**NESNE TESPİTİ (OBJECT DETECTİON) VE HİYERARŞİK SINIFLANDIRMA (HİERARCHİCAL CLASSİFİCATİON)**

Bu kavramlar, genellikle görüntü işleme ve veri madenciliği alanlarında kullanılır. Nesne tespiti, bir görüntü içinde belirli nesnelerin varlığını ve konumlarını tespit etmeyi hedefler. Örneğin, bir resimde insan yüzlerinin tespiti veya bir video içinde araba trafiğinin izlenmesi gibi uygulamalar bu alanda yapılabilir. Bu tür bir sistem, bir görüntü içinde bir nesnenin nerede olduğunu belirleyebilir ve bu nesnenin hangi türde bir nesne olduğunu sınıflandırabilir. Örneğin, bir resim içinde bir araba ve bir köpek olabilir ve sistem bu iki nesnenin nerede olduğunu ve hangisinin araba, hangisinin köpek olduğunu tespit edebilir.

Hiyerarşik sınıflandırma, bir veri setindeki nesnelerin birbirleriyle ilişkili bir şekilde sınıflandırılmasını hedefler. Örneğin, bir hayvan sınıflandırma örneğinde, tüm hayvanların bir "hayvan" sınıfı altında toplandığını düşünebiliriz ve bu sınıfın altında da daha spesifik sınıflar olabilir (örneğin, "memeliler", "küçük evcil hayvanlar", "sürüngenler" gibi). Bu tür bir sınıflandırma, nesnelerin özelliklerini daha iyi anlamaya yardımcı olabilir ve bu özelliklerin nasıl birbirleriyle ilişkili olduğunu gösterir.

**YOL ÜZERİNDEKİ ARAÇLARIN TESPİTİ VE TESPİT EDİLEN ARAÇLARIN HİYERARŞİK OLARAK SINIFLANDIRILMASI**

Araçların tespiti ve tespit edilen araçların hiyerarşik olarak sınıflandırılması, görüntü işleme ve veri madenciliği alanlarında yaygın olarak kullanılan bir uygulamadır. Örneğin, bir otomatik sürüş sisteminde bir görüntü içinde yol üzerinde bulunan araçların tespiti ve bu araçların ne tür araçlar olduğunun sınıflandırılması gerekir.

Bu tür bir sistem, çeşitli yöntemler kullanarak gerçekleştirilebilir. Örneğin, yapay sinir ağları (neural networks) kullanılarak bir görüntü içinde araçların varlığı ve konumları tespit edilebilir ve bu araçların ne tür araçlar olduğu sınıflandırılabilir. Öte yandan, hiyerarşik sınıflandırma için de karar ağaçları (decision trees) gibi algoritmalar kullanılabilir. Örneğin, tespit edilen araçların ne tür araçlar olduğunu belirlemek için, araçların özelliklerine göre bir karar ağacı oluşturulabilir ve bu ağaç sayesinde araçlar hiyerarşik olarak sınıflandırılabilir.

Bu tür bir sistem, araç tespiti ve sınıflandırma işlemini otomatikleştirerek çeşitli uygulamalarda yararlı olabilir. Örneğin, bir otomatik sürüş sisteminde araç tespiti ve sınıflandırma işlemleri, sürüş koşullarının anlaşılmasını ve sistemin daha iyi bir şekilde yolu takip etmesini sağlayabilir. Ayrıca, bir trafik izleme sisteminde de araç tespiti ve sınıflandırma işlemleri, trafik akışının takibi ve trafikteki anormalliklerin tespiti gibi uygulamalar için kullanılabilir.

**MODEL VE YÖNTEM NEDİR?**

Yapay zeka modelleri, yapay zeka sistemlerinin çalışma şeklini tanımlayan algoritmalardır. Bu algoritmalar, verileri işleyerek çıkarımlar yapar ve önerilerde bulunur. Yöntemler ise, bu verileri işlemeyi ve çıkarımlar yapmayı sağlayan adımlar veya yöntemlerdir. Örneğin, bir yapay zeka modeli oluşturmak için kullanılan yöntemler arasında veri toplama, veri temizleme, veri ön işleme, model seçimi ve eğitim gibi adımlar bulunabilir.

Yapay zeka modeli, yapay zeka sisteminin çalışma şeklini tanımlarken yöntemler ise, bu modelin nasıl oluşturulduğunu açıklar. Örneğin, bir sınıflandırma modeli oluşturmak için kullanılan yöntemler arasında **veri toplama, veri ön işleme, model seçimi, eğitim ve test** adımları bulunabilir. Bu adımlar, yapay zeka modelinin nasıl oluşturulduğunu açıklar ve modelin verilere nasıl tepki vereceğini belirler.

**MODELLER NELERDİR?**

Yapay zeka modelleri, veri kümesinde gösterilen davranışı öğrenmeyi amaçlar ve bu öğrendiklerini yeni verilere uygulayarak tahminler yapmaya çalışır. Yapay zeka modelleri, çeşitli problemlerin çözümünde kullanılabilir ve genellikle aşağıdaki kategorilere ayrılırlar:

Sınıflandırma modelleri: Önceden etiketlenmiş veriler kullanılarak, modelin bir nesnenin hangi sınıfa ait olduğunu tahmin etmesini sağlar. Örneğin, bir görüntü sınıflandırma modeli, verilen görüntülerde ne tür bir nesnenin bulunduğunu tahmin edebilir.

