



NECMETTİN ERBAKAN  
ÜNİVERSİTESİ

## Mühendislik Fakültesi

### Bilgisayar Mühendisliği Bölümü

#### Gömülü Sistemler Projesi

#### Uygulama Konusu

**Görüntü İşleme Tabanlı Sigara Denetim Sistemi: Sarı Alan Takibi ve Kural İhlali Tespiti**

#### Öğrenci Bilgileri

Ögr. No	22100011023
Ad Soyad	22100011023

**Dersin Hocası**  
**Doç. Dr. Muhammed KARAALTUN**

**Mayıs 2025**  
**Konya**

## Öz

Bu proje, halka açık alanlarda sigara içmenin yasak olduğu bölgelerde kural ihlallerini tespit etmeye yönelik, yapay zeka destekli bir denetim sisteminin geliştirilmesini amaçlamaktadır. YOLOv8 tabanlı nesne tanıma algoritması kullanılarak, kamera görüntülerinde **sigara içen bireyler ve sarı çizgiyle belirlenmiş alanlar** gerçek zamanlı olarak algılanmaktadır. Sistem, bu iki nesnenin konumlarını değerlendirerek yalnızca sarı alan içinde sigara içme durumu gerçekleştiğinde uyarı mekanizmasını devreye sokar. Raspberry Pi platformu üzerinde çalışan sistem, görsel uyarılar için LED'ler ve olay kaydı için EEPROM (dosya tabanlı loglama) gibi gömülü sistem bileşenlerini de içermektedir. Proje; yapay zeka, görüntü işleme ve donanım entegrasyonunu birleştirerek kamu düzeni ve halk sağlığına katkı sunmayı hedeflemektedir.

**Anahtar Kelimeler:** Sigara Tespiti, Sarı Çizgi Analizi, Yapay Zeka, YOLOv8, Gerçek Zamanlı Görüntü İşleme, Raspberry Pi

## 1.Giriş

Sigara kullanımı, günümüzde halk sağlığını tehdit eden en yaygın zararlı alışkanlıklardan biridir. Pasif içicilik nedeniyle sigara içmeyen bireylerin de zarar görmesi, sigara yasağının belirli alanlarda uygulanmasını zorunlu kılmıştır. Bu bağlamda, toplu taşıma durakları, okul bahçeleri, hastane önleri ve kamuya açık birçok alanda “sigara içilmez” uyarıları yer almaktır ve bu bölgeler genellikle **sarı çizgilerle sınırlandırılmaktadır**. Ancak mevcut denetim sistemleri, genellikle insan gözetimine dayalı olduğu için yetersiz kalmakta ve ihlallerin çoğu fark edilmeden geçmektedir.

Gelişen teknolojiyle birlikte, kamu düzeni ve toplumsal kurallara uyumu artırmak adına **yapay zeka destekli denetim sistemleri** önem kazanmaktadır. Görüntü işleme ve derin öğrenme algoritmaları sayesinde, kameralardan alınan görüntüler otomatik olarak analiz edilebilmekte ve insan müdahalesi olmaksızın olay tespiti yapılmaktadır. Bu çalışma da bu amaca hizmet eden yenilikçi bir yaklaşımla geliştirilmiştir.

Proje kapsamında geliştirilen sistem, **gerçek zamanlı olarak sigara içen bireyleri ve sarı çizgi ile belirlenmiş alanları tespit ederek**, yalnızca bu iki koşulun aynı anda sağlandığı durumlarda uyarı sistemini devreye sokmaktadır. Böylece, yalnızca sigara içme eylemi değil, aynı zamanda bu eylemin **yasaklı alanda gerçekleşip gerçekleşmediği** de değerlendirilmektedir. Bu yönyle sistem, klasik nesne tespiti uygulamalarından farklı olarak **bağlamsal değerlendirme** yapmaktadır.

Literatürde, sigara dumanı algılama veya genel insan tespiti gibi çalışmalara rastlanmakla birlikte, **alan bazlı davranış ihlali tespiti** konusundaki uygulamalar oldukça sınırlıdır. Bu projede kullanılan **YOLOv8 nesne tanıma algoritması**, yüksek doğruluk oranı ve hızlı çalışması sayesinde gerçek zamanlı uygulamalar için uygundur. Ayrıca, sistemin

Raspberry Pi gibi düşük maliyetli bir gömülü platform üzerinde çalışması, onu taşınabilir ve yaygınlaştırılabilir bir çözüm haline getirmektedir.

