

02 KASIM 2025

# GÖRÜNTÜ İŞLEMEDE CNN YÖNTEMİ VE UYGULAMA RAPORU

GÖRÜNTÜ İŞLEME TEKNİKLERİ ÖDEV-3

ŞEYHMUS ERBEKLER  
YAZILIM MÜHENDİSİ

# CNN Yöntemi ve Görüntüye Uygulanması – Ödev Raporu

Bu rapor; seçilen bir görüntü üzerinde CNN tabanlı segmentasyon ile **Canny**, **Sobel**, **Laplacian** ve **hibrit** kenar çıkarımı uygulamalarını ve uygulama adımlarını ayrıntılı olarak açıklar.

## 1. Giriş

Bu çalışmada bir görüntü üzerinde (image.png) nesne segmentasyonu ve kenar çıkarımı gerçekleştirilmiştir. Segmentasyon için CNN tabanlı YOLOv8-seg modeli, kenar çıkarımı içinse Canny, Sobel ve Laplacian yöntemleri ayrıca bu yöntemlerin hibrit bir bileşimi kullanılmıştır. Rapor kapsamında CNN kavramı, ilgili mimariler ve projenin uygulama adımları detaylandırılmıştır.

## 2. CNN (Convolutional Neural Networks) Nedir?

CNN (Evrşimsel Sinir Ağları), görüntülerden anlamlı öznitelikleri (kenar, köşe, doku) otomatik olarak çıkaran derin öğrenme modelleridir. En temel bileşenleri şöyledir:

- **Convolution (Evrşim) Katmanı:** Küçük çekirdeklerle (kernel) görüntü üzerinde kayarak özellik haritaları üretir.
- **Aktivasyon (ReLU, GELU vb.):** Doğrusal olmayanlık ekleyerek ifade gücünü artırır.
- **Pooling (Maks/Aritmetik):** Boyut azaltır, özniteliklerin daha özet ve gürültüye dayanıklı temsilini sağlar.
- **Normalization (Batch/Layer):** Eğitimde kararlılığı artırır.
- **Full-Connected / Head:** Sınıflandırma, tespit veya segmentasyon amacına uygun çıktı üretir.

### 2.1 CNN Tabanlı Görüntü Görevleri

- **Sınıflandırma:** Görüntünün tek bir etikete atanması. (Örn. kedi/kopek)
- **Nesne Tespiti (Detection):** Nesnenin konumu (bbox) + sınıf etiketinin bulunması.
- **Semantik/Örnek Segmentasyon:** Her pikselin sınıflandırılması ve nesne maskelerinin üretilmesi.

### 2.2 Öne Çıkan Mimariler

- **U-Net / DeepLab:** Encoder-decoder yapısıyla piksel düzeyi maskeler üretir (özellikle tıpta yaygın).
- **Mask R-CNN:** Bölge tabanlı tespit + piksel bazlı maske üretimi.
- **YOLOv8-seg:** Gerçek zamanlı tespit ve segmentasyonu aynı mimaride birleştirir (bu çalışmada kullanıldı).

### 3. Bu Çalışmada CNN Nasıl Kullanıldı?

Bu projede, hazır eğitilmiş YOLOv8n-seg ağı (Ultralytics) kullanılmıştır. Model, COCO veri kümesindeki sınıfları (person, cat, dog, car vb.) tanıyacak şekilde eğitilmiştir. Seçilen görüntü (image.png) modelin girişine verildiğinde, model her bir nesne için:

- Sınıf etiketi (ör. person, cat, dog, car)
- Sınır kutusu (bounding box) koordinatları
- Piksel bazlı nesne maskesi (segmentasyon)

Modelin ürettiği maske, orijinal görüntü boyutuna ölçeklenip yarı saydam olarak görüntü üstüne bindirilmiş; ayrıca konturlar ve bbox ile etiketler görselleştirilmiştir.