Regresyon modelleri: Verilen veri kümesinde gösterilen ilişkiyi öğrenmeyi amaçlar ve bu ilişkiyi kullanarak yeni verilere uygun tahminler yapmaya çalışır. Örneğin, ev fiyatları regresyon modeli, verilen evlerin fiyatları ile diğer özellikleri arasındaki ilişkiyi öğrenerek, yeni bir evin fiyatını tahmin etmeye çalışır.

Clustering modelleri: Verilen veri kümesini benzer özelliklere sahip gruplara ayırmayı amaçlar. Örneğin, bir müşteri segmentasyonu modeli, müşterileri benzer alışveriş davranışlarına sahip gruplara ayırmayı amaçlar.

Bu modeller sadece birkaç örnek. Yapay zeka modelleri, bu modellerden daha fazla çeşidi de olabilir ve her bir modelin kendine özgü özellikleri ve kullanım alanları olabilir.

**BİR TEST GÖRÜNTÜSÜ ÜZERİNDEKİ BÜTÜN ARAÇLAR TESPİT EDEBİLMEK İÇİN HANGİ MODEL KULLANILMALI?**

Bir test görüntüsü üzerindeki bütün araçların tespiti için çeşitli modeller kullanılabilir. Görüntü üzerinden nesne tespiti yapmak için birçok farklı model kullanılabilir. Bu tür modellerin birçoğu Convolutional Neural Networks (CNN) tabanlıdır ve görüntüleri girdi olarak alır ve çıktı olarak nesnelerin pozisyonlarını ve etiketlerini verir. Bu tür modellerin eğitimi genellikle etiketli görüntü veri setleri kullanılarak yapılır. ***Hangi yöntem kullanılacağı, amaçlanan uygulamanın özelliklerine göre değişebilir***.

Aşağıdaki yöntemlerden bazıları kullanılabilir:

CNN (Convolutional Neural Network), RNN (Recurrent Neural Network), R-CNN (Region-based Convolutional Neural Network), Fast R-CNN, Faster R-CNN ve Mask R-CNN gibi yapılar, resim sınıflandırma, nesne tespiti ve resim segmentasyon gibi çeşitli görevler için kullanılan sinir ağı mimarileridir.

CNN (Convolutional Neural Network), resim işleme görevlerine özel olarak tasarım edilmiş bir sinir ağı mimarisi türüdür. Özellikle resimlerdeki desenleri veya özellikleri tanıma görevlerinde etkilidir.

RNN (Recurrent Neural Network), doğal dil işleme ve zaman serisi tahmini gibi sıralı verileri içeren görevlerde özellikle uygundur.

R-CNN (Region-based Convolutional Neural Network), nesne tespiti görevlerine özel olarak tasarım edilmiş bir sinir ağı mimarisi türüdür. Öncelikle bir resmin nesneler içerebilecek bölgelerini belirleyerek, bu bölgeleri bir CNN ile sınıflandırmak suretiyle çalışır.

Fast R-CNN, R-CNN mimarisinin hızlı ve etkin bir şekilde geliştirilmiş hali olup, paylaşılan bir convolutional layer kullanarak tüm girdi resmini işleyerek çalışır.

Faster R-CNN, R-CNN ve Fast R-CNN gibi, önceden tanımlanmış bölgeler kullanmayıp, potansiyel nesne bölgeleri üretebilen bir region proposal network kullanarak, Fast R-CNN'den daha hızlı ve etkili bir şekilde çalışır.

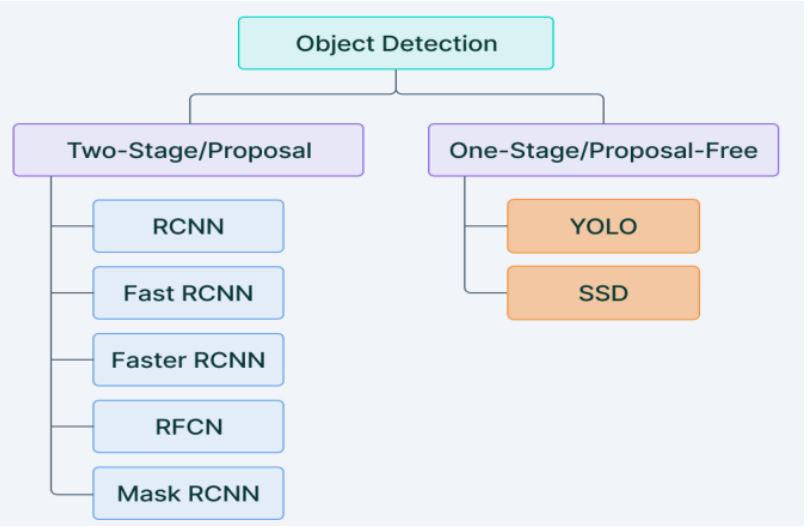
Mask R-CNN, Faster R-CNN mimarisinin bir uzantısıdır ve instance segmentation (örnek segmentasyonu) yapabilme yeteneği ekler. Bu, sadece bir resimdeki nesneleri tespit etmeyi değil, aynı zamanda bunları segmente etmeyi ve her nesne için bir maske oluşturmayı da mümkün kılar.

YOLO (You Only Look Once), nesne tespiti görevlerine özel olarak tasarım edilmiş bir sinir ağı mimarisi türüdür. Özellikle gerçek zamanlı uygulamalar için uygun olması sebebiyle çok hızlı ve etkili olduğu bilinir.

Single Shot Detector (SSD): Bu yöntem, görüntülerdeki nesneleri tespit etmek için kullanılan bir derin öğrenme yöntemidir. SSD modelleri, tek bir görüntü üzerinde nesneleri tespit etmeyi öğrenebilir ve bu sayede, hızlı sonuçlar verebilir.