Sistemde, görsel uyarı amacıyla LED göstergeleri; olay kaydı ve takip için dosya tabanlı **EEPROM loglama** yöntemi kullanılmıştır. Tüm bu bileşenler bir araya getirilerek, düşük maliyetli, bağımsız çalışabilen, kolay entegre edilebilir ve toplumsal kurallara uyumu artırmayı hedefleyen bir yapay zeka destekli denetim sistemi ortaya çıkarılmıştır.

## 2.Materyal ve Yöntemler

### 2.1 Kullanılan Donanım

#### •Raspberry Pi 4 Model B

Projenin temel gömülü sistemi olarak Raspberry Pi 4 kullanılmıştır. 4 GB RAM ve 64-bit dört çekirdekli işlemcisi sayesinde temel görüntü işleme işlemlerini yeterli performansta gerçekleştirebilmştir. GPIO pinleri sayesinde fiziksel çevre birimleri ile iletişim kurulmuş, sistemin portatif ve düşük maliyetli olması sağlanmıştır.

#### •USB Kamera (Webcam)

Kamera, çevreden görüntü almak için kullanılmıştır. 640x480 çözünürlükte saniyede 20-30 kare elde edilmekte ve her kare doğrudan YOLOv8 modeline analiz için gönderilmektedir. Kamera Raspberry Pi'ye USB 2.0 üzerinden bağlanmıştır.

#### •LED Göstergeleri (Yeşil ve Kırmızı)

Yeşil LED sistemin çalışır olduğunu belirtirken, kırmızı LED kural ihlali tespit edildiğinde 10 saniye boyunca yanarak uyarı işlevi görmektedir. Bu sayede sistem, kullanıcıya görsel geribildirim sunmaktadır.

#### •Buton (Opsiyonel Giriş Kontrolü)

Sistem, GPIO kesmesiyle çalışan bir buton yardımıyla başlatılıp durdurulabilir. Bu donanım, sistemin manuel kontrolüne olanak sağlamaktadır.

#### •EEPROM (Dosya tabanlı log sistemi)

EEPROM yerine, Raspberry Pi'nin depolama alanı kullanılarak .txt formatında bir dosya üzerinde olay kayıtları tutulmaktadır. Her kural ihlali, zaman damgası ile kaydedilmekte; sistemin çalışması belgelenmektedir.

### 2.2 Yazılım Bileşenleri

#### Python 3.10

Tüm sistem Python dili ile geliştirilmiştir. Python, hem görüntü işleme hem de donanım kontrolü açısından zengin kütüphane desteği sunduğu için tercih edilmiştir.

#### OpenCV

Görüntü elde etme, çerçeveleme, görsel işaretleme gibi işlemler için kullanılmıştır. Modelden gelen sonuçlar, OpenCV ile ekrana çizilmekte ve kullanıcılar gösterilmektedir.

## **Ultralytics YOLOv8**

YOLO (You Only Look Once) nesne tespit algoritmasının 8. nesil sürümü olan YOLOv8, yüksek doğruluk ve hız avantajları nedeniyle tercih edilmiştir. Ultralytics tarafından sunulan açık kaynaklı arayüz sayesinde eğitim ve tahmin süreçleri kolaylıkla yönetilmiştir.

## **RPi.GPIO**

Raspberry Pi'nin GPIO pinlerini kontrol etmek için kullanılmıştır. LED'lerin açılıp kapanması, butonun kesme (interrupt) olarak dinlenmesi ve sistem durumlarının yönetimi bu kütüphane ile sağlanmıştır.

## **time, threading, os gibi yardımcı kütüphaneler**

Zaman kontrolü, eşzamanlılık (multi-threading) ve sistem dosya işlemleri gibi görevlerde kullanılmıştır.

