YOLO(You Only Look Once) Algoritması gerçek zamanlı nesne takibi yapabilen, halihazırda kullanılan algoritmalara oranla çok hızlı çalışan ve bunun için CNN(Evrişimsel Sinir Ağları) kullanan bir algoritmadır. 2016 yılında YOLOv1 numarasıyla ilk kez kullanıma sunulmuştur. Şu an ise YOLOv10 versiyonu kullanılabilmektedir. YOLO yalnızca diğer algoritmalara oranla değil, kendi versiyonları içerisinde de hızına hız katmıştır. YOLO modeli, sınırlayıcı kutuları tahmin etme prosedürünü sınıf etiketleriyle uçtan uca türevlenebilir bir ağda birbirine bağlayan ilk nesne dedektörüydü.

YOLO ağı üç ana parçadan oluşmaktadır.

1. [**Omurga**](https://blog.roboflow.com/glossary/#:~:text=backbone) : Farklı granülaritelerde görüntü özelliklerini toplayan ve oluşturan bir evrişimli sinir ağı.
2. [**Boyun**](https://blog.roboflow.com/glossary/#:~:text=neck%20-) :Görüntü özelliklerini tahmine iletmek için karıştırıp birleştiren bir dizi katman.
3. [**Baş**](https://blog.roboflow.com/glossary/#:~:text=head%20-) :Boyundan gelen özellikleri tüketir ve kutu ve sınıf tahmin adımlarını atar.

Genellikle göz ardı edilse de model eğitimi esnasında önem arz eden eğitim prosedürlerinden bahsedecek olursak;

* Veri Artırımı :

Bu sayede eğitim seti izole edilmiş sabit verilerden ziyade daha karmaşık ve daha geniş bir anlamsal değişim aralığına maruz bırakılır. Böylece doğruluk oranında artış sağlanması beklenir.

* Kayıp Hesaplamaları:

Bu geliştirme ile kayıp fonksiyonlarının ortalama hassasiyetine etkisini en aza indirip, sonuçları maksimize etmeyi amaçlar.

YOLO algoritmasının çalışma mantığı ve neden hızlı olduğunu açıklamak gerekirse; Algoritma nesne tanımlamayı adından da anlaşılacağı gibi tek seferde yapmaktadır. Diğer algoritmalardan farklı olarak nesne takibini yaparken görüntüyü bölgelere ayırır (Grid). Gridler içerisindeki görüntülere ait nesneleri tek tek kutucuklara (Bounding Box) koyar. Her bir bounding box içerisinde nesne olma olasılığını hesaplar ve Güven Skoru (Trust Score) çıkararak bize kutucuklar içerisindeki nesnelerin yüzde kaç tahmin edilen nesneyle benzerlik gösterdiğini söyler. Bu kutucuklara ait değerler yalnızca güven skoruyla sınırlı değildir. Ayrıca her bir kutunun içerisinde (eğer nesne varsa) orta noktası bu kutunun içerisinde mi diye kontrol eder ve son olarak da uzunluk,yükseklik ve bağıl sınıf olasılığının tespitini yapar.(Bağıl sınıf modelde ne kadar sınıf olduğuna göre değişkenlik göstermektedir.)

Her bir kutucuğun kendi içerisinde aranan nesneyi bulma olasılığı vardır bu da algoritmada karışıklığa neden olmaktadır. Bunun önüne geçmek için bounding box’lardan güven skoru en yüksek olan kutucuğu (non-maximum suppression algorithm) baz alır. Temel mantığıyla YOLOv1 böyle çalışmaktadır. Aynı görüntü içerisinde birden fazla nesnenin orta noktasının tespiti yapıldığı durumlarda YOLOv2 algoritması ortaya çıkmıştır.

Yolov2 algoritmasında anchor(çapa) kullanılarak birden fazla nesnenin orta noktası tespit edildiğinde her nesneye özel kutucuklar oluşturur ve her kutucuğun hesaplanmasını ayrı ayrı yapar. (Güven skoru, yükseklik,uzunluk,bağıl sınıf olasılığı). YOLOv2 algoritmasında Darknet-19 derin ağ mimarisi kullanılmaktadır. Bu ağ mimarisine göre CNN ve nesne takibinde performansı arttırmak için YOLOv1’e ek olarak 11 katman daha eklenmiştir.

