Uçan Araçların (UAV) Nesne ve İniş Alanları Tespiti

*Mustafa ARU, Yazılım Mühendisliği, İstanbul Sabahattin Zaim Üniversitesi, [mustafa.aru@std.izu.edu.tr](mailto:mustafa.aru@std.izu.edu.tr), 030716006*

*Fatih KARAMAN, Yazılım Mühendisliği, İstanbul Sabahattin Zaim Üniversitesi, [fatih.karaman@std.izu.edu.tr](mailto:fatih.karaman@std.izu.edu.tr), 030716022*

*Hacı Burak TAHMAZ, Bilgisayar Mühendisliği, İstanbul Sabahattin Zaim Üniversitesi, tahmaz.haci@std.izu.edu.tr, 03117112*

*Mohamed TRAORE, Bilgisayar Mühendisliği, İstanbul Sabahattin Zaim Üniversitesi, traore.mohamed@std.izu.edu.tr, 030117024*

Proje İsmi ve Özeti

Gelecekte kullanılması yaygın olacak yapay zeka tabanlı uçan araçların yaygınlığı günden güne artmaktadır ve birçok firma bu işlemler için AR-GE çalışmalarına başlamıştır. Uçan araba ve uçan ambulanstan oluşan araçların iniş yapacakları alanların türlerine göre Uçan Araba Park (UAP) ve Uçan Ambulans İniş(UAİ) şeklinde ikiye ayrılması ön görülmektedir. Projedeki amaç, uçan araçların iniş için alt kamera sistemlerinden alınan eş zamanlı videolarıyla insanların, motorlu ve raylı taşıtların tespit edilmesi; sınıflandırılması ve alanın iniş için uygunluğunun kararlaştırılmasıdır. İnsanlar, motorlu ve raylı taşıtlar haricindeki tüm nesneler (bisikletler, hayvanlar vb.) sadece iniş alanın içerisinde olduğu durumda tespit edilecek fakat sınıflandırılmayacaktır. İniş alanın uygunluğu, içerisinde herhangi bir nesnenin bulunup bulunmamasıyla ilgilidir.

# GİRİŞ

Havayolu ulaşımı, insan yapımı hava taşıtlarıyla yapılmakta olan bir ulaşım şeklidir. Bu ulaşım şeklinde, günümüze kadar geçen süreçlerde havayolu taşıtlarını insanlar kullanmaktadır fakat teknolojinin gelişmesiyle gelecekte kullanılması yaygın olacak yapay zeka tabanlı otonom uçan araçların sayısı günden güne artmaktadır ve birçok firma bu işlemler için AR-GE çalışmalarında ilerleme katetmiştir. Bu akıllı araçlar, iniş yapılacak alanın çevresindeki nesneleri tespit edip, alanın uygunluğu hakkında bir karar vermelidir.

Projedeki amacımız uçan araçların kamera sistemlerinden alınan eş zamanlı videolarıyla insanların, motorlu ve raylı taşıtların tespit edilmesi, sınıflandırılması ve alanın iniş için uygunluğunun kararlaştırılmasıdır. Uçan araba ve uçan ambulanstan oluşan araçların iniş yapacakları alanların türlerine göre Uçan Araba Park(UAP) ve Uçan Ambulans İniş(UAİ) şeklinde ikiye ayrılması ön görülmektedir. İnsanlar, motorlu ve raylı taşıtlar haricindeki tüm nesneler(bisikletler, hayvanlar vb.) sadece iniş alanın içerisinde olduğu durumda tespit edilecek fakat sınıflandırılmayacaktır. İniş alanın uygunluğu, içerisinde herhangi bir nesnenin bulunup bulunmamasıyla ilgilidir.