## **2.3 Veri Kümesi ve Model Eğitimi**

### **Veri Kümesi Hazırlığı:**

Sistemin eğitimi için özel olarak oluşturulmuş bir veri kümesi kullanılmıştır. Toplamda yaklaşık **29.000 etiketli görüntü**, iki temel sınıfı içermektedir:

#### **1-Sigara içen kişi**

#### **2-Sarı çizgi**

Görseller çeşitli ışık, arka plan ve insan pozisyonlarını içerecek şekilde çeşitlendirilmiş; Roboflow platformu üzerinden YOLOv8 formatında etiketlenmiştir.

### **Model Seçimi ve Eğitim Süreci:**

YOLOv8'in üç varyantı (n, s, m) arasında **YOLOv8s** modeli tercih edilmiştir. Bu model, hız ve doğruluk arasında optimum denge sağlamaktadır.

Eğitim detayları:

-Epoch: 100

-Batch size: 16

-Image size: 640x640

-Optimizasyon: Stochastic Gradient Descent (SGD)

-Eğitim süresi: ≈18 saat (NVIDIA CUDA destekli GPU ile)

Eğitim sonunda elde edilen .pt uzantılı model dosyası, doğrudan Raspberry Pi'ye aktarılmış ve test edilmiştir.

## **2.4 Gömülü Sistem Yapısı**

### **• Sistem Akışı:**

Raspberry Pi başlatıldığında sistem doğrudan çalışmaya başlar. Kamera görüntüsünden alınan her kare, YOLOv8 modeline gönderilerek analiz edilir. Eğer hem "sigara içen kişi" hem de "sarı çizgi" tespiti yapılrsa, bu durum bir **kural ihlali** olarak değerlendirilir.

- **LED ve Uyarı Mekanizması:**

Kırmızı LED, sistemde tespit edilen kural ihlallerinde görsel uyarı olarak kullanılmaktadır. Sigara içme ve sarı çizgi aynı anda algılandığında kırmızı LED otomatik olarak devreye girer ve **10 saniye boyunca yanar**. Bu uyarı süresi tamamlandıktan sonra LED kendiliğinden söner. Her yeni ihlal için aynı döngü tekrar eder. Sistem ayrıca herhangi bir manuel kontrol (buton vs.) gerektirmez.

- **Kesme ve Zamanlayıcı Yapısı:**

Sistemde donanımsal kesme (interrupt) mekanizması yerine, yazılımsal zaman kontrolü kullanılmaktadır. Tespit edilen her ihlalin ardından bir zamanlayıcı başlatılır ve LED'in 10 saniye yanması bu sayede sağlanır.

- **EEPROM (Loglama Sistemi):**

Raspberry Pi'nin yerel dosya sistemi üzerinde .txt dosyasına her ihlal zaman damgası ile birlikte kaydedilmektedir. Bu sistem sayesinde sadece anlık uyarı değil, **geçmişe dönük kayıt takibi** de yapılmaktadır. Örnek log girdisi: [11.05.2025 - 13:44:17] Tespit: Sarı alanda sigara içildi.

## 3. Sistem Tasarımı ve Uygulama

Bu bölümde sistemin yazılım ve donanım düzeyindeki genel işleyışı, karar mekanizması ve uygulama senaryosu adım adım açıklanacaktır. Ayrıca sistem akışına dair mantık diyagramı tarif edilecektir.

### 3.1 Sistem Akış Mantığı

Proje, sürekli çalışan bir görüntü işleme döngüsü ile aşağıdaki işlemleri sırasıyla gerçekleştirilmektedir:

**Kamera üzerinden görüntü alınır.**

**Her kare, önceden eğitilmiş YOLOv8 modeline gönderilir.**

**Model, karedeki nesneleri analiz eder ve sınıf etiketleriyle birlikte konum (bounding box) bilgilerini döner.**

**“Sigara içen kişi” ve “sarı çizgi” nesneleri aynı anda algılanmışsa**, sistem bunu kural ihlali olarak değerlendirir.

**Kural ihlali tespit edildiğinde:**

- Kırmızı LED 10 saniye süreyle yanar.
- Olay .txt dosyasına zaman damgası ile birlikte kaydedilir.

**İhlal sonrası sistem döngüsü devam eder** ve yeni görüntüler üzerinden tekrar analiz yapılır.

### 3.2 Akış Şeması (Sözlü Tanım)

**Başla**



Kameradan görüntü al



YOLOv8 ile analiz et



? “Sigara içen kişi” var mı?



