**logo, metin, kayak yapma, grafik tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Engineer Hub Programı**

**YOLOv8-L Modeli Performans Analizi Raporu**

**--**

**METEHAN AYHAN**

**Ocak**

**2025**

**YOLOv8-L MODELİ PERFORMANS ANALİZİ**

**İçindekiler**

1. [Giriş](#giri%C5%9F)
2. [Metodoloji](#metodoloji)  
   2.1. [Model](#model)  
   2.2. [Veri Seti](#veri-seti)  
   2.3. [Değerlendirme Yöntemi](#de%C4%9Ferlendirme-y%C3%B6ntemi)
3. [Metrik Tanımları](#metrik-tan%C4%B1mlar%C4%B1)
4. [Sonuçlar ve Değerlendirme](#sonu%C3%A7lar-ve-de%C4%9Ferlendirme)  
   4.1. [Doğrulama (Val) Metrikleri](#do%C4%9Frulama-val-metrikleri)  
   4.2. [Tüm Resimlerde Inference Hızı (FPS)](#t%C3%BCm-resimlerde-inference-h%C4%B1z%)
5. [GPU Kullanımı ve Güç Ölçümü](#gpu-kullan%C4%B1m%C4%B1-ve-g%C3%BC%C3%A)  
   5.1. [Nasıl Ölçüldü?](#nas%C4%B1l-%C3%B6l%C3%A7%C3%BCld%C3%BC)  
   5.2. [Ölçüm Sonuçları](#%C3%B6l%C3%A7%C3%BCm-sonu%C3%A7lar%C4%B)
6. [Genel Değerlendirme ve Öneriler](#genel-de%C4%9Ferlendirme-ve-%C3%B6neril)
7. [Not](#not)
8. Sonuç
9. Kaynakça

**1. Giriş**

Bu raporda, YOLOv8-L (Large) modelinin insan (person) tespiti odaklı bir veri seti üzerinde elde ettiği doğruluk metrikleri (precision, recall, mAP) ve çıkarım (inference) hızı (FPS, latency) analiz edilmektedir. İncelenen veri setinde, 4370 görsel içerisinde toplam 127639 adet insan etiketi (person instance) bulunmakta olup, modelin hem doğruluk hem de hız performansı detaylı şekilde değerlendirilmektedir.

**2. Metodoloji**

**2.1. Model**

* **Kullanılan model:** Ultralytics tarafından sağlanan ön eğitimli YOLOv8-L (Large) .pt dosyası
* **Modelin boyutu (parametre sayısı):** 43.668.288
* **Yaklaşık hesaplama yükü:** 165.2 GFLOPs

**2.2. Veri Seti**

* **Görüntü Sayısı (val set):** 4370 adet
* **Toplam Nesne (person) Etiketi:** 127639
* **Klasör Yapısı:**
  + /images/val ve /labels/val
  + Etiketler, YOLO formatıyla uyumlu dosyalar biçimindedir.

**2.3. Değerlendirme Yöntemi**

1. **Doğrulama (val) Aşaması:**
   * model.val(...) komutu kullanılarak mAP, precision, recall gibi temel metrikler otomatik hesaplandı.
2. **Hız (FPS) Ölçümü:**
   * Val setindeki 4370 görüntü, tek tek model.predict(...) fonksiyonundan geçirilerek zaman ölçümleri toplandı.
   * Böylece “toplam süre” ve “FPS” (saniyedeki kare sayısı) metrikleri belirlendi.

**3. Metrik Tanımları**

* **Precision (P):** Modelin ürettiği pozitif tespitlerin ne kadarının gerçekten doğru olduğu
* **Recall (R):** Gerçek nesnelerin ne kadarının model tarafından tespit edildiği
* **mAP@0.5:** IoU (Intersection over Union) eşiğinin 0.5 olarak alındığı durumda hesaplanan Ortalama Ortalama Hassasiyet (Mean Average Precision) değeri. Nesne tespiti çalışmalarında en yaygın temel performans göstergelerinden biridir.
* **mAP@0.5:0.95:** IoU eşiğinin 0.5’ten 0.95’e kadar (0.05 adımlarla) değiştiği durumlarda elde edilen AP değerlerinin ortalamasıdır. Daha katı bir ölçüm olup değeri genellikle mAP@0.5’ten daha düşüktür.
* **FPS (Frames per Second):** Bir saniyede işlenebilen ortalama kare (görüntü) sayısını ifade eder. Çıkarım hızının göstergesidir.
* **Latency (ms/görüntü):** Tek bir görüntünün işlenmesi için gereken ortalama süre olup, FPS’in tersi olarak da yorumlanabilir.