Haar cascades: Haar cascades, görüntü işleme alanında yaygın olarak kullanılan bir yöntemdir. Bu yöntem, bir görüntü içinde belirli bir nesnenin varlığını tespit etmek için kullanılır. Örneğin, bir test görüntüsü üzerinde araç tespiti için Haar cascades kullanılabilir.

Hough dönüşümü: Hough dönüşümü, görüntü işleme alanında yaygın olarak kullanılan bir yöntemdir. Bu yöntem, görüntü içinde çizgilerin ve dairelerin tespit edilmesine yardımcı olur. Örneğin, bir test görüntüsü üzerinde araç tespiti için Hough dönüşümü kullanılabilir.



**GÖRÜNTÜ İŞLEME YÖNTEMLERİ Mİ MAKİNE ÖĞRENİMİ MODELLERİ Mİ?**

Görüntü işleme yöntemleri ve makine öğrenimi yöntemleri her ikisi de görüntüler üzerinde işlem yapmak için kullanılabilir. Hangisinin daha uygun olduğu, ne tür bir problem çözmeye çalıştığınıza ve ne tür veriye sahip olduğunuza göre değişebilir.

Görüntü işleme yöntemleri, görüntülerdeki nesneleri tanımlamak, tespit etmek ve izlemek için yapay zeka ve bilgisayar bilimleri alanında kullanılan yöntemlerdir. Örneğin, bir görüntüde aracı tespit etmek için, görüntünün ön planında ve arka planında farklı özellikleri tanımlayabilir ve bu özellikleri kullanarak aracı tespit etmeye çalışabilirsiniz.

Makine öğrenimi yöntemleri ise, verilen bir veri kümesi üzerinden modele öğretilen bir problemi çözmeye çalışan yöntemlerdir. Örneğin, makine öğrenimi yöntemlerini kullanarak bir görüntüde aracı tespit etmeye çalışabilirsiniz. Bu yöntemler, veri kümesinde aracın resimleri ve aracın türü gibi etiketler ile eğitilir ve daha sonra görüntülerde aracı tespit etmeye çalışır.

Her iki yöntem de görüntüler üzerinde işlem yapmak için kullanılabilir, ancak hangisinin daha uygun olduğu, **ne tür bir problem çözmeye çalıştığınıza ve ne tür veriye sahip olduğunuza göre değişebilir. Örneğin, eğer veri kümeniz yeterli sayıda etiketli veri içeriyorsa ve problemi çözmek için makine öğrenimi yöntemlerini kullanmayı düşünüyorsanız, bu yöntemler daha uygun olabilir**. Ancak, eğer veri kümeniz yeterli sayıda etiketli veri içermiyorsa veya problemi çözmek için daha esnek bir yönteme ihtiyaç duyuyorsanız, görüntü işleme yöntemleri daha uygundur.

**MODEL SEÇİMİ**

Bir görüntü üzerinde ilk olarak araçları tespit etmek ve daha sonra araçların türlerini hızlı bir şekilde tahmin etmek için, nesne tespiti ve sınıflandırma algoritmalarının bir kombinasyonunu kullanabilirsiniz. Bir yöntem, YOLO gibi hızlı bir nesne tespiti modelini kullanarak görüntüdeki araçları tespit etmek ve daha sonra ayrı bir araç sınıflandırma modelini kullanarak her araçın türünü tahmin etmek olabilir.

Başka bir seçenek de hem nesne tespiti hem de sınıflandırma yapabilen bir tek uçtan uca model kullanmak olabilir. Bu tür bir görev için sıklıkla kullanılan böyle bir model, Faster R-CNN mimarisidir. Faster R-CNN, nesne tespiti görevlerinde özellikle uygundur ve ayrıca resim sınıflandırma ve nesne izleme gibi görevlerde de kullanılmıştır.

Vakamız **nesne tespiti ve hiyerarşik sınıflandırma** problemi üzerine olup elde edilecek sonucun **5-6 saniyeden kısa olması** beklendiği için ***YOLOv5 mimarisi kullanılmaya karar verilmiştir.***

**MODELLERİN BU PROBLEM İÇİN AVANTAJLARI VE DEZAVANTAJLARI**

YOLO (You Only Look Once): YOLO, basitliği ve verimliliği nedeniyle popüler hale gelen hızlı ve doğru bir nesne tespiti modelidir. Bir görüntüdeki nesnelerin sınıf olasılıklarını ve sınırlayıcı kutuları tahmin etmek için tek bir convolutional neural network (CNN) kullanır. YOLO'nun ana avantajlarından biri, görüntüleri gerçek zamanda işleyebilmesidir, bu nedenle hızlı öğrenme gereken uygulamalar için uygun hale gelir. Ancak, YOLO küçük veya çok kapalı nesnelerle zorlanabilir ve başka bazı modellerden daha yüksek olmayan bir doğruluk oranına sahip olabilir.

R-CNN (Regional CNN): R-CNN ve varyantları (Fast R-CNN, Faster R-CNN, Mask R-CNN), bir görüntüdeki nesneleri tespit etmek için bölge önerisi ve sınıflandırma ağlarının bir kombinasyonunu kullanan nesne tespiti modelidir. Bu modeller YOLO'dan daha doğru olmasına rağmen aynı zamanda daha yavaştır ve bu nedenle gerçek zamanlı uygulamalar için uygun değildir.