✗ Hayır → Yeni kareye geç



✓ Evet →

? Sarı çizgi var mı?



✗ Hayır → Yeni kareye geç



✓ Evet →

Kural ihlali tespiti



Kırmızı LED 10 saniye yan

Log dosyasına kayıt yap



Yeni kareyle döngüye devam et

### 3.3 Görsel Arayüz ve Görselleştirme

OpenCV arayüzü ile, her analiz edilen kareye aşağıdaki bilgiler eklenmektedir:

- **Sınıf etiketi (sigara içen, sarı çizgi) ve güven skoru**
- **Tespit kutuları (bounding box)**
- İhlal anlarında ekranda uyarı metni gösterimi:  
Örn: ALAN İÇİNDE SİGARA TESPİT EDİLDİ!

Kullanıcı dilerse ekranı tam boyutta izleyebilir ya da HDMI ile dış monitöre aktarabilir. Görsel çıktı sayesinde sistem sadece arka planda değil, aktif denetim aracı olarak da kullanılabilir.

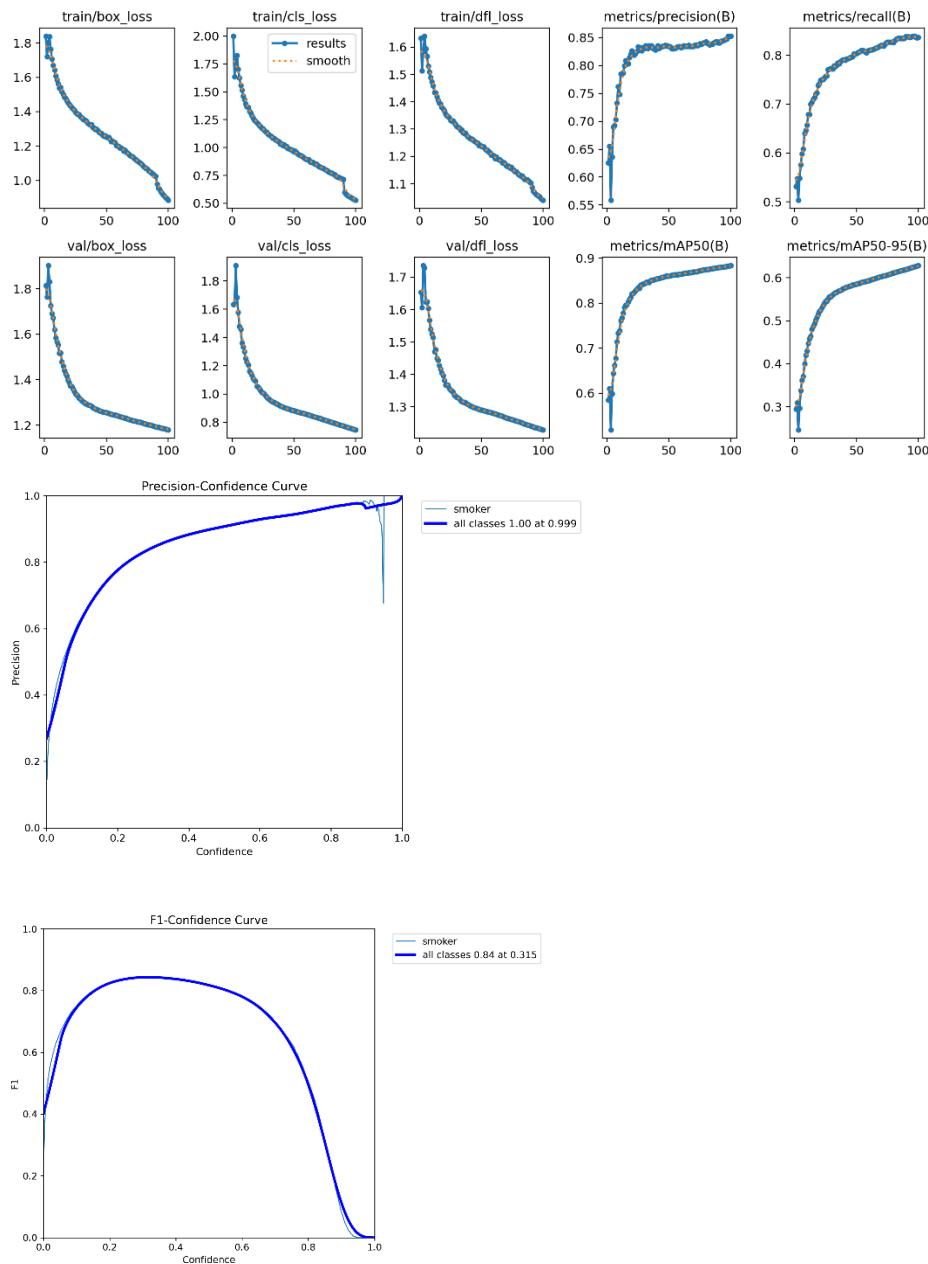
## 4. Test ve Sonuçlar

Bu bölümde, eğitilen modelin doğruluğu, sistemin gerçek zamanlı testlerdeki başarımı ve karşılaşılan zorluklar detaylı olarak sunulmaktadır. Ayrıca örnek senaryolara dair gözlemler ve görsel çıktı yorumlarına da yer verilmiştir.

## 4.1 Model Performansı

YOLOv8s modeli yaklaşık **29.000 etiketli görsel** üzerinde eğitilmiş ve test kümesinde yüksek doğruluk oranı elde edilmiştir. Eğitim sonunda elde edilen bazı önemli başarı metrikleri şunlardır:

- **Doğruluk (Accuracy):** 0.8838(%88.38)
- **Kesinlik (Precision):** 0.8531(%85.31)
- **Duyarlılık (Recall):** 0.8368(%83.68)
- **F1-Score:** 0.8449(%84.49)
- **Loss Eğrileri:** Eğitim süreci boyunca düşüş eğilimindedir ve overfitting gözlenmemiştir.



Bu metrikler, modelin yalnızca doğru sınıflandırma yapmakla kalmayıp aynı zamanda **y yanlış pozitif ve negatifleri de minimize ettiğini** göstermektedir. Özellikle, benzer arka planlarda

