İçindekiler Tablosu

[Görüntü İşleme Ara Sınav Ödevi 1](#_Toc183642876)

[Veri Seti Hakkında Kısa Bilgi 1](#_Toc183642877)

[Gerekli Kütüphanelerin Yüklenmesi 1](#_Toc183642878)

[Verilerin Yüklemesi Adımı 1](#_Toc183642879)

[Verilerin İncelenmesi – Ön Analizi 3](#_Toc183642880)

[Veri Görselleştirme Adımı 5](#_Toc183642881)

[Verilerin/Görüntülerin İstatistiksel Olarak Hesaplanması 6](#_Toc183642882)

[Histogram Analizi 9](#_Toc183642883)

[Görüntü İyileştirme 10](#_Toc183642884)

[Kontrast Germe(Contrast Stretching) 11](#_Toc183642885)

[Histogram Eşitleme 11](#_Toc183642886)

[Gamma Düzeltme 11](#_Toc183642887)

[Tek Görüntü Üzerinde Detaylı İyileştirme İncelemesi 13](#_Toc183642888)

[Gürültü Azaltma 17](#_Toc183642889)

[Döndürme ve Ayna Çevirme 18](#_Toc183642890)

[Frekans Alanında Filtreleme 20](#_Toc183642891)

[Keskinleştirme ve Enterpolasyon 22](#_Toc183642892)

[Sonuç ve Değerlendirme Raporu 24](#_Toc183642893)

# Görüntü İşleme Ara Sınav Ödevi

## Veri Seti Hakkında Kısa Bilgi

**ChestX-ray8 veri seti** kullanılarak göğüs röntgeni görüntüleri üzerinde veri keşfi ve görüntü işleme tekniklerini uygulamak üzere ödev hazırlanmıştır.

ChestX-ray8 (veya ChestX-ray14 olarak da bilinir) NIH Clinical Center tarafından yayınlanan geniş bir göğüs röntgeni veri setidir. Temel özellikleri:

* 30,805 hastadan alınan 112,120 ön-arka göğüs röntgeni görüntüsü içerir.
* 14 farklı göğüs hastalığı etiketini (pnömoni, nodül, kitle vb.) içeren detaylı hastalık etiketleri
* Her görüntü 1024x1024 piksel çözünürlüğünde
* DICOM formatından PNG'ye dönüştürülmüş görüntüler
* Hastalık tespiti ve sınıflandırması için yapay zeka modellerinin eğitiminde yaygın olarak kullanılır
* Görüntüler ve etiketler NLP ile radyoloji raporlarından otomatik çıkarılmıştır

Bu veri seti tıbbi görüntü analizi ve derin öğrenme araştırmalarında önemli bir kaynak olarak kabul edilir.

## Gerekli Kütüphanelerin Yüklenmesi

Aşağıdaki kütüphaneler kullanılacaktır.

# 1. Gerekli Kütüphanelerin İçe Aktarılması

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

import os

import cv2

from skimage import exposure

## Verilerin Yüklemesi Adımı

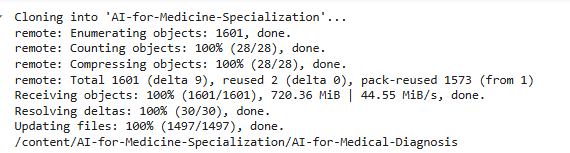
Veriler için github kütüphanesi reposundan veriler çekilmiştir;

# GitHub reposunu klonlayalım

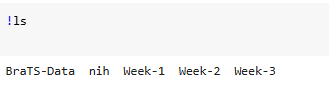
!git clone https://github.com/hardik0/AI-for-Medicine-Specialization

# Çalışma dizinini ayarlayalım

%cd AI-for-Medicine-Specialization/AI-for-Medical-Diagnosis/



Aşağıdaki dizinler bulunmaktadır;



# 2. Veri Setinin Yüklenmesi

train\_df = pd.read\_csv("nih/train-small.csv")

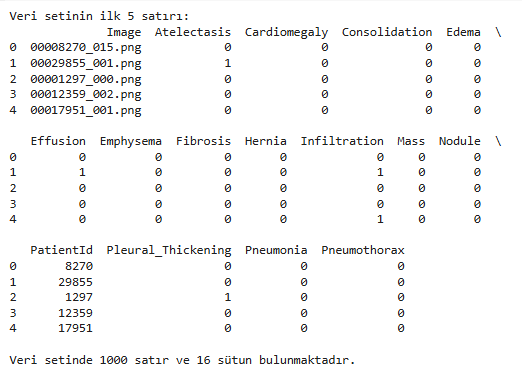
# İlk 5 satırı göster

print("Veri setinin ilk 5 satırı:")

print(train\_df.head())

# Boyut bilgisi

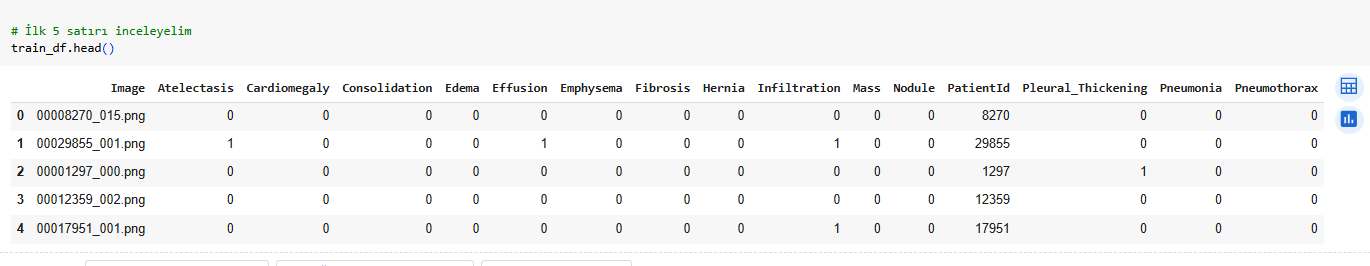
print(f'\nVeri setinde {train\_df.shape[0]} satır ve {train\_df.shape[1]} sütun bulunmaktadır.')



Toplam 1000 görüntü içermektedir. Her görüntü için çeşitli hastalık durumları (0/1) belirtilmiştir

PatientId sütunu hasta kimlik bilgisini içermektedir.

## Verilerin İncelenmesi – Ön Analizi



Buradaki kolon bilgileri;

Veri setindeki kolonlar:

1. Image: Röntgen görüntüsünün dosya adı

2. Atelectasis: Akciğerde sönme/kollaps durumu

3. Cardiomegaly: Kalp büyümesi

4. Consolidation: Akciğerde yoğunlaşma

5. Edema: Ödem, sıvı birikimi

6. Effusion: Plevral efüzyon, akciğer zarında sıvı

7. Emphysema: Amfizem, hava keseciklerinin genişlemesi

8. Fibrosis: Akciğer dokusu sertleşmesi

9. Hernia: Fıtık

10. Infiltration: Akciğerde infiltrasyon

11. Mass: Kitle

12. Nodule: Nodül, küçük yumru

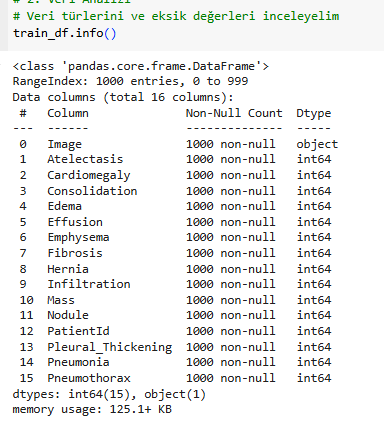
13. PatientId: Hasta kimlik numarası

14. Pleural\_Thickening: Plevra kalınlaşması

15. Pneumonia: Zatürre

16. Pneumothorax: Akciğer zarları arasında hava birikimi

0 ve 1 değerleri ilgili hastalığın varlığını/yokluğunu gösterir.