**Model Çıktıları:**

**1. Görsel için Gerçek Değerler:**

metin, ekran görüntüsü, Majorelle Blue, kişi, şahıs içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. **Görsel için Tahmin Edilen Değer:**

metin, ekran görüntüsü, alkolsüz içki, Majorelle Blue içeren bir resim

Açıklama otomatik olarak oluşturuldu

**-------------------------------------------------------------------------------------------------------------------------**

1. **Görsel için Gerçek Değerler:**

metin, Majorelle Blue, dükkan, atölye, perakende içeren bir resim

Açıklama otomatik olarak oluşturuldu

**2. Görsel için Tahmin Edilen Değerler:**

metin, ekran görüntüsü, Majorelle Blue, dükkan, atölye içeren bir resim

Açıklama otomatik olarak oluşturuldu

**-------------------------------------------------------------------------------------------------------------------------**

**4. Sonuçlar ve Değerlendirme**

**4.1. Doğrulama (Val) Metrikleri**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Class | Images | Instances | P | R | mAP50 | mAP50-95 |
| all | 4370 | 127639 | 0.752 | 0.517 | 0.629 | 0.408 |
| person | 4370 | 127639 | 0.752 | 0.517 | 0.629 | 0.408 |

**Precision (P):** 0.752 (~%75.2)  
 Modelin tespit ettiği kişilerin yaklaşık üçte ikisinden fazlası doğru çıkmaktadır. Bu, “yanlış pozitif” oranının görece düşük olduğunu gösterir.

**Recall (R):** 0.517 (~%51.7)  
Gerçek kişilerin yaklaşık yarısından biraz fazlası tespit edilmektedir. Özellikle kalabalık ve üst üste (occlusion) durumların yoğun olduğu veri setlerinde bu değer makul görülebilir.

**mAP@0.5:** 0.629 (~%62.9)  
Ortalama hassasiyet değerinin %63 civarında oluşu, modelin kutu konumlandırması ve tespit önceliği bakımından orta-üst seviye bir performans sunduğunu gösterir.

**mAP@0.5:0.95:** 0.408 (~%40.8)  
Daha katı bir ölçüt olan 0.5–0.95 IoU aralığında elde edilen ortalama hassasiyet değeridir. IoU eşiği sıkılaştıkça beklenen başarı seviyesinin düşmesi normaldir.

**Inference Hızı (Val Süreci)**

* **Speed:** 0.1 ms preprocess, 10.8 ms inference, 0.3 ms postprocess / resim  
  Val aşamasında, her resim başına ortalama ~11.2 ms sürede işlem yapılmıştır. Bu değer, validation pipeline içindeki (DataLoader, augmentations, batch işleme vb.) ortalama zaman kullanımını temsil eder.

metin, diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, diyagram, çizgi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, diyagram, ekran görüntüsü, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**4.2. Tüm Resimlerde Inference Hızı (FPS)**

4370 görsele (batch=1, tek tek) yapılan tahmin döngüsü sonunda, toplam süre **86.26** saniye olarak ölçülmüş; bunun sonucu ortalama FPS **~50.66** ve latency ise **~20 ms/görüntü** olarak hesaplanmıştır:

Görüntü sayısı: 4370

Toplam süre: 86.26 s

Ortalama FPS: 50.66

Ortalama Latency: 0.020 s/görüntü (~19.74 ms)

* **FPS ~50:** Birçok gerçek zamanlı (30+ FPS) uygulama için yeterli veya oldukça iyi bir hızdır.
* **Latency ~20 ms:** Her bir görüntünün işlenmesi için ortalama 19–20 ms harcanmaktadır. RTX 3070 GPU üzerinde YOLOv8-L seviyesindeki bir model için bu değer oldukça makuldür.