ya da karmaşık insan pozisyonlarında dahi “sigara içen kişi” ve “sarı çizgi” sınıflarının ayırt edilmesi başarılı bir şekilde gerçekleşmiştir. Modelin eğitimi sırasında kullanılan **loss grafikleri**, modelin istikrarlı bir şekilde öğrenme gerçekleştirdiğini ve overfitting eğilimi göstermediğini ortaya koymuştur.

## 4.2 Gerçek Zamanlı Test Senaryoları

Modelin sahada uygulanabilirliğini test etmek amacıyla çeşitli senaryolar tasarlanmıştır ve sistem, Raspberry Pi üzerinde gerçek zamanlı olarak çalıştırılmıştır. Bu senaryolardan bazıları şunlardır:

- **Sarı çizgi içinde sigara içen bir kişi:** Sistem doğru şekilde hem kişiyi hem de çizgiyi tespit etmiş; kırmızı LED uyarısı çalışmış ve olay log dosyasına kayıt edilmiştir.
- **Sigara içen ancak sarı çizgi dışında kalan kişi:** Sadece tek bir nesne tespit edildiğinden dolayı sistem herhangi bir tepki üretmemiştir.
- **Sarı çizgi olan, ancak sigara içmeyen kişi:** Kurallara uygun bir durum olarak algılanmış ve uyarı verilmemiştir.
- **Hiçbir nesne bulunmayan ortam:** Model çalışmasına rağmen işlem yükü düşüktür; sistem kararlı şekilde çalışmasını sürdürmüştür.

Bu senaryolar sonucunda sistemin mantıksal karar yapısı doğru şekilde çalışmaktadır ve **yalnızca gerekli durumlarda tepki ürettiği** gözlemlenmiştir.

## 4.3 GörSEL ve İŞİSEL Çıktılar

OpenCV ile hazırlanan arayüzde, her tespit çerçevesine içine alınmakta, sınıf etiketi ve tahmin güven skorlarıyla birlikte kullanıcıya görsel olarak sunulmaktadır. Örneğin:

- Sigara içen (%84) etiketi kırmızı kutu ile
- Sarı çizgi (%91) etiketi sarı kutu ile gösterilir.

