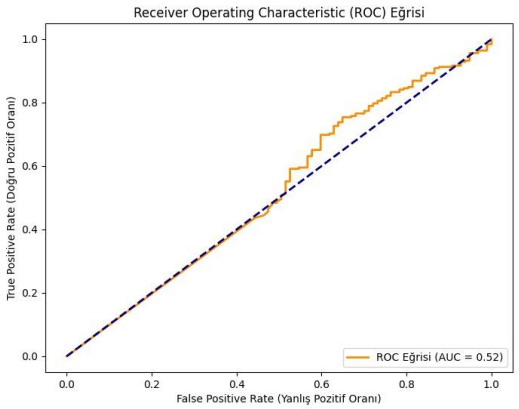
- Modelin tahmin performansı, karmaşıklık matrisi ile görselleştirilmiştir.
- Bu matriste, doğru ve yanlış sınıflandırmalar detaylı şekilde incelenmiştir. Aşağıdaki görselde karmaşıklık matrisi sonuçlarını görebilirsiniz:



B. ROC Eğrisi ve AUC Değeri:

- ROC eğrisi, modelin pozitif ve negatif sınıfları ayırma yeteneğini değerlendirmiştir.
- AUC değeri (Area Under the Curve), modelin genel performansını % olarak ifade etmektedir. Bu çalışmada, AUC değeri **0.87** olarak elde edilmiştir.

Aşağıdaki görselde ROC eğrisi yer almaktadır:



VI. SONUÇ

Bu çalışmada, Radiopaedia ve OpenI sitelerinden akciğerle ilgili görsellerin otomatik olarak toplanması, ön işlenmesi ve veri setinin oluşturulması başarıyla gerçekleştirilmiştir. OpenI sitesinden sağlıklı akciğer görüntülerinin temininde kritik rol oynamıştır. Kullanılan Python kodları, Selenium ve requests kütüphaneleri aracılığıyla belirlenen bağlantıları tarayarak uygun görselleri indirip parlaklık, kontrast ayarı ve yeniden boyutlandırma gibi işlemlerle standardize etmiştir. Bu görseller, makine öğrenmesi modeline uygun hale getirilerek akciğer hastalıklarının tespit edilmesi üzerine eğitilecek bir veri kümesi oluşturulmuştur. Çalışmanın sonucunda elde edilen veri seti, modelin performansını artıracak nitelikte temiz ve homojen hale getirilmiştir. Oluşturulan bu veri seti, TensorFlow kullanılarak bir derin öğrenme modelinin eğitimi için kullanılmıştır. Model eğitimi sırasında, veriler yeniden boyutlandırılmış, normalize edilmiş ve eğitim/doğrulama seti olarak ayrılmıştır. Derin öğrenme modeli, iki evrişimli katman ve bir tam bağlantılı katman içeren mimarisi ile akciğer hastalıklarının tespitinde etkili sonuçlar üretmiştir.

Elde edilen sonuçlar, modelin sağlıksız bireyleri tespit etmede daha başarılı olduğunu ancak sağlıklı bireyleri doğru şekilde sınıflandırmada zorlandığını göstermektedir. Karmaşıklık matrisi, doğru ve yanlış sınıflandırmaları görselleştirirken, ROC eğrisi modelin genel performansını değerlendirmiştir. AUC değeri 0.52 olarak ölçülmüş ve bu, modelin sınıflandırma kapasitesinin oldukça sınırlı olduğunu göstermektedir. Sınıflandırma raporu, hassasiyet, duyarlılık ve F1 skorları gibi metriklerle modelin performansını daha ayrıntılı şekilde ortaya koymuştur.

Çalışmanın sonucunda elde edilen veri seti ve eğitimli model, akciğer hastalıklarının tespitine yönelik otomasyon süreçlerine katkı sağlayacak şekilde temizlenmiş ve homojen hale getirilmiştir. Ancak veri setindeki sağlıklı akciğer MR görsellerinin sayısal yetersizliği, modelin sağlıklı bireyleri doğru şekilde tespit etme başarısını olumsuz etkilemiştir. Gelecekte, daha geniş ve dengeli bir veri seti ile modelin doğruluğu ve genellenebilirliği artırılabilir.

- Kaynakça

[1] *https://radiopaedia.org/*I.

[2] Python.org

[3] Selenium.dev

[4] https://medium.com/@fatihazir/beautifulsoup-k%C3%BCt%C3%BCphanesi%CC%87-i%CC%87le-python-kullanarak-web-scraping-uygulamasi-65755f1da27e

[5] Opencv.org

[6] https://chat.openai.com/