Veri setinin yapısal analizi şunu gösteriyor:

- 1000 örnek (0-999 arası)

- 16 kolon

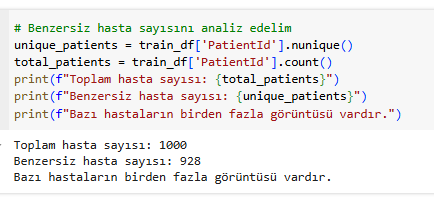
- Tüm kolonlar eksiksiz (non-null)

- Image kolonu hariç tüm veriler int64 tipinde

- Image kolonu object (string) tipinde

- Bellek kullanımı 125.1+ KB

Veri seti temiz ve düzenli görünüyor, eksik veri yok, veri tipleri tutarlı.



Aşağıda veriyi özellikle veri dengesizliği açısından inceliyoruz.

# Toplam satır ve sütun sayısını yazdıralım

print(f'Veri setinin boyutu:')

print(f'Veri setinde {train\_df.shape[0]} satır ve {train\_df.shape[1]} sütun bulunmaktadır')

# Benzersiz hasta sayısını analiz edelim

print('\nHasta ID analizi:')

toplam\_hasta = train\_df['PatientId'].count()

benzersiz\_hasta = train\_df['PatientId'].nunique()

print(f"Toplam hasta kaydı sayısı: {toplam\_hasta}")

print(f"Benzersiz hasta sayısı: {benzersiz\_hasta}")

print(f"Bu durumda {toplam\_hasta - benzersiz\_hasta} adet tekrarlanan hasta kaydı bulunmaktadır.")

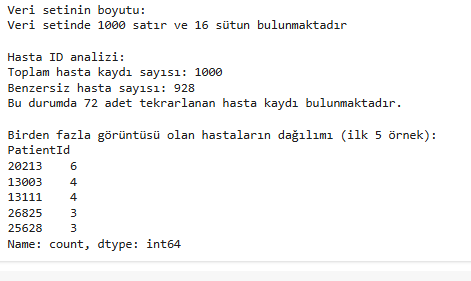
# Tekrarlanan hasta kayıtlarını gösterelim

tekrar\_eden\_hastalar = train\_df[train\_df.duplicated(['PatientId'], keep=False)]['PatientId'].value\_counts()

if len(tekrar\_eden\_hastalar) > 0:

    print("\nBirden fazla görüntüsü olan hastaların dağılımı (ilk 5 örnek):")

    print(tekrar\_eden\_hastalar.head())



Buna göre;

1. Veri kalitesi:

- Eksik veri yok

- Veri tipleri tutarlı

2. Hasta dağılımı:

- Bazı hastaların tekrarlı görüntüleri var (72 hasta)

- En çok görüntüsü olan hasta (ID: 20213) 6 görüntüye sahip

- Çoğu tekrarlı hasta 3-4 görüntüye sahip

Veri dağılımının belki dengesiz olduğu söylenebilir ama bu oran çok düşük. %7.2'lik (72/1000) tekrar oranı çok düşük bir dengesizlik. Model eğitimini önemli ölçüde etkilemeyecektir, özel bir işlem yapmaya gerek yok. Normal eğitime devam edebiliriz.

## Veri Görselleştirme Adımı

Veri görselleştirme adımında rastgele 9 görüntü seçelim;

# 3. Rastgele Görüntü Seçme ve Görselleştirme

# Rastgele 9 görüntü seçelim

images = train\_df['Image'].values

random\_images = np.random.choice(images, 9, replace=False)

img\_dir = 'nih/images\_small/'

# Görüntüleri gösterelim

plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

    img = plt.imread(os.path.join(img\_dir, img\_name))

    plt.subplot(3, 3, i + 1)

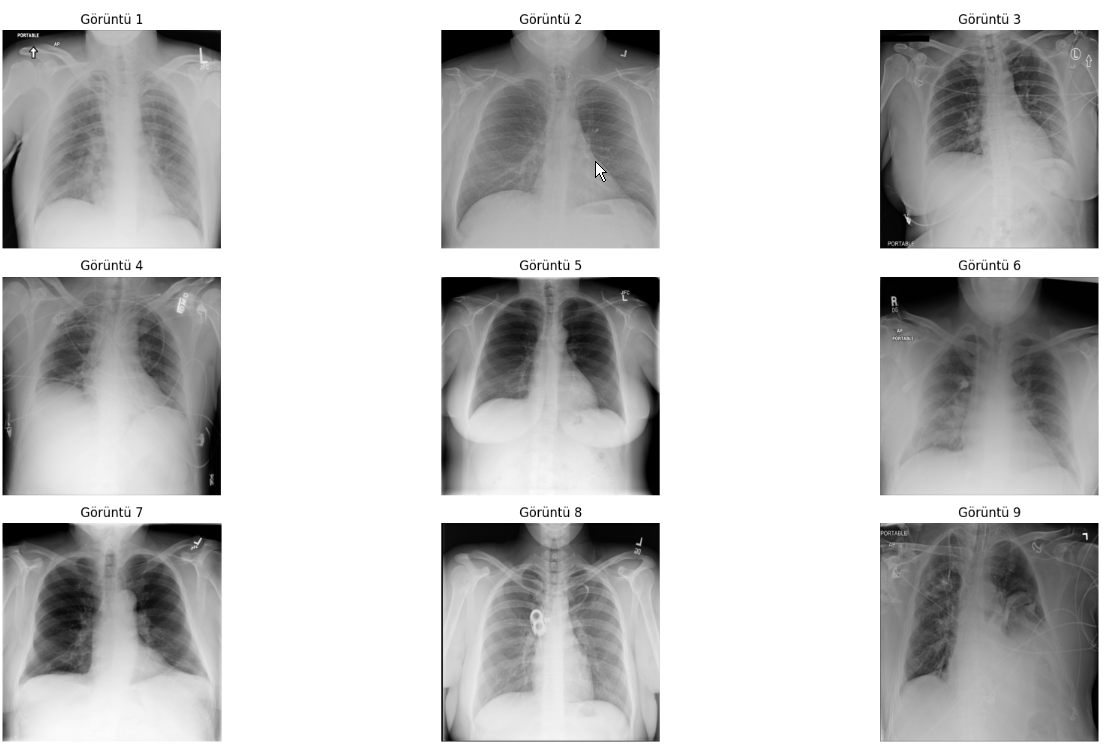
    plt.imshow(img, cmap='gray')

    plt.title(f'Görüntü {i+1}')

    plt.axis('off')

plt.tight\_layout()

plt.show()



## Verilerin/Görüntülerin İstatistiksel Olarak Hesaplanması

# 4. İstatistiksel Özelliklerin Hesaplanması

# Her bir görüntü için istatistikleri hesaplayalım

for i, img\_name in enumerate(random\_images):

    img = plt.imread(os.path.join(img\_dir, img\_name))

    print(f"\nGörüntü {i+1} İstatistikleri:")

    print(f"Maksimum değer: {img.max():.4f}")

    print(f"Minimum değer: {img.min():.4f}")

    print(f"Ortalama değer: {img.mean():.4f}")

    print(f"Standart sapma: {img.std():.4f}")

# Her bir görüntünün istatistiksel özelliklerini hesaplayalım ve gösterelim

plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

    img = plt.imread(os.path.join(img\_dir, img\_name))

    plt.subplot(3, 3, i + 1)

    plt.imshow(img, cmap='gray')

    # İstatistikleri hesapla

    max\_val = img.max()

    min\_val = img.min()

    mean\_val = img.mean()

    std\_val = img.std()