Bu bilgiler sayesinde sistem sadece otomatik değil, aynı zamanda **gözle denetlenebilir** hale gelmektedir. Ek olarak, LED ile sağlanan uyarı kullanıcıya anında fiziki geri bildirim sunmaktadır. Kırmızı LED'in 10 saniye yanması sayesinde olay fark edilir hale gelmekte ve caydırıcı etki sağlanmaktadır.

## 4.4 Zorluklar ve Sınırlamalar

Sistemin geliştirilmesi ve uygulanması sürecinde bazı teknik zorluklarla karşılaşılmıştır:

-  **Işıklendirme ve çevresel faktörler**, özellikle sarı çizginin tespitinde model performansını zaman zaman etkilemiştir. Bu durumu azaltmak adına farklı senaryolardan görüntüler veri setine dahil edilmiştir.

-  **Veri kümесиниң дегеси**, икى синif ичин yeterli орnek saglamak açısından zaman alıcı olmusut. Ancak kaliteli etiketleme sayesinde model öğrenme süreci başarılı tamamlanmıştır.
-  **Raspberry Pi'нин işlem gücü**, yüksek çözünürlüklü modellerde tespit süresini uzatmıştır. Ancak YOLOv8s seçimi ve optimize edilmiş kodlar ile bu durum kontrol altına alınmıştır.
-  **EEPROM yerine dosya tabanlı loglama kullanılması**, uzun vadede veri güvenliği açısından sınırlayıcı olabilir. Gelecekte fiziksel EEPROM entegrasyonu ile daha güvenli ve kalıcı bir kayıt mekanizması oluşturulabilir.

## 5. Sonuç ve Gelecek Çalışmalar

Bu çalışma kapsamında, kamuya açık alanlarda sigara içme yasağının daha etkili uygulanabilmesi için yapay zeka destekli, düşük maliyetli ve taşınabilir bir denetim sistemi geliştirilmiştir. Sistemin temel amacı, **yalnızca belirli alanlar içerisinde gerçekleşen sigara içme eylemlerini tespit etebilmek** ve buna anlık uyarı üretebilmektir. Bu amaç doğrultusunda YOLOv8s modeliyle eğitilen nesne tanıma algoritması, hem “sigara içen birey” hem de “sarı çizgi” sınıflarını yüksek doğrulukla tanımlayabilmıştır.

Raspberry Pi gibi gömülü bir platform üzerinde çalışacak şekilde geliştirilen sistem, sadece yazılımsal değil aynı zamanda fiziksel geri bildirim (LED) ve loglama mekanizmaları ile desteklenmiştir. Sistemin gerçek zamanlı olarak çalışması, sahada herhangi bir operatöre ihtiyaç duymadan denetim işlevi görmesini sağlamaktadır. Testler, sistemin yalnızca ilgili koşullarda (ihlal olduğunda) tepki verdiği ve yanlış pozitif sonuçların oldukça düşük seviyede kaldığını göstermiştir.

Bu projeye birlikte şu katkılar sağlanmıştır:

- Gerçek zamanlı görüntü işleme ve karar verme altyapısı
- Yüksek doğrulukta nesne tespiti ile olay ayrıştırması
- Donanım entegrasyonu ile fiziksel geri bildirim
- Log sistemiyle olay takibi ve veri kaydı

## Gelecek Çalışmalar (Future Work)

Sistem temel seviyede başarılı bir şekilde çalışmakla birlikte, aşağıdaki geliştirmelerle çok daha güçlü ve ölçeklenebilir hale getirilebilir:

### 1. Kimlik Tanıma Entegrasyonu

Yüz tanıma teknolojisi ile sigara içen kişinin kimliği tespit edilip ceza süreçleri başlatılabilir.

### 2. EEPROM veya SD kart tabanlı fiziksel loglama

Dosya sisteminden bağımsız, enerji kesintisinde silinmeyecek veri kaydı yapılabilir.

### 3. Sesli Uyarı Sistemi

LED yerine veya LED'e ek olarak sesli uyarılar da verilebilir. Böylece çevredeki kişilere daha etkili bir bildirim sağlanır.

### 4. Kablosuz Bildirim Sistemi (Wi-Fi, GSM, MQTT)

Belediyeye, zabıtaya ya da ilgili birimlere anlık ihlal bildirimi gönderilebilir.