Görüntüdeki küçük nesnelerin tespitinde eksiklik olduğu hissedildiğinden YOLOv3 gereksinimi ortaya çıkmıştır. YOLOv2’nin performansı küçük nesnelerin tespiti için yeterli olmamıştır. Katman sayısını arttırarak Darknet-53 mimarisini kullanmaya başlamışlardır. Fazla katman, hesaplanması gereken daha fazla iş olduğundan hız konusundan taviz verilmiştir. YOLOv3 ile her katmanda Batch Normalization işlemi yapılmış Leaky ReLU aktivasyon fonksiyonları kullanılarak model eğitim süreci hızlandırılmış, öğrenme kapasitesi arttırılmış ve daha derin ağlarda kararlılık sağlanmıştır. Ayrıca farklı ölçeklerde nesne tespiti yapabilmek için Spatial Pyramid Pooling fonksiyonu da eklenmiştir. YOLOv3’ün öne çıkan özelliği ise 3 farklı ölçekte tespit yeteneği olmuştur.

YOLOv4 versiyonu ile artık Darknet ağ mimarisinden vazgeçilmiş ve PyTorch kullanılmaya başlanmıştır. Buna göre K-Ortalama Kümesi kullanılarak bağlantı kutuları oluşturuldu ve GHM kaybından yaralanıldı.Bu geliştirmeler ile doğruluk ve hız oranında artış sağlandı. YOLOv3 versiyonu gibi eğitim sürecinde kullanılan tekniklerini arttırarak Self-Adversarial Training ve DropBlock Regularization gibi daha iyi öğrenme teknikleri kullanılmıştır. YOLOv4 ile gerçek zamanlı algılama öncelik haline getirilmiştir ve sadece bir GPU üzerinde eğitimini yürütür. Google Brain tarafından geliştirilen EfficientDet ile ağın boyun kısmındaki blokların en iyi formunu bulmak için sinir mimarisini kullanmıştır.

YOLOv5 algoritması 4 farklı opsiyonda gelir. Küçük,Orta, Büyük ve Ekstra Büyük şeklinde. Her bir opsiyonda doğruluk oranı artarak devam eder. Bu algoritma bir önceki versiyonunda kullanılan EfficientDet’ten daha hızlı eğitilir. Süre bakımından kazanım mevcuttur. Ekstra büyük modeli olan YOLOv5(x) neredeyse EfficientDet D4 modeline benzer bir doğruluk oranına sahiptir. YOLOv4 ile gelen Darknet -PyTorch değişimi, v5’in araştırma çerçevesine de uygulanmıştır. C dilinde yazılan Darknet çerçevesi, ağa kodlanan işlemler üzerinde ince işçilik yapılmasını sağlar. Alt seviye dilin nimetlerinden yararlanılmıştır diyebiliriz. Ancak artısı olduğu gibi yavaşlık gibi eksileri de olmaktadır. Yukarıda bahsettiğimiz Veri Artırımı işleminin Yolov5 için nasıl olduğuna bakalım;

Her eğitim grubuyla birlikte çalışan veri yükleyicisi sayesinde veriler çevrimiçi olarak arttırılır. Bu arttırma işlemini yaparken 3 adımda gerçekleşir;

1. Ölçekleme

Görüntü boyutunun artırıp azaltılması işlemini içerir.

1. Renk Alan Ayarlamaları

Görüntülerin renk dağılımını değiştirerek modelin farklı renk tonlarında nesneleri tanıyabilmesini sağlar.

1. Mozaik Artırma

Dört görüntüyü bir araya getirerek yeni bir görüntü oluşturur. Bu, modelin çeşitli görüntülerden öğrenmesini sağlar ve genel performansını artırır. Mozaik artırma, görüntülerin rastgele bir şekilde kombinlenmesiyle gerçekleştirilir.