SSD (Single Shot Detector): SSD, bir görüntüdeki nesnelerin sınıf olasılıklarını ve sınırlayıcı kutuları tahmin etmek için bir CNN kullanan tek aşamalı bir nesne tespiti modelidir. R-CNN ve varyantlarından daha hızlıdır, ancak daha az doğrudur.

RetinaNet, nesne tespiti veri kümelerinde sınıf dengesizliği problemini çözmek için focal loss fonksiyonunun bir varyantını kullanan tek aşamalı bir nesne tespiti modelidir. SSD'den daha doğru olmasına rağmen daha yavaştır.

***Özetle, nesne tespiti modeli seçimi uygulamanızın özel gereksinimlerine göre belirlenir. YOLO, hızlı ve verimli nesne tespiti için iyi bir seçimdir****, R-CNN ve varyantları ise daha doğru ancak daha yavaştır. SSD ve RetinaNet hız ve doğruluk açısından orta seviyededir*.

**VERİ SETLERİNİN DÜZENLENMESİ**

Veri Seti formatı

YOLOv5 (You Only Look Once version 5), Ultralytics LLC tarafından geliştirilen gerçek zamanlı nesne tespit sistemidir. Görüntüler ve videolarda çeşitli nesneleri tespit edebilir.

Girdi verileri bakımından, **YOLOv5 JPEG, JPG**, PNG, BMP ve GIF gibi çeşitli formatlarla çalışabilir ve videolar için AVI, MP4 ve MOV gibi çeşitli formatlarla da çalışabilir. Girdi verileri için belirli gereksinimler, YOLOv5'i çalıştırmak için kullandığınız donanım ve yazılım yapılandırmasına bağlıdır. Genellikle, en iyi performans için yüksek çözünürlüklü görüntüler ve videolar kullanılması önerilir, ancak tam çözünürlük uygulama ve mevcut kaynaklara bağlıdır.

Ayrıca, YOLOv5'in girdi verilerinin, görüntü veya videodaki ilgilendiği nesnelerin konumunu gösteren sınırlamalı kutularla işaretlenmiş olmasını beklediğini de belirtmekte fayda var. Bu sınırlamalı kutular, el ile oluşturulabilir veya ayrı bir nesne işaretleme aracı kullanılarak oluşturulabilir. Sınırlamalı kutu işaretlerinin formatı, kullandığınız YOLOv5'in özel uygulamasına bağlı olacaktır, **ancak genellikle koordinatlar ve sınıf etiketlerinin bir metin (.txt)** dosyasında bir listesi olarak temsil edilirler.

Vakamız için **images’lar** için ***JPG*** , **lables’lar** için ***.*txt *kullanılmasına karar verilmiştir.***

Veri Ayrıştırma Yöntemleri

Veri kümenizi eğitim ve test verileri olarak ayırmak için birkaç yöntem kullanılabilir:

Rastgele Ayırma: Veri kümenizdeki örnekler rastgele bir şekilde eğitim ve test verilerine dağıtılır. Bu yöntem, veri kümeninizin örneklerinin rastgele dağılımına göre işe yarar.

K-Katı Bölme: Veri kümeniz K katı olarak bölünür ve her bir bölüm test verisi olarak kullanılır. Bu yöntem, veri kümeninizdeki örneklerin birbirleriyle ilişkili olma olasılığı daha yüksek ise işe yarar.

Stratejik Bölme: Veri kümeniz stratejik bir şekilde eğitim ve test verilerine ayırılır. Örneğin, veri kümeninizdeki örneklerin zaman dilimlerine göre ayırılabilir. Bu yöntem, veri kümeninizdeki örneklerin zaman dilimlerine göre ilişkili olma olasılığı daha yüksek ise işe yarar.

Ayrık Bölme: Veri kümeniz ayrık olarak eğitim ve test verilerine ayırılır. Bu yöntem, veri kümeninizin örneklerinin ayrık dağılımına göre işe yarar.

Veri ayırma yöntemlerinin hangisi en uygun olduğu, veri kümeninizin özelliklerine ve amaçlarınıza göre değişebilir. Örneğin, veri kümeninizdeki örneklerin zaman dilimlerine göre ilişkili olma olasılığı yüksekse, stratejik bölme yöntemini kullanmak daha uygun olabilir.

**Verilerimiz, YOLOv5** modeline uygun olacak sekilde ***4K K katlı bölme yöntemi kullanılarak train(6164) ve val(2054) şeklinde bölünmüştür. TXT formati da ise kullanılma hazır hale getirilmiştir.***

Veri Çoğaltma

**Veri çoğaltma (data augmentation),** veri setinizdeki verileri değiştirerek yeni veri üreterek veri setinizi büyütmeyi amaçlar. Bu teknik, özellikle çok az sayıda veriye sahip olan veri setlerinde kullanışlıdır ve öğrenme modelinizin daha iyi performans göstermesine yardımcı olabilir. Veri çoğaltma, özellikle görüntüler, sesler ve metinler gibi sayısal veriler için kullanılabilir. Örnek olarak, bir görüntü veri setinizdeki resimleri farklı açılardan yeniden çekilerek, ses dosyalarını farklı seslerle yeniden kaydederek veya metinleri farklı dil ve yazım şekillerine çevirerek veri setinizi büyütebilirsiniz:

Resimleri farklı açılardan yeniden çekmek: Örneğin, bir çiçek resmini farklı açılardan yeniden çekerek, çiçeğin her yönünü gösteren yeni resimler oluşturabilirsiniz.

Ses dosyalarını farklı seslerle yeniden kaydetmek: Örneğin, bir cümleyi farklı insanlar tarafından seslendirilerek, farklı seslerdeki aynı cümleyi içeren yeni ses dosyaları oluşturabilirsiniz.