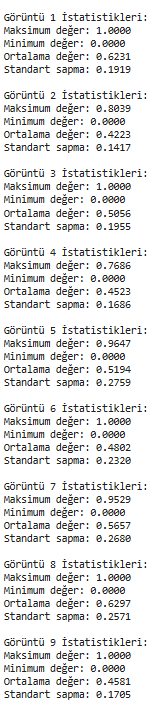
    plt.title(f'Görüntü {i+1}\nMax: {max\_val:.4f}\nMin: {min\_val:.4f}\n'

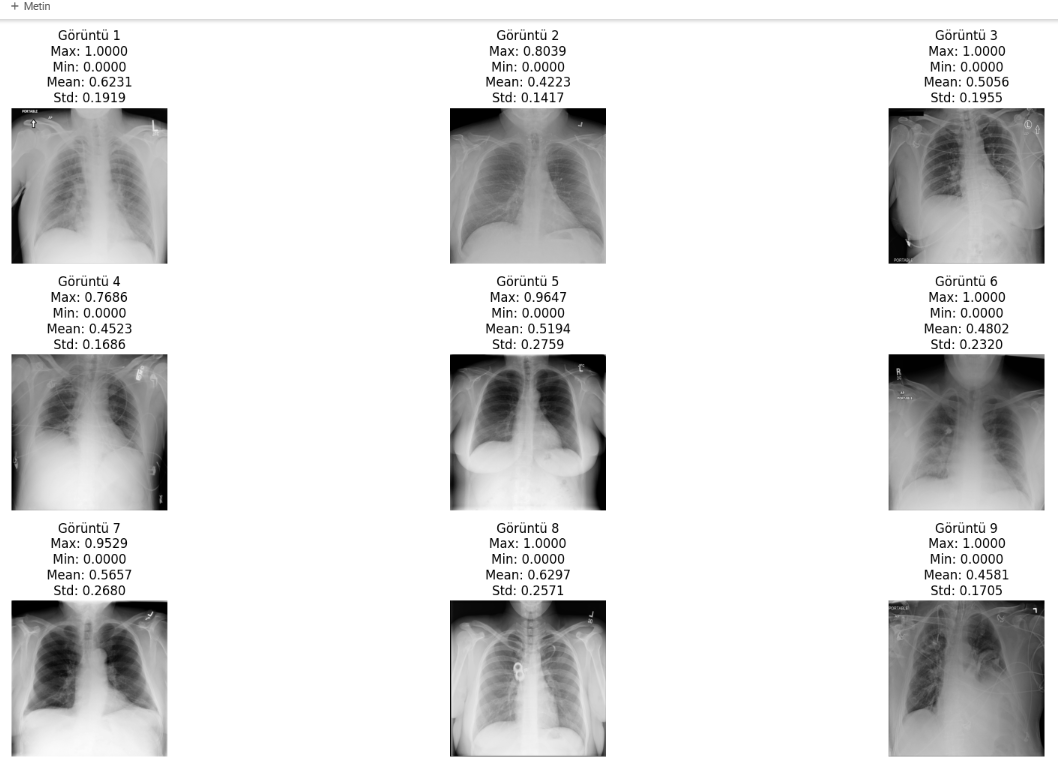
              f'Mean: {mean\_val:.4f}\nStd: {std\_val:.4f}')

    plt.axis('off')

plt.tight\_layout()

plt.show()





İstatistiksel değerler görüntülerin piksel yoğunluk dağılımını gösterir:

- Piksel yoğunluk aralığı tüm görüntülerde 0-1 arasında normalize edilmiş

- Görüntü 1, 3, 6, 8 ve 9'da maksimum piksel değeri 1.0000 (tam beyaz noktalar mevcut)

- Ortalama piksel değerleri 0.42-0.62 arasında değişiyor

- En düşük standart sapma Görüntü 2'de (0.1417), en yüksek Görüntü 5'te (0.2759)

- Minimum değer tüm görüntülerde 0.0000 (tam siyah noktalar mevcut)

Bu değerler görüntülerin kontrast ve parlaklık dağılımlarının farklılık gösterdiğini, ancak genel olarak benzer aralıklarda olduğunu gösteriyor.

Standart Sapma (Std):

- Piksellerin ortalama değerden ne kadar saptığını gösterir

- Düşük std (örn. Görüntü 2: 0.1417): Piksel değerleri ortalamaya yakın, kontrast düşük

- Yüksek std (örn. Görüntü 5: 0.2759): Piksel değerleri daha dağınık, kontrast yüksek

Mean (Ortalama):

- Görüntünün genel parlaklık seviyesini gösterir

- Düşük ortalama: Daha koyu görüntü

- Yüksek ortalama: Daha parlak görüntü

Bu değerler görüntü kalitesi ve normalleştirme için önemlidir.

## Histogram Analizi

Histogram analizi medikal görüntülerde piksel yoğunluğunun dağılımını gösterir:

Temel Amaçlar:

* Kontrast analizi
* Dinamik aralık değerlendirmesi
* Görüntü kalitesi kontrolü
* Normalleştirme ihtiyacının belirlenmesi

Farklı histogram şekilleri:

* Tek tepeli: Homojen görüntü
* Çift tepeli: İki baskın yoğunluk seviyesi
* Geniş dağılım: Yüksek kontrast
* Dar dağılım: Düşük kontrast

# 5. Histogram Analizi

# Her görüntü için histogram çizelim

'''plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

    img = plt.imread(os.path.join(img\_dir, img\_name))

    plt.subplot(3, 3, i + 1)

    plt.hist(img.ravel(), bins=256, color='gray')

    plt.title(f'Histogram {i+1}')

plt.tight\_layout()

plt.show()

'''

# Her görüntü için histogram çizelim

plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

    img = plt.imread(os.path.join(img\_dir, img\_name))

    plt.subplot(3, 3, i + 1)

    plt.hist(img.ravel(), bins=256, color='gray', alpha=0.7)