# İLGİLİ ÇALIŞMALAR

Nesne tanıma konseptinde Regions with CNN features(R-CNN) günümüzde modern anlamda yapılan derin öğrenme bakış açısıyla ele alınan ilk algoritmadır [[1]](#Ref_1). Fakat bu algoritma eğitim ve test konusunda oldukça maliyetli ve günümüzdeki rakiplerine kıyasla standart dışında kalmış durumdadır. Fast R-CNN algoritması, R-CNN algoritmasının yaptığı gibi 2 bin adet “region extraction” yapmak yerine, tüm görsel için bir evrişimsel nitelik haritası çıkartarak ileri aktarımlı ağ yapısı kullanmaktadır [[2]](#Ref_2). Fakat yine de bu model “regional proposal algorithm” tarafından dar boğaza sokulmaktadır ve buna da alternatif olarak Faster R-CNN modeli ortaya çıkmıştır [[3]](#Ref_3). Bu model bu darboğazdan kurtulmak için bu “Region proposel method”u artık kullanmamayı ve tam bir derin öğrenme modeli oluşturmuştur. RPN ve Fast R-CNN dedektörü bu modeli oluşturan ana iki modüldür. Bu model günümüzde de oldukça popüler bir şekilde kullanılmakla beraber, tespit edilen nesnelerin segmentasyonunu yapamamakta, her nesnenin bulunduğu belirli pikselleri seçememekte yalnızca sınırlayıcı kutu koyabilmektedir. Mask R-CNN [[4]](#Ref_4) Facebook AI Research tarafından Segmentasyonla beraber sınırlayıcı kutu ve sınıflandırma yapabilen bir model olarak geliştirilmiştir. Bu her bir nesneyi ayrı ayrı olarak görselden çıkartabilme kabiliyetine sahiptir. Daha önceden Fast R-CNN’de tanıtılan RoI pooling katmanı RoIAlign ile değiştirilmiştir. Burada quantalama yaparken kayan sayıların ondalık değerlerini nitelik haritası çıkarması kolaylaştırılması hedeflenmiştir. Bir diğer literatür taramalarımı sonucu elde ettiğimiz algoritma ise YOLO’ dur [[5]](#Ref_5). Yolo adını “You Look Only Once” yani, yalnızca bir kez bakabilirsin kısaltmasından alır. Günümüzde de hala en popüler algoritma olan YOLO, C ve CUDA’da [[6]](#Ref_6) yazılmış açık kaynak nöral ağ framework’ün Darknet içinde tasarlanmıştır [[7]](#Ref_7). YOLO günümüzde hala maliyet ve doğruluk oranı olarak en makul algoritmalardandır. Fakat YOLOv3 8x8’den küçük nesneleri tanıyamamakta dahası bundan çok az büyük olan nesneleriyse bilgi yetersizliğinden tanıyamamaktadır [[8]](#Ref_8). Bu çalışmalara bakıldığında bizim amacımız en optimum modeli elde etmek olacaktır.

# VERİ SETİ VE ÖZELLİKLERİ

Veri setimiz içerisinde gökyüzünden yere doğru çekilmiş olan fotoğraflar mevcuttur. Bu bağlamda literatür taramalarımızın sonucunda elde ettiğimiz açık kaynaklı drone çekimlerinden oluşan Visdrone, Stanford, CarPark, DOTA gibi veri setlerinden faydalanarak kendi veri setimizi oluşturduk. Şu an elimizdeki veri seti üç bin sekiz yüz yirmi (3820) görüntüden oluşmaktadır. Kendi veri setimizi oluştururken farklı kaynaklardan faydalanmamızın temel sebebi farklı yerlerden çekilmiş görüntüleri barındırmasıdır. Bunun yanı sıra veri setimizde bulunan görüntüler için dikkat ettiğimiz özellikler başlıca çekimin açısı, yere olan uzaklık, nesnelerin çeşitliliği (tır, otomobil, insan, motosiklet vb.) ve çekimlerin yapıldığı hava koşulları gibi durumlardır. Çekimlerin yere olan uzaklığı 25 ile 100 metre arası**,** en fazla 20 derecelik açıyla, aydınlık ve açık havada olması gerekmektedir. Bu durumlar dışındaki hiçbir ihtimal veri setimizde mevcut değildir. İniş alanları için literatürde var olan bir veri seti olmadığından dolayı, veri setinde bulunan görsellerin üzerine iniş alanları ekleyerek bir çeşit sentetik veri elde edilmiştir. Sentetik veri oluşturma işlemleri üzerinde veri artırma yapmak için görselin açısı, renk ölçeği ve iniş alanı boyutu gibi özellikler değiştirilmiştir.

# VERİ ÖN İŞLEME

## Veri Boyutlandırma

Öncelikle verilerin boyutlarının modele uygun hale getirilerek yeniden boyutlandırılması yapılmıştır. Çekim yapılan cihazların türleri, çekim açıları vb. faktörler göz önüne alındığında, veri setinin içindeki bazı verilerin farklı boyutlarda olduğu gözlemlenmiştir. Bu aşamada incelenen verilerin boyutlarının ihtiyaca göre küçültüp büyütülerek modele, aynı boyutta görsel verilmesi sağlanmıştır.