Metinleri farklı dil ve yazım şekillerine çevirmek: Örneğin, bir metni farklı dillerde çevirerek, aynı anlamdaki metinleri farklı dillerde içeren yeni metinler oluşturabilirsiniz.

Görüntüleri bozmak: Örneğin, resimleri bulanıklaştırarak, aynı görüntüyü farklı şekillerde gösteren yeni resimler oluşturabilirsiniz.

Görüntüleri parçalayarak yeniden birleştirme: Örneğin, bir resmi farklı parçalara ayırarak, yeniden birleştirerek farklı şekillerde aynı görüntüyü gösteren yeni resimler oluşturabilirsiniz.

**Verilerimiz, YOLOv5** ***modeli için yeterli görülmüştür*.** *Bu yüzden veri çoğaltma yöntemi kullanılmamıştır.*Modelin doğrulunun düşük olması halinde gerekli görüldüğü kadar artırılabilir.

**MODEL EĞiTiMi iÇiN BELiRLEMiŞ OLDUĞUNUZ HiPERPARAMETRELER**

**train:** weights=yolov5s.pt, cfg=, data=custom\_data.yaml, hyp=data/hyps/hyp.scratch-low.yaml, epochs=500, batch\_size=128, imgsz=128, rect=False, resume=False, nosave=False, noval=False, noautoanchor=False, noplots=False, evolve=None, bucket=, cache=None, image\_weights=False, device=, multi\_scale=False, single\_cls=False, optimizer=SGD, sync\_bn=False, workers=8, project=runs/train, name=exp, exist\_ok=False, quad=False, cos\_lr=False, label\_smoothing=0.0, patience=100, freeze=[0], save\_period=-1, seed=0, local\_rank=-1, entity=None, upload\_dataset=False, bbox\_interval=-1, artifact\_alias=latest

weights: yolov5s.pt: Bu parametre, eğitim sırasında kullanılacak ağırlık dosyasının yolunu gösterir.

cfg: Bu parametre, eğitim sırasında kullanılacak ağ yapısının yolunu gösterir.

data: custom\_data.yaml: Bu parametre, eğitim verisi için kullanılacak veri yapısını ve veri kümesinin yolunu gösterir.

hyp: data/hyps/hyp.scratch-low.yaml: Bu parametre, eğitim hiperparametrelerinin yolunu gösterir.

epochs: 500: Bu parametre, eğitim için kaç epoch (eğitim turu) yapılacağını gösterir.

batch\_size: 128: Bu parametre, her bir epoch sırasında kullanılacak olan batch (öğrenme adımı) boyutunu gösterir.

imgsz: 128: Bu parametre, eğitim için kullanılacak olan resimlerin boyutunu gösterir.

rect: False: Bu parametre, resimlerdeki etiketlerin dikdörtgen olarak işaretlenip işaretlenmediğini gösterir.

resume: False: Bu parametre, eğitimin önceki bir checkpoint'tan devam edip etmeyeceğini gösterir.

nosave: False: Bu parametre, eğitim sırasında checkpoint dosyalarının kaydedilip kaydedilmeyeceğini gösterir.

noval: False: Bu parametre, eğitim sırasında validasyon kümesi kullanılıp kullanılmayacağını gösterir.

noautoanchor: False: Bu parametre, otomatik olarak anchor (bağlantı noktası) üretecek bir sistemin kullanılıp kullanılmayacağını gösterir.

noplots: False: Bu parametre, eğitim sırasında performans göstergelerinin görselleştirilip görselleştirilmeyeceğini gösterir.

evolve: 300: Bu parametre, eğitim sırasında hiperparametrelerin evrimleştirilerek optimize edileceğini gösterir. Bu parametre, evolüsyon için kaç epoch kullanılacağını gösterir.

bucket: Bu parametre, eğitim verisi için kullanılacak Google Cloud Storage bucket (depo) adını gösterir.

cache: None: Bu parametre, eğitim verisi için kullanılacak cache (önbellek) dosyasının yolunu gösterir.

image\_weights: False: Bu parametre, eğitim sırasında resimlerin ağırlıklarının manuel olarak atanıp atanmayacağını gösterir.

device: Bu parametre, eğitim işleminin hangi cihazda (CPU, GPU, vb.) yapılacağını gösterir.

multi\_scale: False: Bu parametre, eğitim sırasında çoklu ölçekli resimler kullanılıp kullanılmayacağını gösterir.

single\_cls: False: Bu parametre, eğitim sırasında sadece tek bir sınıfın öğrenileceğini gösterir.

optimizer: SGD: Bu parametre, eğitim sırasında kullanılacak optimizasyon algoritmasını gösterir.

sync\_bn: False: Bu parametre, eğitim sırasında Batch Normalization (BN) işleminin eşzamanlı olarak yapılıp yapılmayacağını gösterir.

workers: 8: Bu parametre, eğitim sırasında veri yükleme işleminin paralel olarak kaç iş parçacığı ile yapılacağını gösterir.

project: runs/train: Bu parametre, eğitim sırasında üretilecek raporların ve diğer artefakların kaydedileceği klasörü gösterir.

name: exp: Bu parametre, eğitim süreci için kullanılacak özel isimdir.

exist\_ok: False: Bu parametre, eğitim sırasında özel isimli bir klasörün zaten var olup olmadığını gösterir.

quad: False: Bu parametre, eğitim sırasında dörtlü anchor (bağlantı noktası) kullanılıp kullanılmayacağını gösterir.