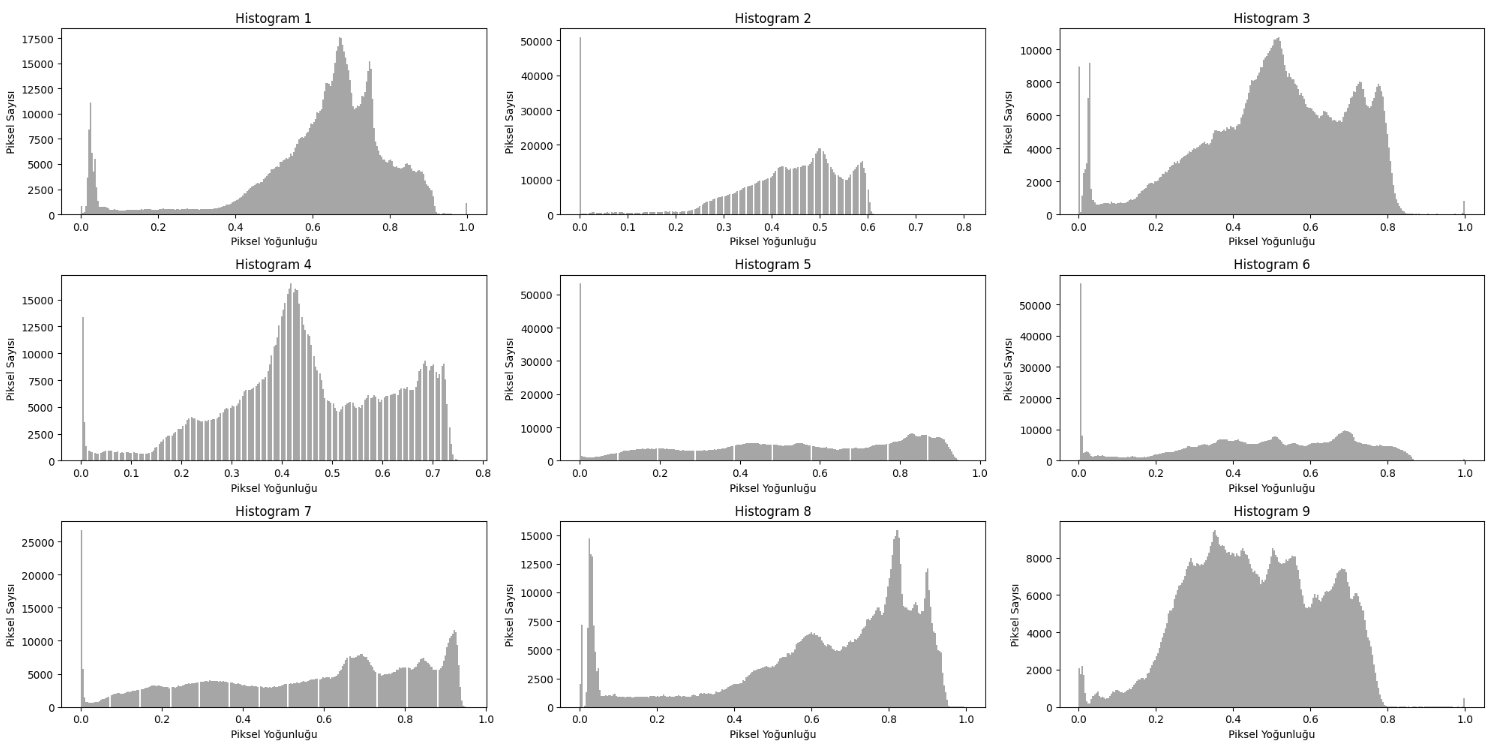
    plt.title(f'Histogram {i+1}')

    plt.xlabel('Piksel Yoğunluğu')

    plt.ylabel('Piksel Sayısı')

plt.tight\_layout()

plt.show()



Her histogramın gösterdiği önemli noktalar:

1. Görüntü 1, 3, 8: Yüksek piksel yoğunluğu (0.6-0.8 arası), yüksek kontrast, kemik yapılar belirgin

2. Görüntü 2, 4: Orta yoğunluk (0.3-0.6 arası), dengeli kontrast, yumuşak doku detayları iyi görünür.

3. Görüntü 5, 6, 7: Geniş dağılım, düşük kontrast, detay kaybı riski.

4. Tüm görüntülerde 0'a yakın değerlerde küçük pikler var (arka plan siyah bölgeler)

Bu dağılımlar görüntü iyileştirme ihtiyacını belirlemede önemli. Özellikle 5, 6, 7 numaralı görüntülerde kontrast iyileştirmesi faydalı olabilir.

Genel olarak yorumlarsak;

\* Genel Yorumlar

Düşük yoğunluklu bölge (koyu) akciğer dokusunu temsil ediyor

Yüksek yoğunluklu bölge (açık) kemik yapıları temsil ediyor

\* Piksel yoğunluğu (x ekseni):

Sol taraftaki değerler (0.0): Koyu renk (siyah) pikselleri temsil eder.

Sağ taraftaki değerler (1.0): Açık renk (beyaz) pikselleri temsil eder.

Orta değerler (0.5 civarı): Gri tonlarını temsil eder.

\* Piksel sayısı (y ekseni):

Dik eksende belirli bir yoğunluğa sahip kaç piksel olduğunu gösterir.

Pik yüksekliği, o yoğunluğa sahip piksel sayısının ne kadar fazla olduğunu belirtir.

## Görüntü İyileştirme

Bu bölümde görüntü kalitesini artırma için çeşitli teknikler uygulayacağız.

Her bir görüntü için;

* Kontrast germe
* Histogram eşitleme
* Gamma düzeltme

İşlem ve işlemlerini yapalım.

### Kontrast Germe(Contrast Stretching)

* Piksel değerlerini daha geniş bir aralığa yayar
* min-max normalizasyon kullanır

formülasyonu şu şekildedir;



### Histogram Eşitleme

* Piksel yoğunluklarını eşit dağıtır
* Kümülatif dağılım kullanır



### Gamma Düzeltme

* Parlaklık düzeylerini ayarlar
* Karanlık/aydınlık bölgeleri dengeler



Şimdi örneklerimizde bunları uygulayalım;

# 6.1 Görüntü İyileştirme

# Örnek bir görüntü üzerinde işlemler yapalım. Bu bölümde görüntü kalitesini artırmak için çeşitli teknikler uygulayacağız.

#sample\_img = plt.imread(os.path.join(img\_dir, random\_images[0]))

# Her görüntü için kontrast germe, histogram eşitleme ve gamma düzeltme uygulayalım

for i, img\_name in enumerate(random\_images[:3]): # İlk 3 görüntü için yapalım

    img = plt.imread(os.path.join(img\_dir, img\_name))

    # Kontrast germe

    min\_val, max\_val = img.min(), img.max()

    contrast\_stretched = (img - min\_val) / (max\_val - min\_val)

    contrast\_stretched = (contrast\_stretched \* 255).astype(np.uint8)

    # Histogram eşitleme

    equalized = exposure.equalize\_hist(contrast\_stretched)

    equalized = (equalized \* 255).astype(np.uint8)

    # Gamma düzeltme

    gamma = 1.2

    gamma\_corrected = np.power(equalized / 255.0, gamma) \* 255

    gamma\_corrected = gamma\_corrected.astype(np.uint8)

    # Sonuçları gösterelim

    plt.figure(figsize=(15,5))

    plt.subplot(141)

    plt.imshow(img, cmap='gray')

    plt.title(f'Orijinal {i+1}')

    plt.axis('off')

    plt.subplot(142)

    plt.imshow(contrast\_stretched, cmap='gray')

    plt.title('Kontrast Germe')

    plt.axis('off')

    plt.subplot(143)

    plt.imshow(equalized, cmap='gray')

    plt.title('Histogram Eşitleme')

    plt.axis('off')

    plt.subplot(144)

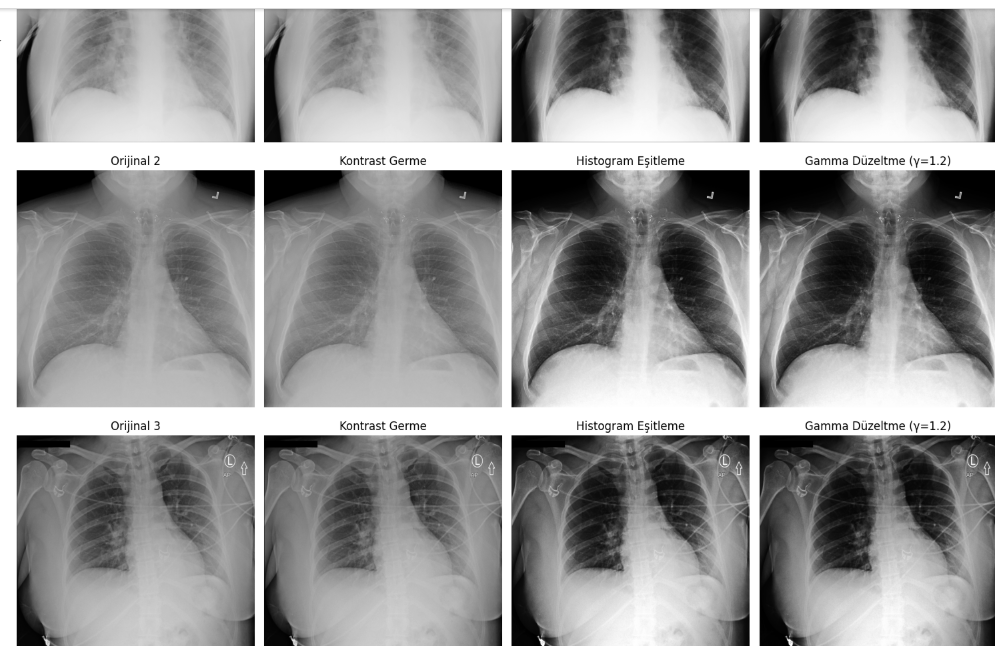
    plt.imshow(gamma\_corrected, cmap='gray')

    plt.title(f'Gamma Düzeltme (γ={gamma})')

    plt.axis('off')

    plt.tight\_layout()

    plt.show()