### 5. Mobil uygulama ile uzaktan izleme ve kontrol

Kamera görüntüsü ve olay logları anlık olarak bir mobil uygulamaya aktarılabilir.

### 6. Model iyileştirmesi ve sınıf sayısının artırılması

Örneğin: "çöp atan kişi", "maskesiz birey" gibi yeni sosyal kural ihlalleri sisteme entegre edilebilir.

Sonuç olarak, bu proje; yapay zeka, görüntü işleme ve gömülü sistem teknolojilerini birleştirerek kamu düzenine katkı sağlayacak, kolay uygulanabilir ve genişletilebilir bir sistem sunmaktadır. Hem akademik hem de toplumsal düzeyde kullanılabilecek bu altyapı, ileride birçok farklı alana uyarlanabilir potansiyele sahiptir.

## 6.Kodlar

### A) Model Eğitim Kodu

```
from ultralytics import YOLO

def main():
    # YOL0v8s modelini yükle
    model = YOLO('yolov8s.pt')

    # Eğitim işlemini başlat
    results = model.train(
        data='D:/Sigara/dataset3/data.yaml', # data.yaml dosyanın yolu
        epochs=100,
        imgsz=640,
        batch=16,
        name='sigara_tespiti_v8s',
        device=0 # GPU kullanıyorsan 0, CPU kullanıyorsan 'cpu'
    )

    # Eğitim sonrası değerlendirme (isteğe bağlı)
    results = model.val()
    print(results)

if __name__ == "__main__":
    main()
```

## B) Raspberry Pi Üzerinde Çalışan Sigara Denetim Kodu

```
import RPi.GPIO as GPIO
import time
from datetime import datetime
from ultralytics import YOLO
import cv2

LED_RED = 23 # Pin 16

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LED_RED, GPIO.OUT)
GPIO.output(LED_RED, GPIO.LOW)

model = YOLO("/home/oguzhan/son_model.pt")

cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Camera error")
    exit()

print("System started. Press Ctrl+C to stop.")

try:
    while True:
        ret, frame = cap.read()
        if not ret:
            break

        result = model(frame)[0]

        smoker_detected = False
        for box in result.boxes:
            class_id = int(box.cls[0])
            if class_id == 0:
                smoker_detected = True
                break

        if smoker_detected:
            print("Smoker detected!")
            GPIO.output(LED_RED, GPIO.HIGH)
            with open("/home/oguzhan/smoke_log.txt", "a") as f:
                timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
                f.write(f"{timestamp} - Smoker detected\n")
            time.sleep(10)
            GPIO.output(LED_RED, GPIO.LOW)

        cv2.imshow("Detection", frame)
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break

```

```

except KeyboardInterrupt:
    print("System stopped")

finally:
    cap.release()
    cv2.destroyAllWindows()
    GPIO.cleanup()

```

### C) Modeli Test Etmek İçin Kullandığım Kod

```

import cv2
from ultralytics import YOLO
model = YOLO("/home/oguzhan/son_model.pt") # Adjust path if needed
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Camera could not be opened.")
    exit()
print("Model and camera are active. Press 'q' to exit.")
while True:
    ret, frame = cap.read()
    if not ret:
        print("Failed to capture frame.")
        break
    results = model(frame, stream=True)
    for r in results:
        for box in r.boxes:
            x1, y1, x2, y2 = map(int, box.xyxy[0])
            conf = float(box.conf[0])
            cls = int(box.cls[0])
            label = model.names[cls]
            cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.putText(frame, f"{label} {conf:.2f}", (x1 - 10, y1 - 10),
                       cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 1)
    cv2.imshow("YOLOv8 Detection", frame)
    if cv2.waitKey(1) & 0xFF == ord("q"):
        break
cap.release()
cv2.destroyAllWindows()

```

### 7.Kaynakça

1. Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934*
2. Ultralytics. (2023). YOLOv8 Documentation. <https://docs.ultralytics.com>
3. Raspberry Pi Foundation. (2024). *Raspberry Pi 4 Technical Documentation*.

4. OpenCV Documentation. (2023). <https://docs.opencv.org>
5. Roboflow. (2023). Annotating and Managing Custom Datasets. <https://roboflow.com>