cos\_lr: False: Bu parametre, eğitim sırasında Cosine Annealing learning rate (öğrenme hızı) kullanılıp kullanılmayacağını gösterir.

label\_smoothing: 0.0: Bu parametre, eğitim sırasında etiketlerin düzgünleştirilmesi işlemini gösterir.

patience: 100: Bu parametre, eğitim sırasında "early stopping" işleminin kullanılacağını gösterir. "Early stopping" işlemi, eğitim sırasında belirli bir epoch sayısına ulaşılması durumunda eğitimi durdurmak için kullanılır. Bu parametre, "early stopping" için kaç epoch bekleneceğini gösterir.

freeze: [0]: Bu parametre, eğitim sırasında ağın belirli katmanlarının dondurulup dondurulmayacağını gösterir. Bu parametre, dondurulacak katmanların sıra numaralarını liste olarak gösterir.

save\_period: -1: Bu parametre, eğitim sırasında kaç epoch aralıklarla checkpoint dosyalarının kaydedileceğini gösterir.

seed: 0: Bu parametre, eğitim için kullanılacak rastgele sayı üretecinin seed değerini gösterir.

local\_rank: -1: Bu parametre, eğitim sırasında distributed (dağıtılmış) eğitim kullanılıp kullanılmayacağını ve hangi cihazın hangi rolü üstleneceğini gösterir.

entity: None: Bu parametre, eğitim sırasında eğitim etkinliğinin takip edileceği proje veya organizasyonu gösterir.

upload\_dataset: False: Bu parametre, eğitim verisinin Google Cloud Storage'a yüklenip yüklenmeyeceğini gösterir.

bbox\_interval: -1: Bu parametre, eğitim sırasında kaç epoch aralıklarla bounding box (sınır kutusu) verilerinin kaydedileceğini gösterir.

artifact\_alias: latest: Bu parametre, eğitim sırasında üretilen artefakların yönetim sistemine yüklenecek olan isimdir.

Bu hiperparametrelerin nasıl belirleneceği**, modelin amaçlarına ve veri kümesine göre değişebilir. Bu hiperparametreleri ayarlamak, modelin performansını optimize etmeyi amaçlar ve genellikle deney ve yanılma yöntemi ile yapılır.**

! python train.py --img 128 --batch 128 --epochs 500

--data custom\_data.yaml --weights yolov5s.pt

Bu bir train.py adlı bir betiğin çalıştırılmasını gösterir, bazı komut satırı argümanları ile birlikte. Aşağıda, her bir argümanın açıklaması bulunmaktadır:

--img 128: Bu, veri kümesindeki görüntülerin boyutunu belirtir. 128 değeri, görüntülerin genişliğinin ve yüksekliğinin 128 piksel olduğunu gösterir.

--batch 128: Bu, eğitim sırasında kullanılacak olan lot (batch) boyutunu belirtir. Lot (batch) boyutu, bir ileri/geri geçişte kullanılan eğitim örneklerinin sayısıdır.

--epochs 500: Bu, yapılacak olan eğitim epoch (tur) sayısını belirtir. Bir epoch (tur), eğitim veri kümesinin tamamının bir kez tamamen geçilmesidir.

--data custom\_data.yaml: Bu, eğitim için kullanılacak veri kümesini içeren bir dosyanın yolunu belirtir. Dosya, YAML formatında olmalıdır.

--weights yolov5s.pt: Bu, modeli başlatmak için önceden eğitilmiş ağırlıkları içeren bir dosyanın yolunu belirtir. Dosya, PyTorch formatında olmalıdır.

**hyperparameters:** lr0=0.01, lrf=0.01, momentum=0.937, weight\_decay=0.0005, warmup\_epochs=3.0, warmup\_momentum=0.8, warmup\_bias\_lr=0.1, box=0.05, cls=0.5, cls\_pw=1.0, obj=1.0, obj\_pw=1.0, iou\_t=0.2, anchor\_t=4.0, fl\_gamma=0.0, hsv\_h=0.015, hsv\_s=0.7, hsv\_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, mosaic=1.0, mixup=0.0, copy\_paste=0.0

lr0: Bu parametre, eğitim sırasında kullanılan başlangıç öğrenme oranını belirler. Öğrenme oranı, modelin eğitim sırasında nasıl öğreneceğini belirleyen bir parametredir ve genellikle modelin eğitim sırasında güncellenir.

lrf: Bu parametre, eğitim sırasında kullanılan son öğrenme oranını belirler. Bu değer, modelin eğitim sırasında öğrenme oranının sona doğru ne kadar azaltılacağını belirler.

momentum: Bu parametre, modelin eğitim sırasında momentumu belirler. Momentum, bir önceki adımda yapılan değişikliklerin, modelin şu anki adımda nasıl kullanılacağını belirler. Yüksek momentum değerleri, modelin önceki adımda yaptığı hataları düzgün bir şekilde düzeltmesine yardımcı olurken, düşük momentum değerleri modelin daha fazla adapte olmasına yardımcı olur.

weight\_decay: Bu parametre, modelin ağırlıklarının eğitim sırasında ne kadar azaltılacağını belirler. Ağırlık azaltımı (weight decay), modelin eğitim sırasında overfitting'e (aşırı özelleştirme) engel olmak için kullanılır.

warmup\_epochs: Bu parametre, modelin eğitim sırasında öğrenme oranının (learning rate) ne kadar süre ile yavaş yavaş arttırılacağını belirler. Bu, modelin eğitim sırasında daha iyi performans göstermesine yardımcı olur.