Görüntü işleme sonuçlarının analizi:

1. Kontrast Germe:

- Minimal değişim gösteriyor

- Orijinal görüntüler zaten iyi kontrastlı

2. Histogram Eşitleme:

- En belirgin iyileştirme

- Kemik yapılar ve yumuşak doku detayları daha net

- Akciğer alanları daha belirgin

3. Gamma Düzeltme (γ=1.2):

- Histogram eşitlemeye benzer sonuçlar

- Daha yumuşak geçişler

- Aşırı kontrast artışı yok

Sonuç: Bu veri seti için histogram eşitleme en etkili yöntem görünüyor.

### Tek Görüntü Üzerinde Detaylı İyileştirme İncelemesi

Tek görüntü üzerinde biraz daha detaylı inceleme yaparsak;

# 6.2 Görüntü İyileştirme

'''

Görüntü İyileştirme Yorumu:

1. Kontrast Germe:

   - Piksel değerlerini tam aralığa yayarak kontrastı artırır

   - Düşük kontrastlı bölgelerin görünürlüğünü iyileştirir

'''

from skimage import exposure

# Örnek bir görüntü yükleyelim

sample\_img = train\_df.Image[0]  # İlk görüntüyü alalım

raw\_image = plt.imread(os.path.join(img\_dir, sample\_img))

# Calculate minimum and maximum for contrast stretching

min\_val, max\_val = raw\_image.min(), raw\_image.max()

# Step 1: Apply contrast stretching

min\_val, max\_val = raw\_image.min(), raw\_image.max()

contrast\_stretched = (raw\_image - min\_val) / (max\_val - min\_val)  # Normalize to [0, 1]

contrast\_stretched = (contrast\_stretched \* 255).astype(np.uint8)  # Scale to [0, 255] for visualization

# Step 2: Apply histogram equalization after contrast stretching

# Step 2: Apply histogram equalization using skimage

equalized\_image = exposure.equalize\_hist(contrast\_stretched)  # Returns values in [0, 1]

equalized\_image = (equalized\_image \* 255).astype(np.uint8)  # Scale to [0, 255]

# Step 3: Apply gamma correction on the equalized image

gamma = 1.2  # Adjust gamma value as needed

gamma\_corrected = np.power(equalized\_image / 255.0, gamma) \* 255

gamma\_corrected = gamma\_corrected.astype(np.uint8)

# Sonuçları gösterelim

plt.figure(figsize=(18, 10))

plt.subplot(2, 4, 1)

plt.imshow(raw\_image, cmap='gray')

plt.title('Original Image')

plt.colorbar()

plt.subplot(2, 4, 2)

plt.imshow(contrast\_stretched, cmap='gray')

plt.title('Contrast-Stretched Image')

plt.colorbar()

plt.subplot(2, 4, 3)

plt.imshow(equalized\_image, cmap='gray')

plt.title('Contrast Equalized Image')

plt.colorbar()

plt.subplot(2, 4, 4)

plt.imshow(gamma\_corrected, cmap='gray')

plt.title(f'Gamma-Corrected Image (Gamma={gamma})')

plt.colorbar()

# Histogramları gösterelim

plt.subplot(2, 4, 5)

plt.hist(raw\_image.ravel(), bins=256, color='gray')

plt.title("Original Image Histogram")

plt.subplot(2, 4, 6)

plt.hist(contrast\_stretched.ravel(), bins=256, color='gray')

plt.title("Contrast-Stretched Histogram")

plt.subplot(2, 4, 7)

plt.hist(equalized\_image.ravel(), bins=256, color='gray')

plt.title("Equalized Histogram")

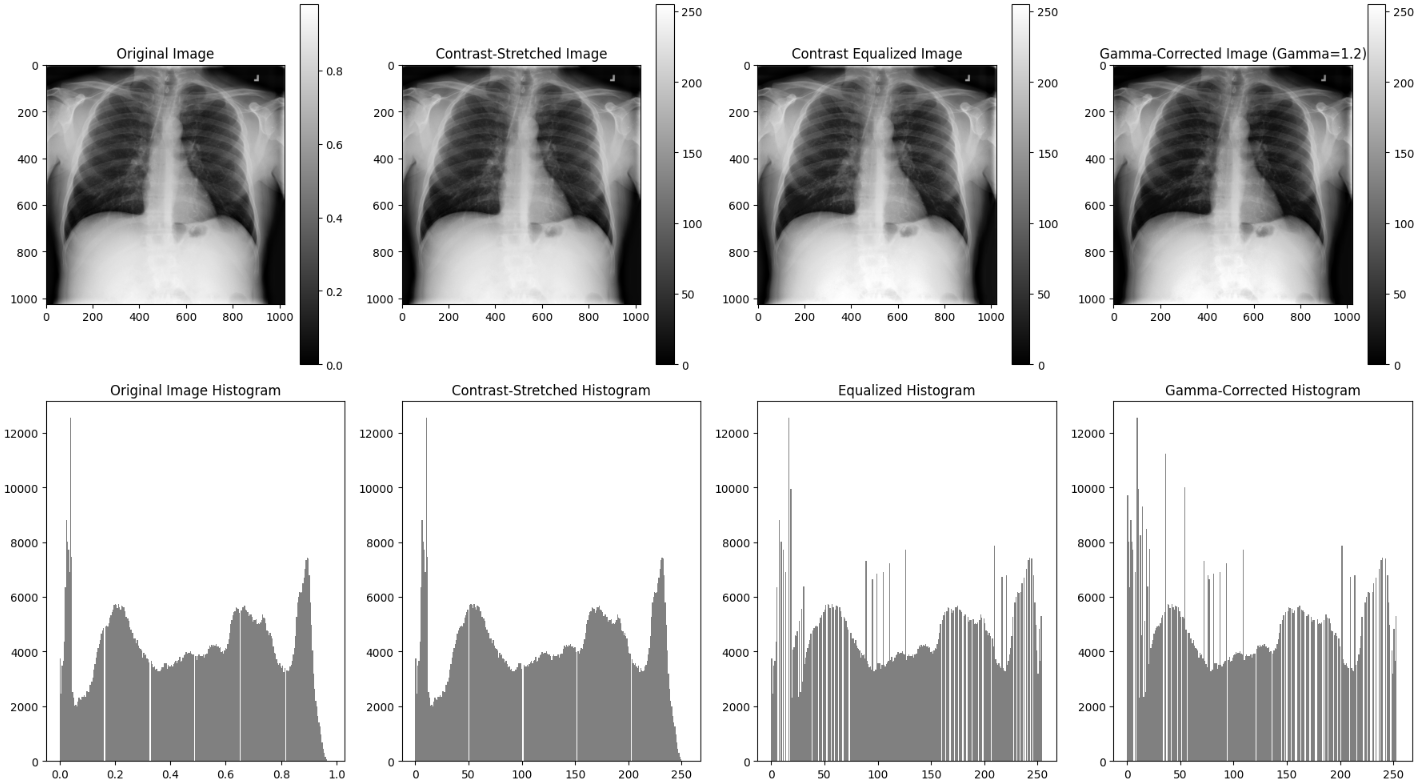
plt.subplot(2, 4, 8)

plt.hist(gamma\_corrected.ravel(), bins=256, color='gray')

plt.title("Gamma-Corrected Histogram")

plt.tight\_layout()

plt.show()



from skimage.io import imread

# Eğer sample\_img bir dosya adı içeriyorsa, tam yolu oluşturun ve yükleyin

file\_path = os.path.join(img\_dir, sample\_img)  # Dosya yolunu birleştir

sample\_img = imread(file\_path)  # Görüntüyü yükle

# Kontrast germe

min\_val, max\_val = sample\_img.min(), sample\_img.max()

contrast\_stretched = (sample\_img - min\_val) / (max\_val - min\_val)

contrast\_stretched = (contrast\_stretched \* 255).astype(np.uint8)