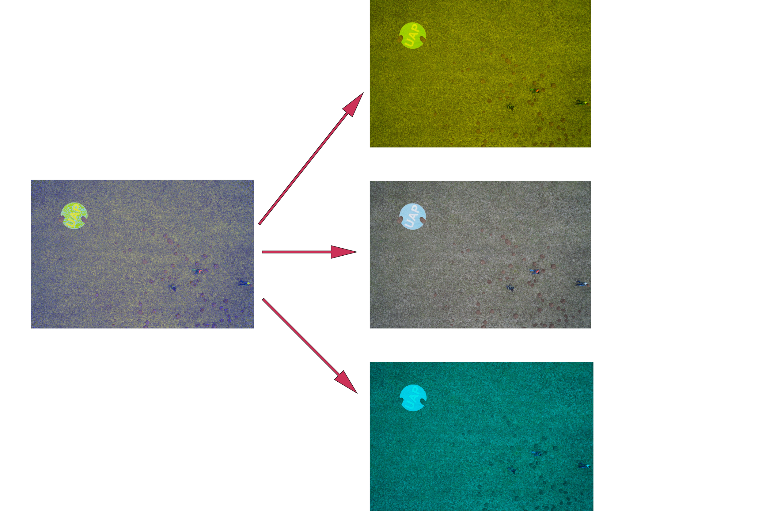
Graphical user interface, application

Description automatically generated

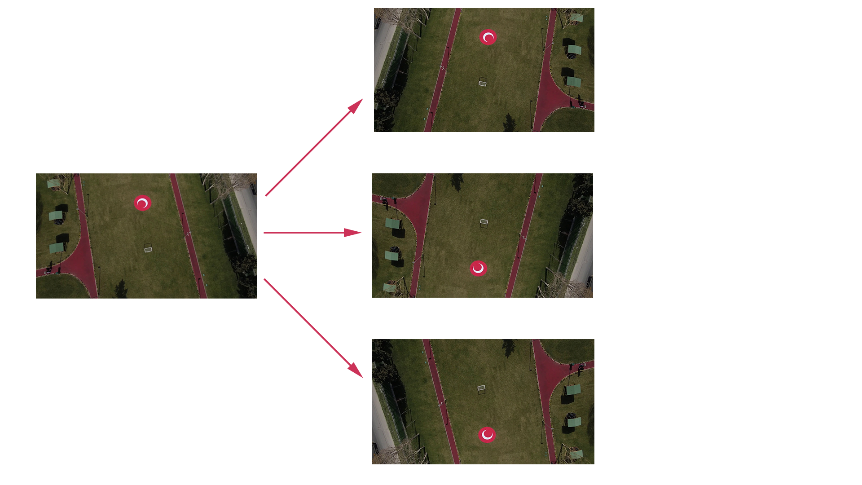
Şekil 1. Veri Boyutlandırma

## Veri Artırma

Veri artırma mevcut veriler kullanılarak yeni eğitim verileri oluşturulmasıdır. Bu işlem sonunda modelin eğitiminde daha fazla veri kullanılarak modelin doğruluk oranının artırılması hedeflenmiştir. Model için eğitilecek veri setinin döndürme, şekil değiştirme, ışıklandırma, eğme gibi yapılacak birçok değişikliklerle veri sayısı artırılmıştır. Buna ek olarak, veri setimizi oluştururken açık kaynaklarda bulunmayan iniş alanlarının veri arttırma işlemi de burada yürütülmüştür.



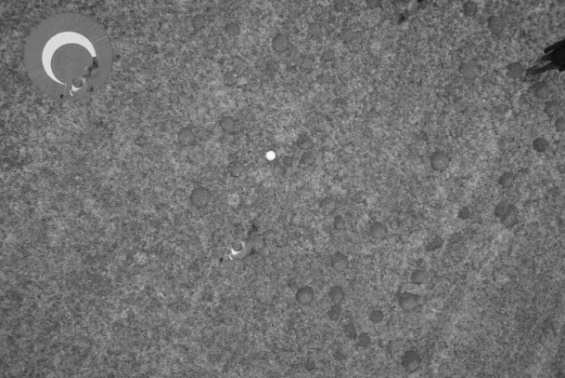
Şekil 2. Renk Değiştirerek Veri Artırma



Şekil 3. Döndürme İşlemiyle Veri Artırma

## Veri Normalizasyonu

Kullanacağımız görseller, renkli olmakla birlikte piksel değerleri 0 ile 255 arasında yer almaktadır. Bu piksel değerleri yapay sinir ağları için uygun değerler olması adına [0,1] arasındaki değerlere standardize edilecektir [[9]](#Ref_9).



