T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ

1.ÖDEV RAPORU

B211200009
İrem KABAOĞLU

ISE 456 Bilgisayar Görmesine Giriş
Dr.Öğr.Üyesi Fatma AKALIN

Eye Disease Detection Veri Kümesinin ResNet50 Modeli ile Eğitilmesi

Ödevin temel hedefi, görüntü işleme teknikleri kullanarak göz hastalıklarının tespitine odaklanan bir sınıflandırma modeli geliştirmektir. Bu amaç doğrultusunda, çeşitli görüntü işleme teknikleri uygulanarak, görsellerin daha doğru ve verimli bir şekilde sınıflandırılması sağlanmıştır. Ödevin bir diğer önemli bileşeni, özgün bir kenar tespit filtresinin geliştirilmesi ve bu filtre ile kenarların belirginleştirilmesidir. Ayrıca, transfer learning yöntemi ile modelin sınıflandırma başarısı artırılmıştır. Görüntü işleme teknikleri, renk uzayı dönüşümleri, piksel düzeyindeki manipülasyonlar, kenar tespiti, morfolojik işlemler, bitwise işlemleri ve eşikleme gibi adımlarla zenginleştirilmiş ve modelin genel performansı iyileştirilmiştir.

Görüntü işleme süreci aşamalar halinde aşağıda anlatılmıştır.

1-) İlk olarak Eye Disease Detection Dataset isimli veri kümesini indirip Train isimli klasörün içerisine girip NORMAL, DRUSEN, DME, ve CNV kategorileri için rastgele 200'er görüntüyü Python kullanarak bir klasöre kopyalayıp kendi data setimi oluşturdum. Aşağıda, veri setinin oluşturulmasına ilişkin kod parçacıklarımın ekran görüntülerini bulabilirsiniz.

In [4]:
```python
import os
import random
import shutil

# Kaynak ve hedef klasörler
source_dir = r'C:\Users\USER\Desktop\train\train'  # Orijinal veri setinin olduğu klasör
output_dir = r'C:\Users\USER\Desktop\ödev'  # Seçilen görüntülerin kaydedileceği klasör

# Kategoriler
categories = ["NORMAL", "DRUSEN", "DME", "CNV"]
image_count = 200  # Her kategoriden seçilecek görüntü sayısı


os.makedirs(output_dir, exist_ok=True)
for category in categories:
    os.makedirs(os.path.join(output_dir, category), exist_ok=True)

# Görüntüleri rastgele seçip kopyalama işlemi
for category in categories:
    source_category_path = os.path.join(source_dir, category)
    output_category_path = os.path.join(output_dir, category)

    # Kategori klasöründeki tüm görüntüleri listele
    all_images = [img for img in os.listdir(source_category_path) if img.endswith((".jpg", ".png", ".jpeg"))]

    # Eğer klasörde 200'den az görüntü varsa hata vermemesi için kontrol
    selected_images = random.sample(all_images, min(image_count, len(all_images)))

    # Seçilen görüntüleri yeni klasöre kopyala
    for img in selected_images:
        shutil.copy(os.path.join(source_category_path, img), os.path.join(output_category_path, img))

    print(f"{category} kategorisinden {len(selected_images)} görüntü seçildi ve kopyalandı.")

print("✅ İşlem tamamlandı! Seçilen görüntüler 'selected_dataset' klasörüne kaydedildi.")
```

```
NORMAL kategorisinden 200 görüntü seçildi ve kopyalandı.
DRUSEN kategorisinden 200 görüntü seçildi ve kopyalandı.
DME kategorisinden 200 görüntü seçildi ve kopyalandı.
CNV kategorisinden 200 görüntü seçildi ve kopyalandı.
✅ İşlem tamamlandı! Seçilen görüntüler 'selected_dataset' klasörüne kaydedildi.
```

In [ ]:

2-) Bu aşamada, geliştirdiğim özgün görüntü işleme hiyerarşisi, NORMAL, DRUSEN, DME ve CNV olmak üzere dört sınıfa ait tüm görüntülere uygulanmıştır. Her bir görüntü, aşağıdaki sıralı adımlar doğrultusunda işlenmiştir:

- Grayscale dönüşümü ve CLAHE ile kontrast artırımı
- Gaussian Blur uygulaması
- Thresholding (eşikleme) işlemi
- Bitwise AND işlemi ile maskeleme
- Özel kenar tespit filtresi
- Morfolojik Closing işlemi

Her bir kategori için işlenmiş görüntüler ayrı klasörlerde saklanmıştır. Bu sayede, transfer learning ile sınıflandırma yapılmadan önce işlenmiş veri seti hazır hale getirilmiştir.