**5. GPU Kullanımı ve Güç Ölçümü**

YOLOv8-L modelinin val seti (4370 görüntü) üzerindeki çıkarım (inference) işlemleri sırasında GPU bellek kullanımı (VRAM) ve güç tüketimi değerleri NVIDIA Management Library (NVML) üzerinden **pynvml** paketi yardımıyla anlık olarak toplanmıştır.

**5.1. Nasıl Ölçüldü?**

1. **Bellek Kullanımı:**
   * Her tahmin adımından sonra nvmlDeviceGetMemoryInfo(handle) fonksiyonu kullanılarak used alanından o anki VRAM kullanımı (byte) olarak ölçülmüştür.
   * Döngü boyunca kaydedilen en yüksek bellek kullanımı (MB cinsinden) raporlanmıştır.
2. **Güç Tüketimi:**
   * Her tahmin adımında nvmlDeviceGetPowerUsage(handle) fonksiyonu ile o anki GPU güç çekişi (mW) okunmuştur.
   * Elde edilen değerlerden ortalama ve maksimum güç tüketimi (Watt) hesaplanmıştır.
3. **Süre ve FPS:**
   * Tüm görüntüler işlenmeden önce ve sonra time.time() komutuyla zaman kaydı alınarak toplam süre (saniye) ölçülmüştür.
   * FPS = ToplamGörüntüSayısı / ToplamSüre formülüyle ortalama kare/saniye değeri hesaplanmıştır.

**5.2. Ölçüm Sonuçları**

=== GPU ÖLÇÜMLERİ ===

|  |  |
| --- | --- |
| Toplam Görüntü Sayısı | 4370 |
| Süre (sn) | 93.65 |
| FPS (ortalama) | 46.66 |

|  |  |
| --- | --- |
| En Yüksek GPU Bellek Kullanımı | 1247.4 MB |
| Ortalama Güç Tüketimi | 124.6 W |
| Maksimum Güç Tüketimi | 139.2 W |

* **En Yüksek GPU Bellek Kullanımı (1247.4 MB):**  
  640×640 çözünürlükte, tek tek resim işlenirken yaklaşık 1.2 GB VRAM kullanımı gözlenmiştir. RTX 3070’in 8 GB belleğine kıyasla bu düşük bir orandır; dolayısıyla bellek darboğazı yaşanmamıştır.
* **Güç Tüketimi:**
  + Ortalama 124.6 W, anlık en yüksek 139.2 W olarak ölçülmüştür.
  + Kısa süreli yüksek tüketim (139.2 W), modelin yoğun GPU işlemlerinde geçici frekans artışlarından kaynaklanabilir.
* **Toplam Süre (93.65 s) ve FPS (~46.66):**
  + 4370 görüntü ~94 saniyede işlenmiş, ortalama 46–47 FPS elde edilmiştir.
  + Gerçek zamanlı uygulamalar için gereken 30 FPS sınırının üzerinde bir değer olduğu görülmektedir.

**6. Genel Değerlendirme**

* **Genel Performans:**  
  YOLOv8-L, kalabalık insan tespiti içeren veri setinde %75 precision, %52 recall ve %63 mAP@0.5 değerleriyle orta-üst seviyede bir başarı sunmaktadır.
* **Hız:**  
  Tüm resimlerde yapılan toplu (batch=1) testte ~50 FPS seviyesinde performans kaydedilmiştir. Canlı video analizi gibi gerçek zamanlı uygulamalar için bu oran oldukça yeterli veya iyidir.

**7. Not**

Bu rapor **EnginHub** Programı kapsamında **Metehan Ayhan** tarafından hazırlanmıştır. Burada raporlanan metrikler; Kapsül Teknoloji Platformundaki Aselsan Konya Laboratuvarındaki bilgisayarların GPU ortamında kullanılan GPU (RTX 3070, 8GB VRAM) ile batch boyutu, resim boyutu (imgsz), veri seti içerikleri ve Ultralytics sürümü gibi parametrelere bağlıdır. Farklı donanımlar veya farklı sürümlerle yapılan testlerde elde edilen sonuçlar değişiklik gösterebilir. Bu raporda belirtilen değerler, spesifik olarak 4370 görselden oluşan bir insan (person) tespiti veri seti üzerinde yapılan test ölçümlerine dayanmaktadır.