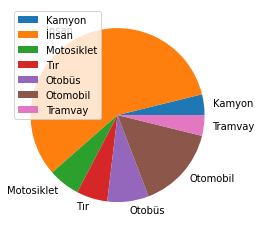
Şekil 4. Görüntünün Normalize Edilmesi

## Veri Etiketleme

Etiketleme işlemini yapabilmek için [https://cvat.org/](https://cvat.org/" \t "_blank" \o "https://cvat.org/) web uygulamasını ve “LabelImg” isimli bir araç kullanılmıştır. Bu araç, kullanılan model için tam olarak istenen sonuçlar veremediği fark edilmiştir. CVAT ise bir web uygulaması olup, eş zamanlı olarak çok sayıda insanın aynı etiketleme projesinde çalışabilmesinden ötürü bu noktada sadece CVAT üzerinde ilerleme karar kıldık.Model tarafından belirlenmesi gereken nesneler tespit edilmiş ve etiket olarak sisteme kaydedilmiştir. Veri setleri üzerine özenle çizilen kareler sayesinde nesne belirlenmiş ve karenin etiketi yazılmıştır. Örnek etiketleme işlemi görseli aşağıda verilmiştir [[10]](#Ref_10).



Şekil 5. Veri Etiketleme



Şekil 6. Etiketlenen Nesneler

## Verileri Eğitim ve Test İçin Ayırma

Eğitimde kullanılan verilerle modeli test etmek, modelin daha öncesinde gördüğü verileri tekrardan yorumlaması demek olmakla beraber anlamlı sonuçlar elde edilmemesi demektir. Bu bağlamda veriler eğitilecek model için hazırlandıktan sonra son aşama olarak verilerin eğitim ve test için iki parçaya bölünmesi gerekmektedir. Eğitim ve Test için ayırma işleminde oranı belirlemek ve doğru oranı seçmek önemlidir. Literatür taramalarında veriler genellikle %80 eğitim, %20 test olarak ayrılmıştır. Bu proje de bu oranları bizler de uyguladık, ileriki aşamalarda değiştirilerek modelin iyileştirme performansı kontrol edilecektir [[11]](#Ref_11).

# VERİ KEŞFİ VE GÖRSELLEŞTİRME

## Veri İnceleme

Veri keşfi konusunda yaptığımız çalışmalar sonucunda [[12]](#Ref_12) toplanan tüm verilerin ekrana basılması ve görüntülenmesi işlemi burada yapılmıştır. Veri seti içerisinde mevcut veriler arasına karışmış, projeden bağımsız verilerin olmadığının tespit edilmesi için tüm verileri hızlı bir şekilde gözden geçirmek için kullanılmıştır ve bu noktada model için uygunluğuna karar verilmiştir. Çeşitli şartlara bağlı bu verilerin uygun olanları iyileştirmek üzere saklanmıştır. Uygun olmayan veriler, veri ayıklamada da bahsedildiği üzere değerlendirmesi o aşamada dikkate alınır.

## Veri Görselleştirme

Klasik formatlarda gösterilen verileri ilk bakışta anlamak zor gelmektedir. Bu sebepten dolayı verinin görselleştirilmesi aşamasına ihtiyaç duymaktayız. Bu aşamada, incelenen verilerin anlaşılır kılınması için çeşitli araçlar sayesinde göze hitap edecek şekilde veriler görselleştirilmiştir. Python ile veri görselleştirme aşamasında genel olarak kullanılan kütüphane Matplotlib kütüphanesidir fakat genelde Matplotlib ile görselleştirme nispeten fazla kod istemektedir ve ilkel kalmaktadır. Bu projede Seaborn [[13]](#Ref_13) kütüphanesi kullanarak veriler görselleştirilmiştir.



Şekil 7. Veri Setindeki Görüntüler

## Veri Ayıklama

Veri seti ve özellikleri aşamasında ideal veri setinden bahsedilmiştir. Burada belirtilmiş olan kurallara göre verileri incelenmiş olup uygun gördüklerimizi i veri setinde tutmaya devam edip uygun olmadığını düşündüğümüz verileriyse veri setinden ayırdık. Bu verilerin modelin eğitim, doğrulama ve test süreçlerinde kullanılmamasıyla modeli olumsuz etkileyecek durumlardan kaçınılmıştır.