### 4. Kenar Çıkarımı Yöntemleri

#### 4.1 Canny

Canny, görüntü azaltımı (Gaussian blur), gradyan hesaplama, non-maximum suppression ve çift eşikleme adımlarını içerir. Otomatik eşik yaklaşımı (medyan temelli) ile görüntüye adaptif biçimde uygulanmıştır.

#### 4.2 Sobel

Sobel operatörü, x ve y doğrultularındaki gradyanları hesaplar. Bu çalışmada Sobel büyüklük haritası normalize edilerek kenar şiddetleri görselleştirilmiştir.

#### 4.3 Laplacian

Laplacian ikinci türev tabanlı bir yöntemdir; yön bağımsız olarak ani yoğunluk değişimlerini vurgular ve yüksek frekans bileşenlerine duyarlıdır.

#### 4.4 Hibrit Yaklaşım

Sobel büyüklük haritası Otsu ile ikilendirilip düşük eşikli Canny çıktısı ile OR işlemine tabi tutulmuş, son olarak morfolojik açma ile görüntü temizliği yapılmıştır. Böylece Canny'nin keskin kenarları ile Sobel'in yönsel avantajları tek bir haritada birleştirilmiştir.

### 5. Proje Kurulumu ve Çalışma Adımları

1. PyCharm'da Odev-3 projesini oluşturdum.
2. Gerekli paketleri kurulum: pip install ultralytics opencv-python numpy matplotlib
3. Görselimiz ; Odev-3/images/ içine image.png adıyla kopyalandı
4. segment\_edge\_cnn.py dosyası oluşturuldu ve gerekli kodlar yazıldı
5. Rapor oluşturma Scripti oluşturuldu: python segment\_edge\_cnn.py
6. İlk çalıştırmada YOLOv8n-seg ağırlığı indirilecektir (internet gerekli).
7. Uygulamayı çalıştırdığımızda proje klasöründeki outputs içine 7 farklı işlenmiş görsel düşüyor ve rapor oluşuyor.

## 6. Klasör Yapısı ve Çıktılar

Odev-3/

```
├── images/
│   └── image.png
├── outputs/
│   ├── 1_segmentation_overlay.png
│   ├── 2_segmentation_contours.png
│   ├── 3_edges_canny.png
│   ├── 4_edges_sobel_magnitude.png
│   ├── 5_edges_laplacian.png
│   ├── 6_edges_hybrid.png
│   ├── 7_edges_overlay_color.png
│   ├── README_CNN_NOTES.txt
│   ├── DETECTIONS.txt
│   └── rapor.pdf (make_report.py ile oluşturulur)
├── segment_edge_cnn.py
└── make_report.py
```

- 1–2 numaralı görseller segmentasyonun görsel çıktılarıdır (maske bindirme ve kontur/bbox).
- 3–6 numaralı görseller çeşitli kenar çıkarma sonuçlarını göstermektedir.
- 7 numaralı görselde üç yöntem renk kodlu olarak aynı resim üzerinde üst üste gösterilmiştir.
- README\_CNN\_NOTES.txt ve DETECTIONS.txt rapora metin olarak eklenebilir.
- make\_report.py çalıştırılarak outputs/rapor.pdf otomatik üretilir.

## 7. Sonuç ve Öneriler

CNN tabanlı YOLOv8-seg ile nesnelerin piksel düzeyi maskeleri başarıyla elde edilmiş; Canny, Sobel, Laplacian ve hibrit yöntemlerle kenarlar karşılaştırmalı biçimde çıkarılmıştır. İleride U-Net veya DeepLab gibi mimarilerle özel amaçlı (ör. tıbbi görüntü) veri üzerinde yeniden eğitim yapılarak sonuçlar daha da iyileştirilebilir.

## 8. Kaynakça

- [1] Ultralytics YOLOv8 Documentation, <https://docs.ultralytics.com/>
- [2] OpenCV Documentation – Edge Detection, <https://docs.opencv.org/>
- [3] Ronneberger et al., "U-Net: Convolutional Networks for Biomedical Image Segmentation", MICCAI, 2015.