YOLOv6 Algoritması hız ve doğruluk arasında ciddi bir denge sunar. Bu versiyon ile YOLO algoritmasına Çift Yönlü Birleştirme (BiC) modülü uygulanmış, Çapa Destekli Eğitim (AAT) ve COCO veri kümesinde doğruluğu yüksek tespitler için omurga ve boyun tasarımı geliştirmesi gibi geliştirmeler yapılmıştır. Temel olarak bu geliştirmeler ile gelen özelliklere bakacak olursak ;

* Çift Yönlü Birleştirme(Binary Combination) Modülü : Dedektörün boynuna BiC modülü eklenerek peformans kazanımları sağlamştır.
* Çapa Destekli Eğitim (Anchor Aimed Training): Bu model ile hem çapa tabanlı hem de çapasız yaklaşımların avantajlarını birleştirerek, daha geniş veri setleriyle çalışma imkanı sunar.
* Geliştirilmiş Omurga ve Boyun Tasarımı : Yüksek çözünürlüklü setlerde dahi, COCO veri kümesinde yüksek performansa ulaşır.
* Kendi Kendine Damıtma Stratejisi : Daha küçük modellerin performansını arttırmak için, eğitim sırasında yardımcı regresyon dalını geliştirmiş, belirgin hız düşüşünü engellemiştir.

YOLOv7 geliştirmeleri ile bulut teknolojisi, mobil GPU ve GPU cihazlarını destekleme gibi geliştirimeler yapılmıştır. Geleneksel optimizasyon mimarisinde gerçek zamanlı nesne algılayıcılarının aksine, eğitim sürecinin optimizasyonunu ön planda tutar. Bu şekilde maliyet artırılmadan doğruluk oranında iyileştirme amaçlanmaktadır. YOLOv7 geliştirmleleri ile gelen yeniliklere bakacak olursak;

* Model Yeniden Parametrelendirilmesi : Eğim-yayılım yolu kavramıyla farklı ağlardaki katmanlara uygulanabilir hale gelmiştir.
* Dinamik Etiket Atama : YOLOv7 versiyonu ile ilk kez gelmiştir. Modelin daha doğru ve tutarlı etiket atamasına yardımcı olur. Farklı nesne ve pozisyonlarda doğru tespite odaklanır.
* Genişletilmiş ve Bileşik Ölçekleme : Genişletilmiş ölçekleme özelliği ile görüntü boyutlarını dinamik şekilde artırarak modeli daha geniş bir veri seti ile eğitmeyi sağlar. Farklı ölçekleme faktörlerinin kombine edilmesiyle çeşitli nesnelerin pozisyonlarını bulmaya yardımcı olur.
* Verimlilik: Gerçek zamanlı nesne dedektörünün parametrelerini yaklaşık %40 ve hesaplamasını %50 oranında etkili bir şekilde azaltanilir bu sayede daha hızlı tespit ve yüksek doğruluk sağlanır.

YOLOv7(x) versiyonu ile YOLOv5(x) versiyonları karşılaştırıldığında doğruluk oranında %2.2 artış ve parametrelerde %22 oranında azalma görülmüştür.

YOLOv8 algoritması ile birlikte gelen geliştirmelere bakacak olursak, gerçek zamanlı nesne algılama performansını arttırmak için tasarlanmış çeşitli geliştirmeler mevcuttur.

* Gelişmiş Omurga ve Boyun Mimarisi: Omurgada modül değişikliğine gidilmiştir (C3 modülünden C2f modülüne geçilmesi) Boyunda ise Decoupled Head denilen Ayrı Kafa kullanımına başlanmıştır. Bu şekilde daha tutarlı tahminler yapması sağlanmıştır. Nesne olasılığı dalı kaldırılarak modelin daha sade olması sağlanmıştır.
* Çapasız Bölünmüş Ultrlytics Başlığı: Çapa tabanlı yaklaşımlara göre optimize edilmiş hız-doğruluk dengesi sunar ve çeşitli uygulamalar için kullanılabilir hale getirir.
* Çeşitli Önceden Eğitilmiş Modeller: Bir dizi önceden eğitilmiş model sunarak özelleştirilmiş bir kullanım için uygun modeli bulmayı kolaylaştırır.