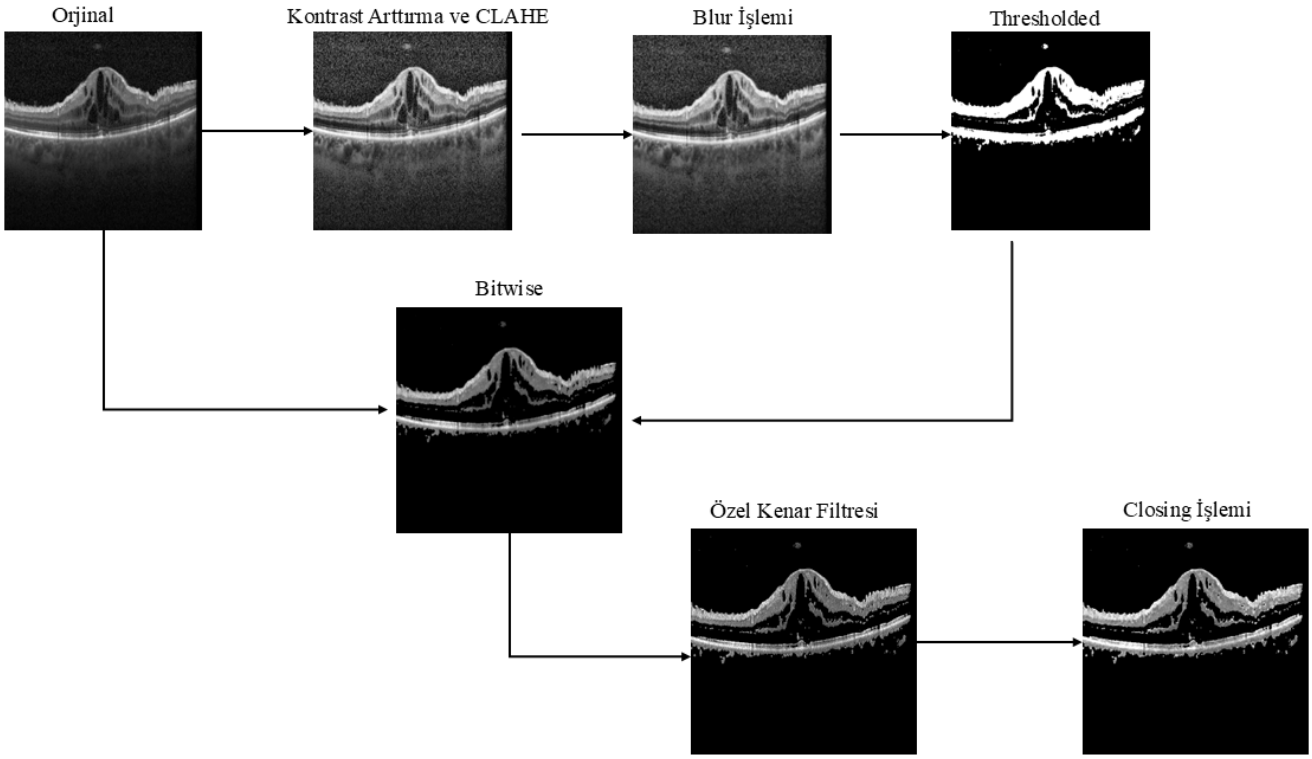
- **Kenar Tespiti: Dalgaya Dayanıklı Özel Filtre (Benim Filtrem)**

Ödev kapsamında, görselden önemli yapısal bilgileri çıkarmak için bir özel kenar tespit filtresi geliştirdim.

```
[32]: custom_filter = np.array([
          [ 0, -1,  0],
          [-1,  4, -1],
          [ 0,  0,  0]
      ], dtype=np.float32)
```

Merkezdeki '4' değeri, çevresindeki piksellerin toplam değerini azaltarak yüksek kontrastlı alanları (kenarları) öne çıkarır.-1 değerleri, kenar bilgilerini çeker; böylece merkez piksel ile çevresi arasında fark varsa bu fark vurgulanır. Bu filtre, özellikle yatay desenleri bastırır, dikey kenarları vurgular. Bu, retina yapısındaki bazı dikey damarları veya yapısal detayları daha belirgin hale getirmede etkili olabilir.

Aşağıda, görsel üzerinde gerçekleştirilen tüm işlemlerin akış diyagramını görebilirsiniz.



3-)Görüntü işleme işlemlerini gerçekleştirdikten sonra Python kodu ile, orijinal dataset klasöründeki her sınıfı %80 eğitim ve %20 doğrulama verisi olacak şekilde train/ ve val/ klasörlerine ayırdım. Bu işlem, transfer learning için veri setini hazırlamanın önemli bir adımıydı. Eğitim ve doğrulama verilerini ayrı klasörlere bölerek, modelin eğitimi sırasında doğru bir şekilde doğrulama yapabilmesini sağladım. İşlenmiş verilerin bulunduğu klasöründeki her sınıfı %80 eğitim ve %20 doğrulama verisi olacak şekilde train/ ve val/ klasörlerine ayırdım.

4-) Transfer Learning yöntemini kullanarak ResNet50 modeli ile çalışmalarımı gerçekleştirdim. Her iki veri kümesini de bu model ile eğittim ve eğitim sürecine ait doğruluk ve kayıp değerlerini gösteren grafikler oluşturdum.

5-) Son olarak, elde edilen sonuçlar kullanılarak model performans karşılaştırması gerçekleştirilmiştir. Bu karşılaştırma, doğruluk (accuracy), duyarlılık (recall), özgüllük (specificity), kesinlik (precision) ve F1-skoru (F1 score) gibi ölçütler üzerinden yapılmış ve işlenmiş görüntüler ile orijinal görüntüler arasındaki performans farkları analiz edilmiştir.Analiz tablosunu aşağıda göreblirsiniz.

Ölçüt	Orijinal Veri	İşlenmiş Veri
Doğruluk (Accuracy)	0.34	0.39
Duyarlılık (Recall)	0.34	0.39
Özgüllük (Specificity)	0.779	0.798
Kesinlik (Precision)	0.32	0.47
F1-Score	0.30	0.36

Histogram Grafiği ile Gösterim