Şekil 8. Hatalı Örnek Görsel

## Veri Seti Şekillendirme

Ayıklanan verilerin eksiklik durumları incelenir ve diğer etiketlere göre eksik nesnelerin belirlenerek eklemeler yapılmasıdır. Örneğin araba verisi bisiklet verisine göre oldukça fazla olmasından dolayı model bisiklet tanımlama konusunda başarılı olmayacaktır. Bu sebepten dolayı modele bisiklet verisi eklenir ve nesnelerin oranları korunur. Böylelikle model daha stabil bir şekilde eğitilir ve daha iyi sonuçlar alınır.

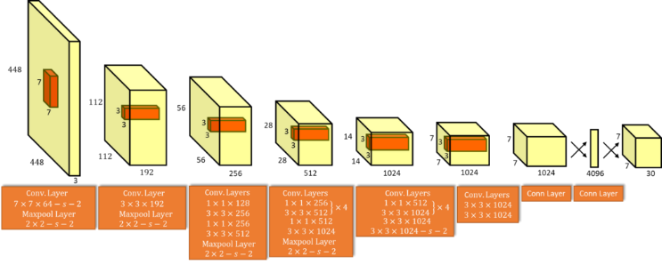
A screenshot of a video game

Description automatically generated with medium confidence

Şekil 9. Eksik Nesnelerin Arttırılması

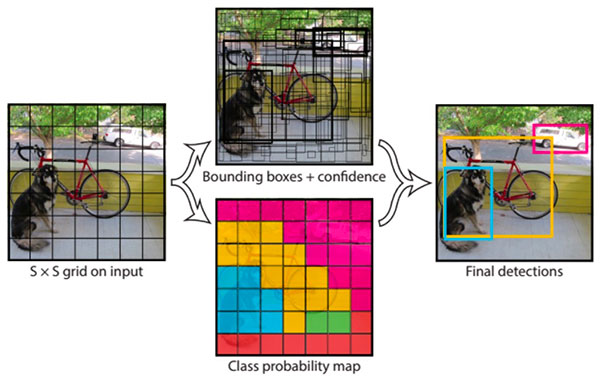
# YÖNTEMLER

Projede, bir derin öğrenme probleminin ele alınmasının yanı sıra spesifik olarak nesne tanıma özelinde bir çalışma yürütülmüştür. Bu yüzden klasik derin öğrenme algoritmalarının veya metodlarının yerine modern nesne tanıma algoritmalarından faydalanılmıştır. Bu doğrultuda ilk olarak YOLO algoritmasıyla probleme çözüm aranmıştır.



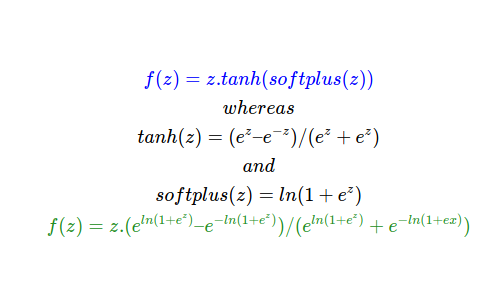
Şekil 10. Yolo Mimarisi

YOLO, gerçek zamanlı olarak nesne tanıma için evrişimli ağları kullanan bu algoritma R-CNN tabanlı algoritmalardan farklı olarak girdi görüntü üzerinde belirli bölgeleri önermek sonradan tahminleme yapmak yerine tek bir ileri yayılım ile girdi görüntünün bütün bölgeleri üzerinde (farklı ölçeklerde) kutu(bounding box) ve sınıf (category or class) tahminleme yapar. Ardından yapılan tahminlemelerde elde edilen sonuçlarda non-maximum suppression (maksimum olmayan bastırma) yöntemini kullanarak farklı boyutlu kutulara ve farklı olasılık değerlerine sahip olan nesneleri için en yüksek olasılığa ve IoU değerine sahip olan kutu ve sınıf seçilir. YOLO algoritması 3 ana teknik kullanarak çalışmaktadır. Residual blocks, bu aşamada girdi görüntü SxS boyutlu gridlere ayırılır. Bounding box regression, bu aşamada ise oluşturulan her gridin yani sınırlayıcı kutunun özellikleri belirlenir. Kutunun en ve boyu (wx, hw), merkezi (bx,by), sınıfı(nesne türü) ve olasılığı (sınıfın doğruluğu) Intersection Over Union (IOU), bu aşamada ise her kutu için kesin referansa göre IoU metriği değeri hesaplanır ve buna göre non-maximum suppression uygulanır ve son sonuçlar elde edilir.