warmup\_momentum: Bu parametre, modelin eğitim sırasında momentumunun ne kadar süre ile yavaş yavaş arttırılacağını belirler.

warmup\_bias\_lr: Bu parametre, modelin eğitim sırasında bias parametrelerinin (modelin ağırlıklarının bir özel kısmı) öğrenme oranını belirler.

box: Box parametreleri, modelin eğitim sırasında nesnelerin nerede olduğunu tahmin etmesine yardımcı olan parametrelerdir. Bu parametre, modelin eğitim sırasında box parametrelerine ne kadar ağırlık verileceğini belirler. Genellikle, modelin eğitim sırasında box parametrelerinin önemini belirleyen bu değer, modelin nesneleri doğru bir şekilde tahmin etmesine yardımcı olur.

cls: Bu parametre, modelin eğitim sırasında sınıflandırma parametrelerinin önemini belirler.

cls\_pw: Bu parametre, modelin eğitim sırasında sınıflandırma parametrelerine ağırlık verme miktarını belirler.

obj: Bu parametre, modelin eğitim sırasında nesne parametrelerinin önemini belirler.

obj\_pw: Bu parametre, modelin eğitim sırasında nesne parametrelerine ağırlık verme miktarını belirler.

iou\_t: Bu parametre, modelin eğitim sırasında nesnelerin birbirleriyle olan "örtüşme oranlarını" (intersection-over-union, IOU) belirler. IOU, iki nesnenin birbirlerine kaç yüzde oranında örtüştüğünü belirler.

anchor\_t: Bu parametre, modelin eğitim sırasında "anchor" parametrelerinin önemini belirler. Anchor parametreleri, modelin eğitim sırasında nesnelerin nerede olduğunu tahmin etmesine yardımcı olan parametrelerdir.

fl\_gamma: Bu parametre, modelin eğitim sırasında "focal loss" parametrelerinin önemini belirler. Focal loss, modelin eğitim sırasında düşük öğrenme oranlarına sahip nesnelerin tahmin edilmesine yardımcı olan bir parametredir.

hsv\_h: Bu parametre, modelin eğitim sırasında görüntülerin "HSV" (hue, saturation, value) renk uzayında H (hue) kanalının önemini belirler.

hsv\_s: Bu parametre, modelin eğitim sırasında görüntülerin "HSV" (hue, saturation, value) renk uzayında S (saturation) kanalının önemini belirler.

hsv\_v: Bu parametre, modelin eğitim sırasında görüntülerin "HSV" (hue, saturation, value) renk uzayında V (value) kanalının önemini belirler.

degrees: Bu parametre, modelin eğitim sırasında görüntülerin döndürülme açısını belirler.

translate: Bu parametre, modelin eğitim sırasında görüntülerin yatay ve düşey olarak ne kadar hareket ettirileceğini belirler.

scale: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar büyütüleceğini veya küçültüleceğini belirler.

shear: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar kıvrılacağını belirler.

perspective: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar perspektif değiştirileceğini belirler.

flipud: Bu parametre, modelin eğitim sırasında görüntülerin yukarıdan aşağıya doğru ne kadar yansıtılacağını belirler.

fliplr: Bu parametre, modelin eğitim sırasında görüntülerin soldan sağa doğru ne kadar yansıtılacağını belirler.

mosaic: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar "mozaik" (parçalı) hale getirileceğini belirler.

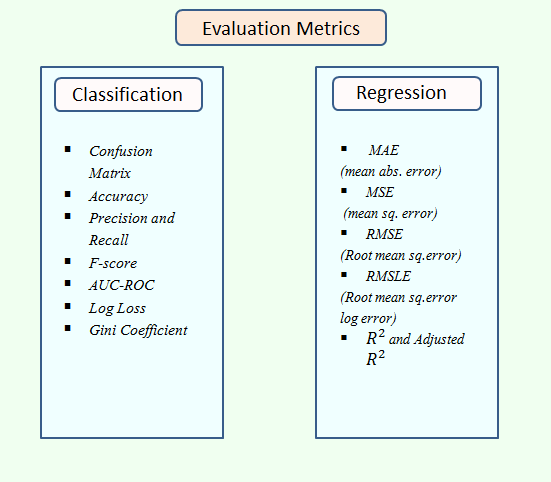
mixup: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar karıştırılacağını belirler.

copy\_paste: Bu parametre, modelin eğitim sırasında görüntülerin ne kadar "kopyala-yapıştır" işlemiyle değiştirileceğini belirler.

anchors: Bu parametre, modelin eğitim sırasında kullanılan anchor parametrelerinin sayısını belirler. Anchor parametreleri, modelin eğitim sırasında nesnelerin nerede olduğunu tahmin etmesine yardımcı olan parametrelerdir.

**TEST GÖRÜNTÜLERiNiN DEĞERLENDiRiLMESi**

1. **Konfüzyon Matrisi**
2. **Doğruluk (Accuracy)**
3. **Hassasiyet (Precision)**
4. **Duyarlılık (Sensitivity) veya Tam Doğruluk (Recall)**
5. **F1 Skoru**
6. **ROC Eğrisi ve AUC (Area Under the Curve)**
7. **Fowlkes-Mallows Skoru**
8. **Jaccard Benzerlik Oranı**
9. **Hamming Uzaklığı**
10. **Hata Oranı**
11. **Logaritmik Hata (Log Loss)**
12. **Matthews Korelasyon Katsayısı (Matthews Correlation Coefficient)**
13. **Çapraz Doğruluk (Cross Validation)**
14. **Hata Karesi Ortalaması (Mean Squared Error - MSE)**
15. **Kök Ortalama Hata Kareleri (Root Mean Squared Error - RMSE)**
16. **Kök Ortalama Hata Kareleri İçin İstatistik (Coefficient of Determination - R²)**
17. **Hata İstatistiği (Error Statistic)**
18. **İç Özgünlük (Inner Consistency)**

****

Konfüzyon matrisi, bir sınıflandırma modelinin performansını ölçmek için kullanılan bir araçtır. Konfüzyon matrisi, doğru pozitif (TP), yanlış pozitif (FP), doğru negatif (TN), ve yanlış negatif (FN) değerlerini içerir. Bu değerler, modelin sınıflandırma performansını değerlendirmede kullanılır.