# Histogram eşitleme

'''

 Histogram Eşitleme:

   - Piksel dağılımını dengeler

   - Detayların daha belirgin hale gelmesini sağlar

'''

equalized = exposure.equalize\_hist(contrast\_stretched)

equalized = (equalized \* 255).astype(np.uint8)

# Gamma düzeltme

'''

Gamma Düzeltme:

   - Parlaklık ayarlaması yapar

   - Gamma>1 koyu bölgeleri daha belirgin hale getirir

'''

gamma = 1.2

gamma\_corrected = np.power(equalized / 255.0, gamma) \* 255

gamma\_corrected = gamma\_corrected.astype(np.uint8)

# Sonuçları gösterelim

plt.figure(figsize=(15,5))

images = [sample\_img, contrast\_stretched, equalized, gamma\_corrected]

titles = ['Orijinal', 'Kontrast Germe', 'Histogram Eşitleme', 'Gamma Düzeltme']

for i, (img, title) in enumerate(zip(images, titles)):

    plt.subplot(1, 4, i+1)

    plt.imshow(img, cmap='gray')

    plt.title(title)

    plt.axis('off')

plt.tight\_layout()

plt.show()



Görüntü işleme sonuçlarının analizi:

1. Orijinal görüntü: Standart X-ray görüntüsü, orta düzey kontrast

2. Kontrast Germe:

- Minimal değişim

- min-max normalizasyonu (0-255) uygulanmış

3. Histogram Eşitleme:

- Detaylar daha belirgin

- Akciğer dokusundaki ince ayrıntılar görünür

- En etkili iyileştirme bu aşamada

4. Gamma Düzeltme (γ=1.2):

- Orta ton değerlerinde ince ayar

- Histogram eşitlemeye göre daha yumuşak geçişler

Kod, sıralı işlemlerle (kontrast->histogram->gamma) görüntü kalitesini kademeli olarak iyileştiriyor.

## Gürültü Azaltma

Gürültü azaltma (noise reduction), görüntüdeki istenmeyen rastgele varyasyonların veya bozulmaların temizlenmesi işlemidir.

Gürültü Türleri:

1. Salt-and-pepper (tuz-biber): Rastgele siyah ve beyaz pikseller

2. Gaussian gürültü: Normal dağılımlı rastgele bozulmalar

3. Speckle gürültü: Çarpımsal gürültü

Tıbbi görüntülerde gürültü kaynakları:

- X-ray dedektör sistemleri

- Elektriksel parazitler

- Düşük doz çekimler

Gürültü azaltma önemlidir çünkü:

- Tanı doğruluğunu artırır

- Görüntü detaylarını belirginleştirir

- Otomatik analiz sistemlerinin performansını iyileştirir.

# 7.1 Gürültü Azaltma

# Median ve Gaussian filtreleme uygulayalım

#Bu bölümde görüntüdeki gürültüyü azaltmak için filtreleme teknikleri uygulayacağız.

median\_filtered = cv2.medianBlur(gamma\_corrected, 5)

gaussian\_filtered = cv2.GaussianBlur(gamma\_corrected, (5,5), 0)

# Sonuçları gösterelim

plt.figure(figsize=(15,5))

plt.subplot(131)

plt.imshow(gamma\_corrected, cmap='gray')

plt.title('Gamma Düzeltme Sonrası')

plt.axis('off')

plt.subplot(132)

plt.imshow(median\_filtered, cmap='gray')

plt.title('Median Filtre')

plt.axis('off')

plt.subplot(133)

plt.imshow(gaussian\_filtered, cmap='gray')

plt.title('Gaussian Filtre')

plt.axis('off')

plt.tight\_layout()

plt.show()

Gürültü azaltma teknikleri ve sonuçların analizi:

1. Median Filtre:

- Salt-pepper gürültüsünü giderir

- Piksel komşuluğundaki medyan değeri kullanır

- Kenarları korur

- Kod: `cv2.medianBlur(gamma\_corrected, 5)`

2. Gaussian Filtre:

- Yumuşak geçişli bulanıklaştırma yapar

- Gaussian dağılım kullanır

- Detayları koruyarak gürültüyü azaltır

- Kod: `cv2.GaussianBlur(gamma\_corrected, (5,5), 0)`

Görüntü sonuçları:

- Median: Keskin detayları korurken gürültüyü azaltmış

- Gaussian: Daha yumuşak geçişler sağlamış

- Her iki filtre de X-ray görüntüsünün kalitesini artırmış

## Döndürme ve Ayna Çevirme

Döndürme (Rotation) ve ayna çevirme (Flipping) görüntü dönüştürme teknikleridir:

Döndürme:

- Görüntüyü belirli açılarla döndürür (90°, 180°, 270°)

- Kullanım: `cv2.rotate(image, cv2.ROTATE\_90\_CLOCKWISE)`

Ayna Çevirme:

- Yatay çevirme: Sol-sağ yönünde aynalama

- Dikey çevirme: Yukarı-aşağı yönünde aynalama

- Kullanım: `cv2.flip(image, 1)` (yatay) veya `cv2.flip(image, 0)` (dikey)

Medikal görüntülemede kullanım amaçları:

- Görüntü standardizasyonu

- Veri çoğaltma (data augmentation)

- Anatomik yönlendirme düzeltme

# 8.1 Döndürme ve Ayna Çevirme

# Rastgele açı ile döndürme

angle = np.random.uniform(0, 10)

center = (median\_filtered.shape[1] // 2, median\_filtered.shape[0] // 2)

rotation\_matrix = cv2.getRotationMatrix2D(center, angle, 1.0)

rotated = cv2.warpAffine(median\_filtered, rotation\_matrix,

                        (median\_filtered.shape[1], median\_filtered.shape[0]))

# Yatay çevirme

flipped = cv2.flip(median\_filtered, 1)

# Sonuçları gösterelim

plt.figure(figsize=(15,5))

plt.subplot(131)

plt.imshow(median\_filtered, cmap='gray')

plt.title('Orijinal')

plt.axis('off')

plt.subplot(132)

plt.imshow(rotated, cmap='gray')

plt.title(f'Döndürülmüş ({angle:.1f}°)')

plt.axis('off')

plt.subplot(133)

plt.imshow(flipped, cmap='gray')

plt.title('Yatay Çevrilmiş')

plt.axis('off')

plt.tight\_layout()

plt.show()



Bu dönüşümler iki amaçla yapılır:

1. Yanlış pozisyonda çekilmiş röntgenleri düzeltmek (6.5° döndürme)

2. Görüntüleri çoğaltarak yapay zeka modelinin eğitim verisini artırmak (yatay çevirme)

Tıpkı eğri duran bir fotoğrafı düzeltmek gibi, röntgen görüntülerini de düzeltebilir veya çoğaltabiliriz.