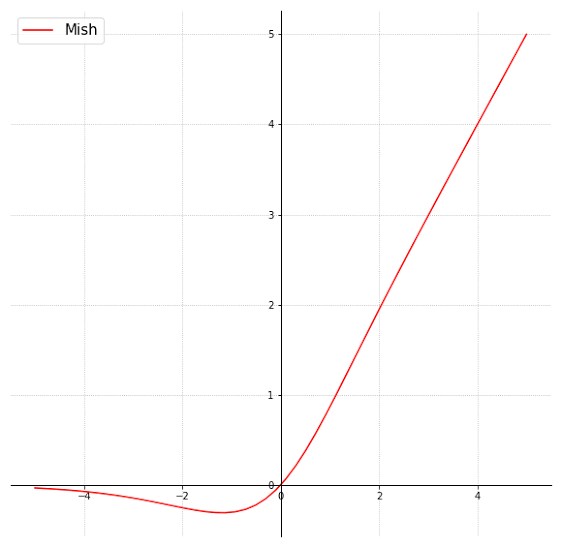
Şekil 11. En uygun kutu bulma işlemi

Benzer şekilde, YOLO nesnelerin genelleştirilebilir hallerini öğrenir, böylece doğal görüntüler üzerinde eğitildiğinde, algoritma diğer nesne tespit yöntemlerinden daha iyi göstermektedir. YOLOv3 lojistik regresyon kullanarak her sınırlayıcı kutu için nesnellik puanı tahmin etkmektedir. Sınırlayıcı kutu “ground truth” olan noktayla kesiştiği durumlarda bir olarak işaretlenir. Sınıflandırma için, her kutu, çok etiketli sınıflandırmayı (multi label classificaition) kullanarak sınırlayıcı kutunun içerebileceği sınıfları tahmin eder. Softmax, performans göz önünde bulundurulduğu için kullanılmamaktadır. Öznitelik çıkarımı içinse bir önceki sürümlere göre yeni bir çözüm kullanmaktadır. YoloV4, yenilenen omurgası (backbone) başta olmak üzere getirdiği yeni özellikler sayesinde Yolov3’e göre MsCOCO veri seti üzerinde mAP değeri %10 FPS ise %13 daha iyi sonuç elde etmiştir. Bu özellikler;Cross Stage Partial Connections(CSP): Evrişimli Sinir Ağının öğrenme kapasitesini artıran yeni bir omurga. Self-adversarial-training (SAT): İleri ve geri yayılımda çalışan yeni bir veri artırma yöntemidir. Mosaic data augmentation: 4 tane eğitim görüntüsünü karıştırarak yeni görüntüleri üreten yeni bir veri artırma yöntemidir. Bu veri artırma yöntemi küçük nesneleri tanımada yolov4’ü diğer yolo versiyonlarından daha başarı kılmıştır. DropBlock regularization, evrişimli sinir ağı için iyi bir regularizasyon yöntemidir. Mish-activation: Kendi kendini düzenleyen, monotonik olmayan yeni bir aktivasyon fonksiyonudur.



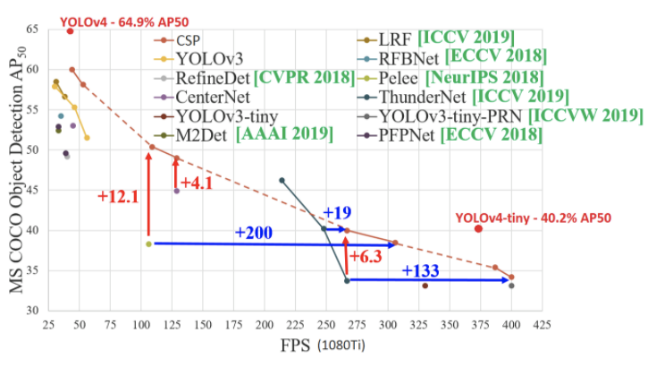
Şekil 12. Mish aktivasyon fonksiyonu formülü

Bu fonksiyon hiperbolik tanjant ve softplus aktivasyon fonksiyonların kombinasyonuyla oluşturulmuştur. Hesaplanması çoğu aktivasyon fonksiyonuna göre daha maliyetli olmasına rağmen Mish aktivasyon fonksiyonu sayesinde ReLU (Rectified Linear Unit, r(x) = max(x,0)) aktivayon fonksiyonuyla yaşanan ağdaki negatif nöronların iptal edilmesi ve gradyan kaybı sorunları ortadan kaldırılmıştır.