Doğru pozitif (TP) değeri, modelin pozitif olarak sınıflandırdığı örneklerin aslında pozitif olduğu örneklerin sayısıdır.

Yanlış pozitif (FP) değeri, modelin pozitif olarak sınıflandırdığı örneklerin aslında negatif olduğu örneklerin sayısıdır.

Doğru negatif (TN) değeri, modelin negatif olarak sınıflandırdığı örneklerin aslında negatif olduğu örneklerin sayısıdır.

Yanlış negatif (FN) değeri ise, modelin negatif olarak sınıflandırdığı örneklerin aslında pozitif olduğu örneklerin sayısıdır.

Konfüzyon matrisi genellikle aşağıdaki gibi gösterilir:

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu değerlerden yola çıkarak, bir sınıflandırma modelinin performansını ölçmek için birçok metrik hesaplanabilir, bunlar arasında duyarlılık (Sensitivity), hassasiyet (Precision), F1 skoru (F1 Score) gibi metrikler bulunmaktadır.

(Accuracy) - Doğruluk (Accuracy), modelin doğru pozitif ve doğru negatif sonuçları verme olasılığını ölçer. Doğruluk aşağıdaki formülle hesaplanır:

Doğruluk (Accuracy) = (TP + TN) / (TP + TN + FP + FN)

Bu formül, konfüzyon matrisi değerlerinden yola çıkarak doğruluk değerini hesaplar. Doğruluk değeri, 0 ile 1 arasında bir sayıdır ve modelin performansını ölçer. Daha yüksek doğruluk değeri, modelin daha iyi bir performans sergilediğini gösterir.

Bir sınıflandırma modelinin performansını ölçmek için birçok metrik hesaplanabilir. Bu metrikler, konfüzyon matrisi değerlerinden yola çıkarak hesaplanır. Aşağıda, bazı önemli metrikler ve nasıl hesaplandıkları gösterilmiştir:

Duyarlılık (Sensitivity) veya Tam Doğruluk (Recall) - Modelin pozitif olarak sınıflandırdığı örneklerin aslında pozitif olduğu örneklerin oranıdır. Duyarlılık aşağıdaki formülle hesaplanır:

Duyarlılık (Sensitivity) = TP / (TP + FN)

Hassasiyet (Precision) - Modelin doğru pozitif sonuçları verme olasılığıdır. Hassasiyet aşağıdaki formülle hesaplanır:

Hassasiyet (Precision) = TP / (TP + FP)

F1 Skoru (F1 Score) - Hassasiyet ve duyarlılığın birleşimidir. F1 skoru aşağıdaki formülle hesaplanır:

F1 Skoru (F1 Score) = 2 \* (Hassasiyet \* Duyarlılık) / (Hassasiyet + Duyarlılık)

Çarpım Hassasiyeti (Multiply Accurate): Modelin doğru pozitif ve doğru negatif sonuçları verme olasılığıdır. Çarpım hassasiyeti aşağıdaki formülle hesaplanır:

Çarpım Hassasiyeti (Multiply Accurate) = (TP \* TN) / (TP + TN)

ROC Eğrisi ve AUC (Area Under the Curve) - Bir modelin pozitifler ve negatifleri ayırt etme performansını ölçen bir grafiktir. AUC, ROC eğrisinin altındaki alanı ölçer. 1'e yakın bir AUC değeri, iyi bir model olduğunu gösterir.

Log loss, ağırlıklı hata karesi veya karmaşıklık kaybı olarak da bilinir ve bir sınıflandırma modelinin performansını ölçmek için kullanılan bir ölçümdür. Log loss, 0 ile 1 arasında bir olasılık değeri çıkaran bir sınıflandırma modelinin çıktısını dikkate alır. Örneklerin beklenen olasılığı ve örneklerin gerçek etiketleri dikkate alınarak hesaplanır. Hedef, log loss'u minimize etmektir, yani tahmin edilen olasılık gerçek etiketle çok yakın olur.

Gini katsayısı ise, ekonomide adaletsizlik ölçümü olarak kullanılan bir ölçümdür. Makine öğrenimi bağlamında, genellikle ikili sınıflandırma modelinin performansını değerlendirmek için kullanılır. Gini katsayısı, rastgele seçilen bir pozitif örneğin, rastgele seçilen bir negatif örnekten daha yüksek sıralanma olasılığı olarak düşünülebilir. Gini katsayısı yüksek olan bir model, pozitif ve negatif örnekleri iyi ayırt edebildiğinden iyi performansa sahip sayılır.

***Önemli olan, hangi metriğin kullanılacağına karar verirken problemi ve amaçlarınızı dikkatlice düşünmektir.***

MERT UZGÖZ

linkedin.com/in/mertzgz

[mertzgz@gmail.com](mailto:mertzgz@gmail.com)

+90 542 396 8119