## Frekans Alanında Filtreleme

Frekans alanında filtreleme üç ana kavramı içerir:

1. Fourier Dönüşümü (DFT):

- Görüntüyü uzamsal alandan frekans alanına dönüştürür. Görüntüyü farklı frekanslara ayırır. Tıpkı müziğin bas, orta ve tiz sesler gibi ayrılması benzeri.

- `cv2.dft()` fonksiyonu kullanılır

2. Düşük Geçiren Filtre:

- Yüksek frekanslı bileşenleri (gürültü ve keskin kenarlar) atenüe eder

- Merkezdeki 60x60 piksellik alanı korur

- Yumuşatma etkisi yaratır

3. Ters Fourier Dönüşümü:

- Filtrelenmiş görüntüyü tekrar uzamsal alana dönüştürür

- `cv2.idft()` ile gerçekleştirilir

Sonuçta gürültü azaltılmış, yumuşatılmış bir görüntü elde edilir.

# 9. Frekans Alanında Filtreleme

# Fourier dönüşümü

dft = cv2.dft(np.float32(rotated), flags=cv2.DFT\_COMPLEX\_OUTPUT)

dft\_shift = np.fft.fftshift(dft)

# Düşük geçiren filtre maskesi

rows, cols = rotated.shape

crow, ccol = rows//2, cols//2

mask = np.zeros((rows, cols, 2), np.uint8)

mask[crow-30:crow+30, ccol-30:ccol+30] = 1

# Filtreleme ve ters dönüşüm

fshift = dft\_shift \* mask

f\_ishift = np.fft.ifftshift(fshift)

img\_back = cv2.idft(f\_ishift)

filtered = cv2.magnitude(img\_back[:,:,0], img\_back[:,:,1])

# Sonuçları gösterelim

plt.figure(figsize=(15,5))

plt.subplot(131)

plt.imshow(rotated, cmap='gray')

plt.title('Orijinal')

plt.axis('off')

plt.subplot(132)

spectrum = np.log(1 + np.abs(dft\_shift[:,:,0]))

plt.imshow(spectrum, cmap='gray')

plt.title('Fourier Spektrumu')

plt.axis('off')

plt.subplot(133)

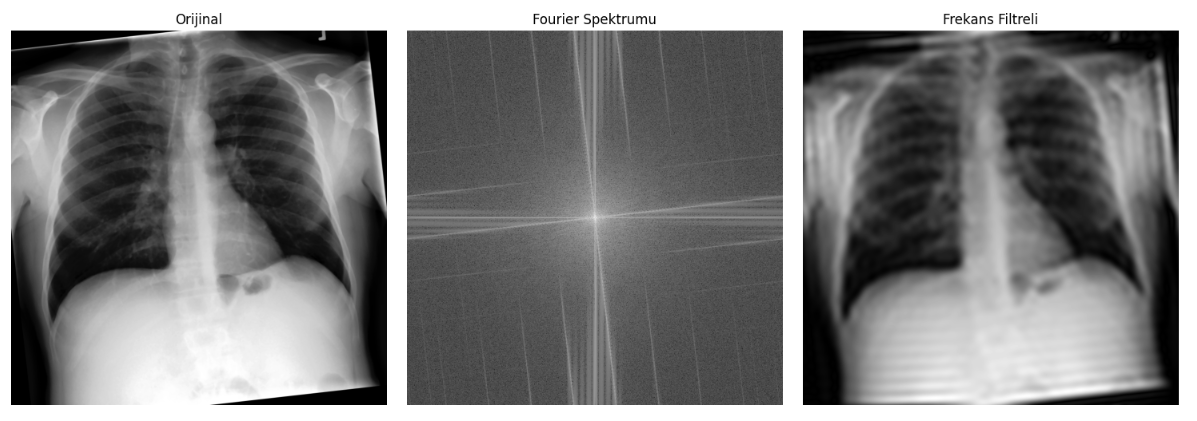
plt.imshow(filtered, cmap='gray')

plt.title('Frekans Filtreli')

plt.axis('off')

plt.tight\_layout()

plt.show()



Sonuçları karşılaştıralım:

1. Orijinal: Normal X-ray görüntüsü

2. Fourier Spektrumu: Görüntünün frekans dağılımı (ortadaki parlak nokta düşük frekansları gösterir)

3. Frekans Filtreli: Daha yumuşak, gürültüsü azaltılmış görüntü

Son görüntüde kenarlar ve detaylar biraz yumuşamış, ancak ana yapılar (kemikler ve akciğer dokusu) korunmuş durumda. Bu durumda görüntü kalitesi zaten iyi olduğu için belirgin bir iyileştirme sağlamadı. Filtre daha çok gürültülü, düşük kaliteli X-ray görüntülerinde faydalıdır.

## Keskinleştirme ve Enterpolasyon

Keskinleştirme ve enterpolasyon iki farklı amaçla yapılır:

Keskinleştirme:

- Görüntüdeki detayları belirginleştirir

- Özellikle tıbbi görüntülerde lezyonları ve yapıları daha görünür kılar

- Tanı koymayı kolaylaştırır

*Gaussian bulanıklaştırma uygulanır*

*Orijinal görüntü ile bulanık görüntü birleştirilir*

*Detayları ve kenarları belirginleştirir*

Enterpolasyon:

- Görüntüyü büyütürken kalite kaybını minimize eder

- Küçük detayları incelemek için zoom yapılabilir

- Farklı boyutlardaki görüntüleri standart hale getirir

*Görüntü boyutu 2 katına çıkarılır*

*Doğrusal (linear) enterpolasyon kullanır*

*Yeni pikseller mevcut pikseller arasında hesaplanır*

Her iki işlem de özellikle medikal görüntü analizinde tanı doğruluğunu artırmak için önemlidir.

# Keskinleştirme

plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

   img = plt.imread(os.path.join(img\_dir, img\_name))

   # Gaussian blur

   blurred = cv2.GaussianBlur(img, (0,0), 1.0)

   # Keskinleştirme

   sharpened = cv2.addWeighted(img, 1 + 1.5, blurred, -1.5, 0)

   plt.subplot(3, 3, i + 1)

   plt.imshow(sharpened, cmap='gray')

   plt.title(f'Keskinleştirilmiş Görüntü {i+1}')

   plt.axis('off')

plt.tight\_layout()

plt.show()

# Enterpolasyon

plt.figure(figsize=(20,10))

for i, img\_name in enumerate(random\_images):

   img = plt.imread(os.path.join(img\_dir, img\_name))

   # Yeni boyutlar

   new\_width = int(img.shape[1] \* 2)

   new\_height = int(img.shape[0] \* 2)

   # Enterpolasyon

   resized = cv2.resize(img, (new\_width, new\_height), interpolation=cv2.INTER\_LINEAR)

   plt.subplot(3, 3, i + 1)