Şekil 13. Mish fonksiyonu

YOLOv4-Tiny algoritması YOLOv4’ün bir sıkıştırılmış versiyonu olarak düşünülebilir. YOLOv4’e göre çok daha hızlı eğitim ve test olanağı sunmaktadır. Bununla beraber küçük nesnelerin tanınmasında çeşitli sorunlar çıkarmaktadır. Hızın, doğruluktan daha değerli olduğu durumlarda YOLOv3 veya YOLOv4’ü kullanmak yerine bu modeli kullanmak daha uygun olabilmektedir.



Şekil 14. Kullanılan algoritmaların karşılaştırılması

Bu çalışmada Darknet framework’ünden faydalanılarak YoloV3, YoloV4 ve YoloV4-Tiny kullanılmıştır. YoloV3 ve YoloV4’ü aynı model için iki farklı sonuç alıp ikisi arasında değerlendirme için, YoloV4-Tiny’i ise daha az karmaşık olan verilerin kullanıldığı modeli eğitmek için tercih edilmiştir. Bu sade veriler aslında diğer verilerden sadece etiket yönünden (iniş alanları için artırılmış veriler hariç) farklılık göstermekle birlikte, iniş alanlarının tespiti ve bunların uygunluk durumlarını tespit etmek amacıyla kullanılmıştır. Fotoğrafların aynı olmasına karşın sadelik oluşturmasının sebebi; birinde araç ve insanların etiketlenmiş olması(çok sayıda etiket) diğerindeyse sadece iniş alanlarının etiketlenmiş olmasıdır.

# DENEYLER, SONUÇLAR, TARTIŞMA

İlk olarak en hızlı eğitilen YoloV4 mimarisi olmuştur. Yaklaşık 5 saatlik eğitimin ve 2000 yineleme sonrasında, sonuç olarak arabaların tespiti yapılmıştır.



Şekil 15. YoloV4 mimarisine ait sonuçlar

Bir sonraki eğitim ise YoloV4-Tiny mimarisine aittir. Yaklaşık 8 saatlik eğitimin ve 6000 yinelemenin ardından iniş alanlarının uygunluğu konusunda karar verme işlemi başarıyla gerçekleştirilmiştir.



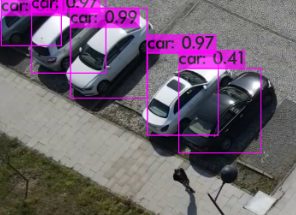
Şekil 16. YoloV4-Tiny mimarisine ait sonuçlar

Bir sonraki eğitim ise YoloV3 mimarisine aittir. Yaklaşık 5 saatlik eğitimin ve 4000 yinelemenin ardından iniş alanlarının uygunluğu konusunda karar verme işlemi gerçekleşmiştir. Bu işlem YoloV3 ve YoloV4-Tiny birleşimi sonucu elde edilen modeldir. Hem insan ve taşıt hem de iniş alanını başarılı bir şekilde tespit etmektedir.



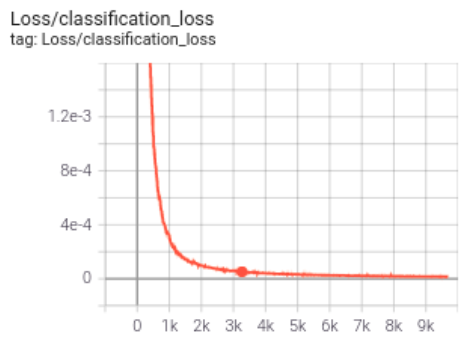
Şekil 17. YoloV3 ve YoloV4-Tiny mimarilerine ait sonuçlar

2 adımda çalışan algoritmalardan FASTER R-CNN algoritması denenmiştir. Ortalama olarak iyi sonuç alınmasına rağmen fps sorunu yaşanmıştır. Gerçek zamalı probemler için uygun olunmadığı fark edilmiş ve bu modelden vazgeçilmiştir. Ana model için (insan ve taşıt) denenen modellerden biri de yolov3-spp algoritmasıdır. Bu uygulamanın sonucunda elde edilen model arabaları iyi bir şekilde tanımasına rağmen insanları hemen hemen hiç tanımamıştır.