Çeşitli bilgisayarlı görme görevlerini destekler( Nesne Algılama, Örnek Segmentasyon, Poz/Anahtar Nokta Algılama, Yönlendirilmiş Nesne Algılama ve Sınıflama gibi) Her model için kendi içinde optimize edilmiştir. Çıkarım, Doğrulama, Aktarma ve Eğitim gibi çeşitli çalışma modlarıyla uyumludur.

YOLOv9 sürümü ile Programlanabilir Gradyan Bilgisi (PGI) ve Genelleştirilmiş Verimli Katman Toplama Ağı (GELAN) gibi teknolojiler ilk kez kullanılmıştır. Bu teknolojilerin entegre kullanılmasıyla birlikte modelimizin öğrenme kapasitesi arttırılmıştır. Ayrıca algılama süreci boyunca önemli bilgileri saklayarak performans ve hız konusunda ciddi geliştirmeler yapmışlardır. Bu sürümde derin sinir ağlarındaki bilgi kaybının yarattığı zorluklar üzerinde iyileştirmeler yapılmıştır. Bu iyileştirmeler için Bilgi Darboğazı İlkesi ve Tersinir İşlevlerin Yenilikçi Kullanımı temel alınmıştır.

* Bilgi Darboğazı İlkesi : Bu ilke derin öğrenmede bir zorluğu ortaya koymaktadır. Veriler bir ağın ardışık katmanından geçtikçe bilgi kaybı potansiyeli artar. Kaybın en aza indirilmesi hedeflenmiştir.
* Tersinir Fonksiyonlar: Eğer bir fonksiyon ifade edildiği şekilden, herhangi bir bilgi kaybı olmadan tersine de çevrilebiliyorsa tersinir fonksiyon olarak kabul edilir. Nesne tespiti görevinde kritik rol oynamaktadır.

YOLOv10 algoritmasının geliştirilmesi ile Maksimum olmayan bastırma (NMS) ortadan kaldırılmış ve çeşitli model bileşenleri optimize edilerek bu senenin 23 Mayısında kullanıma verilmiştir. Gerçek zamanlı nesne tespit algorimaları nesne kategorilerini ve konumlarını en kısa sürede tespit etmeyi amaç edinmişlerdi. Lakin YOLO algoritmasının NMS’ye olan bağlılığı ve mimari yetersizlikler YOLO algoritmasının performansını olumsuz etkilemekteydi. YOLOv10 ile birlikte bu bağlılıklar ortadan kaldırıldı ve optimizasyon konusunda önceki sürümlere oranla ilerleme kaydedildi. YOLOv10 sürümünde 4 model ölçeği yerine 6 model ölçeği kullanıma sunulmuştur. Bunlar v10-N (son derece kısıtlı kaynak) v10-S (Hız ve doğruluğu dengeleyen versiyon) v10-M ( Genel amaçlı kullanım ) v10-B(Daha yüksek doğruluk ve genişliği arttırılmış dengeli versiyon) v10-L(Hesaplama kaynakları ile birlikte yüksek doğruluk) v10-X(maksimum doğruluk ve performans) Performans karşılaştırmalarına örnek verecek olursak YOLOv10(B), YOLOv9(C)’den %46 daha az gecikme ve %25 daha az parametreye sahiptir.

YOLO algoritmalarının sonuncusu ve en gelişmişi olan YOLOV11 ile hız ve doğruluk oranının mevcut sürümler içerisindeki en yüksek optimizasyonuna ulaşılmıştır. Öyle ki YOLOv11-M modeli YOLOv8-M ‘den %22 daha az parameter ile çalışıp çok daha yüksek doğruluk oranına sahip olmuştur. YOLOv11 modelinin çok yönlülüğü sayesinde Nesne algılama, Örnek Segmentasyon, Görüntü sınıflandırması İnsan vücudundaki önemli noktaların tespit ve takibi yönlendirilmiş nesne algılama (OBB) gibi çeşitli görevlerde kullanılabilmektedir.