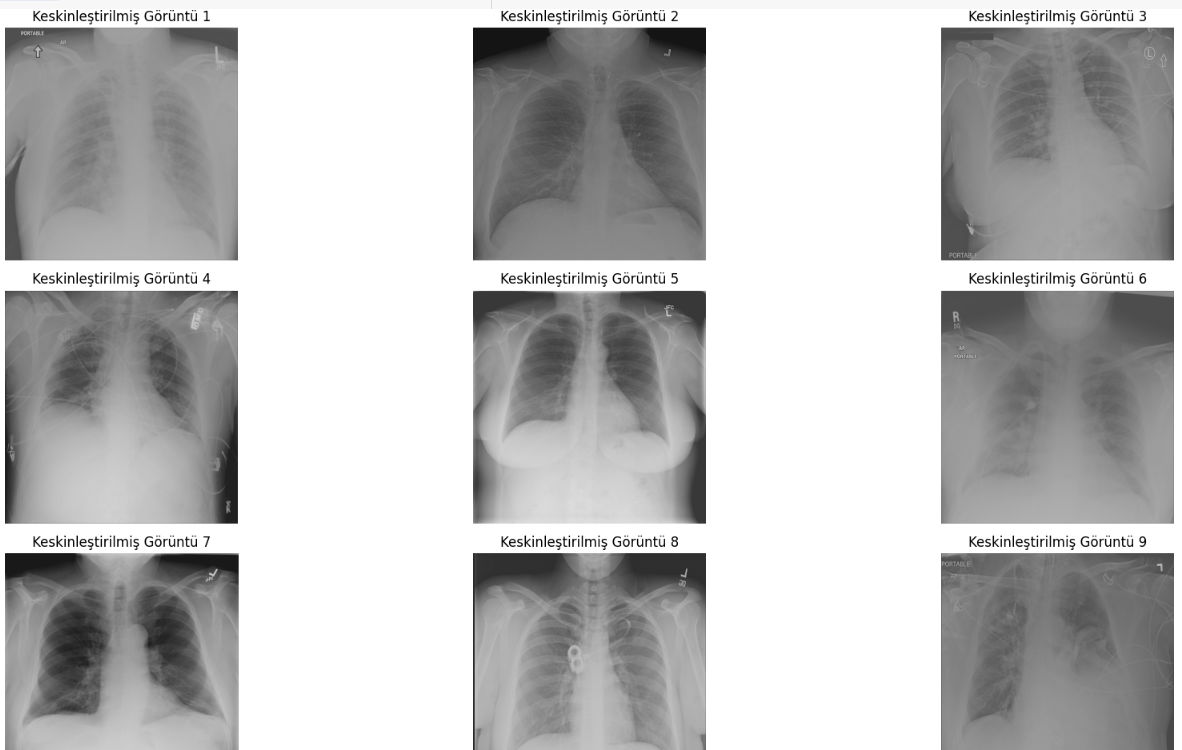
   plt.imshow(resized, cmap='gray')

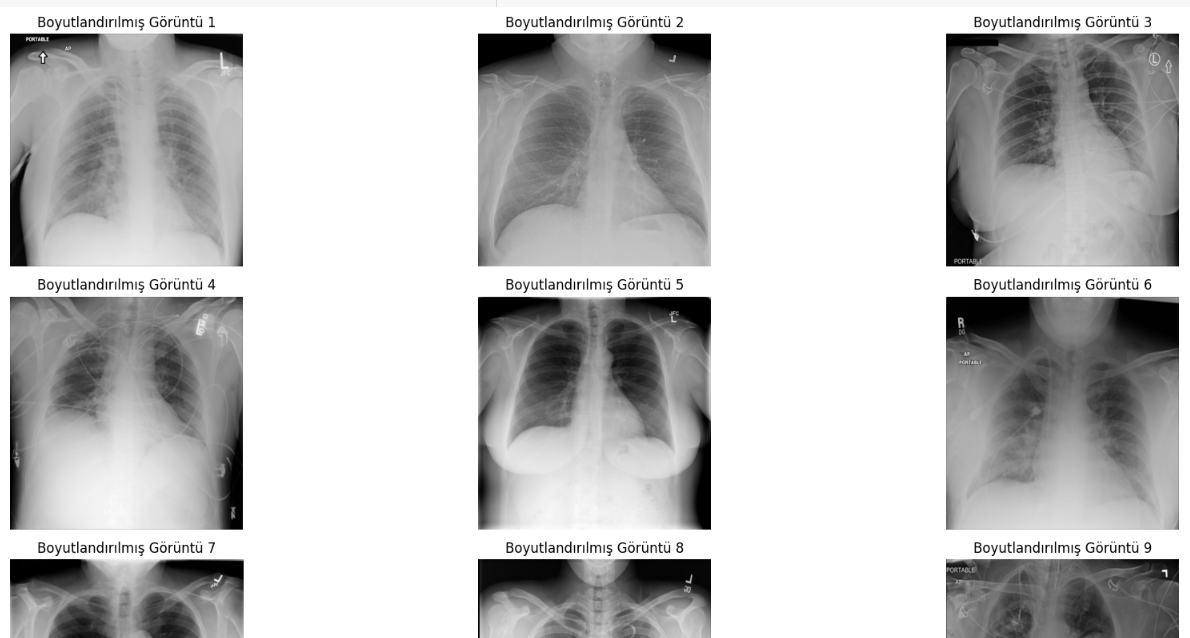
   plt.title(f'Boyutlandırılmış Görüntü {i+1}')

   plt.axis('off')

plt.tight\_layout()

plt.show()





1. Keskinleştirme sonuçları (üst resim):

- Akciğer dokusu ve kemik detayları daha belirgin

- Kenarlar daha net

- Özellikle görüntü 5 ve 8'de iyileşme belirgin

2. Boyutlandırma sonuçları (alt resim):

- Görüntüler 2 kat büyütülmüş

- Detay kaybı minimal

- İnceleme için daha uygun boyut

Her iki işlem de görüntü kalitesini artırmış, ancak keskinleştirme işlemi daha belirgin fayda sağlamış.

# Sonuç ve Değerlendirme Raporu

Bu çalışmada, NIH Clinical Center'ın ChestX-ray8 veri seti üzerinde kapsamlı bir görüntü işleme analizi gerçekleştirilmiştir. 1000 X-ray görüntüsü ve 14 farklı hastalık etiketi içeren veri seti üzerinde çeşitli teknikler uygulanmış ve sonuçları değerlendirilmiştir.

**Veri Analizi;**

- 30,805 hastadan 112,120 göğüs röntgeni

- 1024x1024 piksel çözünürlük

- %7.2'lik hasta tekrar oranı (minimal dengesizlik)

- Eksik veri yok, tutarlı veri tipleri

**Uygulanan Teknikler ve Sonuçları**

***1. Histogram Analizi***

- Piksel dağılımları incelendi

- Kontrast ve parlaklık farklılıkları tespit edildi

- İyileştirme gereken alanlar belirlendi

***2. Görüntü İyileştirme***

- Kontrast Germe: Minimal etki, orijinal görüntüler zaten iyi kontrastlı

- Histogram Eşitleme: En etkili teknik, detaylar belirginleşti

- Gamma Düzeltme: Yumuşak geçişler sağladı

***3. Gürültü Azaltma***

- Median Filtre: Keskin detayları koruyarak gürültüyü azalttı

- Gaussian Filtre: Yumuşak geçişler sağladı

- Her iki filtre de görüntü kalitesini artırdı

***4. Görüntü Dönüşümleri***

- Döndürme ve ayna çevirme teknikleri başarıyla uygulandı

- Veri çoğaltma ve standardizasyon sağlandı

***5. Frekans Alanı İşlemleri***

- Fourier dönüşümü başarıyla uygulandı

- Düşük geçiren filtre ile yumuşatma sağlandı

**Genel Değerlendirme**

1. Teknik Başarı:

- Histogram eşitleme en etkili yöntem

- Gürültü azaltma teknikleri başarılı

- Görüntü standardizasyonu sağlandı

2. Klinik Fayda:

- Detayların görünürlüğü arttı

- Tanı doğruluğuna katkı potansiyeli

- Yapay zeka eğitimi için uygun veri kalitesi

3. Gelecek Çalışmalar:

- Derin öğrenme modelleri için veri hazırlığı

- Otomatik tanı sistemleri geliştirme

- Farklı hastalık grupları için özelleştirilmiş işleme teknikleri

Bu çalışma ile, tıbbi görüntü işleme tekniklerinin X-ray görüntüleri üzerindeki etkisi daha iyi anlaşılmıştır. Teşekkür ederim.

**Aytekin Özel-Üsküdar Üniversitesi Yapay Zeka Yüksek Lisans**