**8. Sonuç**

Proje başlangıcında, **CrowdHuman** gibi büyük çaplı kalabalık insan verilerini **\*.odgt** formatında aldık ve içerisindeki fbox, vbox veya benzeri koordinat bilgilerini uygun bir Python betiğiyle çözümleyerek YOLO formatına dönüştürdük. Bu aşamada, her **\*.odgt** satırında “ID” değeri resim adını, “**fbox**” veya “**vbox**” alanları ise xmin, ymin, width, height biçiminde kişi nesnelerinin konumunu temsil ediyordu. Dönüşümde, geleneksel piksel cinsinden koordinatları normalize ederek (x\_center = (x + w/2) / image\_width, vb.) labels/.txt dosyalarına kaydettik. Böylece her resmin aynı adında bir .txt dosyası oluştu ve images/val ile labels/val klasörleri tam anlamıyla **YOLOv8’in** beklentisine uygun hale getirildi.

Bu hazırlıkların ardından, **Ultralytics’in** YOLOv8 kütüphanesini (ör. pip install ultralytics) kurduk ve Python 3.10 altında Kapsül Teknoloji Platformundaki Aselsan Konya Laboratuvarındaki bilgisayarların GPU ortamında **(NVIDIA RTX 3070)** çalışabilecek şekilde ayarlarımızı yaptık. Veri seti tanımını **data.yaml** dosyasıyla yaptık (val: parametresi images/val klasörünü işaret etti). Daha sonra, model.val(...) komutunu kullanarak doğrulama (val) aşaması gerçekleştirdik; burada model, 4370 adet validation resmini tek tek **işleyerek precision, recall, mAP@0.5, mAP@0.5:0.95** gibi temel performans göstergelerini otomatik hesapladı. Aynı süreçte, duplicate labels gibi verinin ufak tutarsızlıkları fark edildi ve bu tip uyarı mesajları çıktı kaydına eklendi.

Ardından, FPS (hız) ölçümü yapabilmek için, val klasöründeki tüm görselleri model.predict(...) yöntemiyle batch=1 senaryosunda bir döngüde işledik. Döngü öncesi ve sonrası **time.time()** yardımıyla süreyi ölçüp toplam işlenen görüntü sayısına böldük; böylece ortalama FPS ve latency (ms/görüntü) değerlerini elde ettik. Burada GPU’nun ilk çalıştırma (warm-up) gecikmesini dengelemek için, döngü başında birkaç tahmin işlemine yer verdik.

Ek olarak, GPU bellek (VRAM) kullanımını ve güç tüketimini ölçmek **için NVIDIA Management Library (NVML)** arayüzünden yararlandık. Python’da **pynvml** kütüphanesiyle, her döngü adımında nvmlDeviceGetMemoryInfo çağrısı yaparak anlık VRAM kullanımını (MB cinsinden) gözlemleyip en yüksek değeri kaydettik. Aynı şekilde nvmlDeviceGetPowerUsage fonksiyonu ile anlık güç tüketimi (**mW**) ölçümleri alıp ortalama ve maksimum güç değerlerini hesapladık. **Sonuçta ortalama 124.6 W civarı güç, maksimum 139.2 W anlık pik, ~1.2 GB VRAM kullanımı ve ~46-50 FPS seviyesinde bir gerçek zamanlı inference hızı gözlemlendi.**

Bütün bu ölçümler sonunda, projenin hem veri hazırlama (etiket format dönüşümü, klasörleme), hem model doğrulama (val metrikleri), hem de performans/hız analizleri (FPS, latency, VRAM, güç tüketimi) tamamlanmış oldu. Elde edilen metrikler, YOLOv8-L gibi büyük bir modelin gerçek zamanlı uygulamalarda (30+ FPS) rahatlıkla kullanılabileceğini ve aynı zamanda orta-üst seviyede doğruluk sağladığını ortaya koydu.

**9. Kaynakça**

<https://github.com/autogyro/yolo-V8>

<https://github.com/ultralytics/ultralytics/issues/2690>

<https://www.stereolabs.com/blog/performance-of-yolo-v5-v7-and-v8>