Şekil 18. YoloV3-SPP İnsan tanımama, taşıt tanıma

Sonuç olarak iniş alanlarının uygunluğunun tespit edilmesi için YoloV4-Tiny mimarisi daha hızlı, tutarlı ve doğru bir sonuç vermiştir. Nihai proje amacını yerine getirmek için uygun bir mimaridir. İnsanların ve taşıtların tespit edilmesi içinse YoloV3 mimarisi daha başarılı olmuştur. Eğitim konusunda yavaş olmasına ragmen sunduğu başarı oranı sayesinde kullanım için yeterlidir. Uçan araçtan alınan video frame sayısından daha hızlı tespit etmesinden dolayı bu gecikme, diğer modellere göre yavaş fakat projenin amacına uygunluğu konusunda yeterlidir.



Şekil 19. FRCNN modelinde loss function

Projenin daha tutarlı çalışması için etiketli verilerin arttırılması, kullanılmayan veri arttırma yöntemlerinin kullanılması ve daha fazla yinelemeli modelin eğitilmesi gerekmektedir. İniş alanları için herhangi bir veri kaynağı olmamasından dolayı veri üretme aşamasında çokça zaman almasından dolayı ilgili kaynaklar yeteri kadar dağıtılamamıştır.

# KATKILAR

Takımdaki her kişi her aşamada aktif şekilde rol almıştır. Belli alanlara ağırlık verilmesi için iş bölümü yapılmıştır. Yapılan iş bölümü aşağıda ifade edilmiştir.

Mustafa ARU, model için verilerin etiketlenmesi ve veri inceleme aşamasına ağırlık vermiştir. Raporlama alanında Latex kullanılarak ön raporun son haline getirilmesinde görev almıştır. CVAT websitesinde(http://cvat.org/) belirlenen etiketler yardımıyla görüntülerin üzerinde kareler oluşturularak etiketleme yapılmıştır. Verilerin eğitim ve test verileri şeklinde ayrılmasına, raporlama ve proje takvimi hakkında da katkı sağlamakla beraber YOLOv3 modelinin eğitim ve düzenlemeleri hakkında yardımcı olmuştur.

Mohamed TRAORE, Seaborn, matplotlib ve easyimage python kütüphanelerini kullanarak, veri görselleştirme aşamasına ağırlık vermiştir. Ek olarak veri etiketleme ve normalize etme aşamalarında, local makine üzerinde çalışan projeyi colab ortamına taşınması ve iki farklı modelin eşzamanlı olarak çalışabilmesi üzerinde çalışmıştır. Son olarak model oluşturma konusunda ve model eğitimlerinde görev almıştır.

Hacı Burak TAHMAZ veri ayıklama, veri seti şekillendirme ve veri etiketleme aşamalarına ağırlık vermiştir. CVAT websitesi kullanılarak verilerin etiketlenmesi, verisetinin siteye yüklenmesi ve kişilere, etiketleme görevlerini ataması için altyapıyı oluşturmuştur. Hazır bir model için veriler test edildi, sonuçlara göre verilerde bulunan eksik nesnelerin belirlenmesinde katkı sağlamıştır. YOLOv4 eğitiminde görev almıştır fakat çok uzun süren bir eğitim colab üzerinde alınan hatalar ve oturum süresinden dolayı başarısız sonuçlanmıştır.

Fatih KARAMAN veri inceleme, veri boyutlandırma ve veri artırma aşamalarına ağırlık verilmiştir. Photoshop gibi görüntü düzenleme araçları kullanılarak görsellerin üzerine iniş alanları eklendi. Ek olarak iniş alanların üzerine çeşitli nesneler eklenerek yeni fotoğraflar oluşturulmuştur. Tüm görüntülerin renk aralıklarının ve eklenen nesnelerin derecelerinin ayarlanmasında da görev almıştır. YOLOv4-tiny modelinin configrasyonu ve eğitilemesinde görev almıştır.

# REFERANSLAR

1. <https://arxiv.org/abs/1311.2524>

2. <https://arxiv.org/abs/1504.08083>

3. <https://arxiv.org/abs/1506.01497>

4. <https://arxiv.org/abs/1703.06870>

5. <https://arxiv.org/abs/1506.02640>

6. <https://arxiv.org/abs/1506.02640>

7. <https://pjreddie.com/darknet/yolo/>

8. <https://onlinesources.izu.edu.tr:2102/document/9213538>

9. <https://machinelearningmastery.com/how-to-normalize-center-and-standardize-images-with-the-imagedatagenerator-in-keras/>

10. <https://cvat.org/>

11. <https://blog.roboflow.com/train-test-split/>

12. <https://neptune.ai/blog/data-exploration-for-image-segmentation-and-object-detection>

13. <https://www.veribilimiokulu.com/seaborn-ile-veri-gorsellestirmesi/>