2. Proje Mevcut Durum Değerlendirmesi

Ön tasarım raporunda algoritmalar ve ön işleme konularının isabetli belirlendiği doğrulanmıştır. Algoritma listesine tek-aşamalı nesne tespit algoritması olan RetinaNet eklenmiştir. Yeniden eğitilen bir RetinaNet modeli ile Uçan Araba Park Alanı ve Uçan Araba İniş Alanı noktalarının tespit etme oranında etkili bir artış sağlanmaya çalışılmıştır. Mimari tanımı ön tasarım raporuna göre detaylandırılmıştır. RetinaNet modeli az sayıda resim ile yeniden eğitilmiştir. YOLO modeli dronelar aracılığıyla çekilmiş videolardan elde edilen ve yaklaşık 5000 resimden oluşan etiketlediğimiz veri seti ile eğitilmiştir.

3.Algoritmalar ve Yazılım Mimarisi

3.1 Algoritmalar

3.1.1 Transfer Öğrenme ve Veri Artırımı Yöntemleri

Yapay zekâ projelerinde en önemli kısımlardan birisi olan veri setini hazırlamak, veri setini kullanarak eğitilen algoritmaların doğruluk oranını bire bir etkilediğinden en çok zaman alan işlemlerden birisidir. Gereken veri miktarını azaltmak ve başarı oranını arttırmak amacıyla ‘’transfer öğrenme’’ tabanlı yöntemler kullanılac aktır. Eğitim aşamasına başlanıldığındaalgoritmanın içindeki ağırlıklar (weights) standart olarak rastgele atanır. Hazırlananveri kümesini rastgele ağırlıklar ile başlayarak eğitmek yerine önceden eğitilmiş bir modelin katmanları kullanılacaktır.

3.1.2 RetinaNet

Bir modeli sıfırdan eğitmek yerine tek aşamalı detektör olan RetinaNet [1] tanımı ve kontrol noktaları transfer öğrenme için kullanılmıştır. RetinaNet, eğitim sırasında sınıf dengesizliğini gidermek için bir odak kaybı işlevi kullanan tek aşamalı bir nesne algılama modelidir. RetinaNet, bir omurga ağından ve göreve özel iki alt ağdan oluşan tek, birleşik bir ağdır. Omurga ağı genişledikçe isabeti artmakta fakat çıktı hızı azalmaktadır. RetinaNet çok az eğitim verisiyle nesne tespiti yapabilmektedir. Yarışma kapsamında sağlanan örnek eğitim videosu, model eğitiminde kullanılabilir.

3.1.3 Yolo

YOLO, sınırlayıcı kutuları uzamsal olarak ayırmak ve sınıf olasılıklarını ilişkilendirmek için nesne algılamayı bir gerileme problemi olarak çerçeveler.

YOLO, işlenmemiş görüntülerden doğrudan nesnelerin sınırlayıcı kutularını ve sınıflarını tahmin eden tek bir sinir ağı kullanır.

3.2 Yazılım Mimarisi

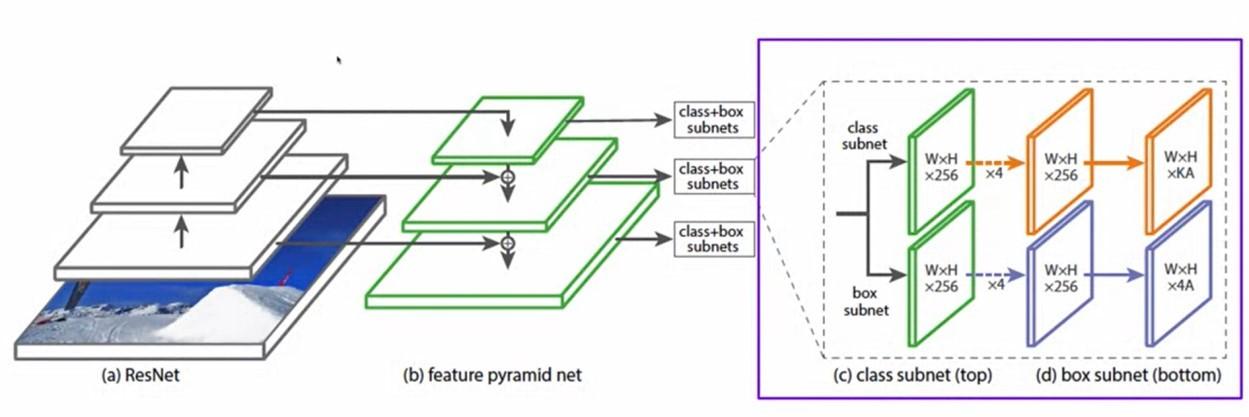
Mimari yazılım mimarisi ve sistem mimarisi olarak iki farklı şekilde hazırlanmıştır. Yazılım mimarisi, veri setinin hazırlanması ve model eğitimini içermektedir. Sistem mimarisi, modele girdi olacak görsellerin hazırlanması ve modelin üretmiş olduğu sonuçları yarışma standardına uygun olarak JSON nesnesinin oluşturulmasını sağlamaktadır.

Yapay zekâ projelerinde veri kümesini hazırlamak, en çok zaman alan ve en önemli noktalardan biridir. Eğitilen algoritmaların yüksek doğrulukta nesne tespiti yapabilmesi için çok miktarda veri gerekmektedir. Veri miktarını artırmak için veri artırım modülü kullanılmaktadır. Veri artırım modülü, görsellerine ayrılmış olan videoların her bir görseli için döndürme, aynalama, ölçeklendirme, kırpma, gürültü ekleme, görselin farklı hava olaylarındaki karşılığını üretme ve gece-gündüz karşılığını üretme metotlarını kullanarak veri miktarını artırmaktadır. Oluşturulan veri seti Ericsson EVA [20] yazılımı ile etiketlenmiştir. Yazılım, etiketlenmiş olan veri setini YOLO standardında indirilmesini sağlamaktadır.

4. Özgünlük

İnsan ve araç içeren birçok açık kaynak veri seti bulunmaktadır. Bu veri setleri Uçan Araba Park Alanı (UAP) ve Uçan Araba Acil İniş Alanı (UAI) bilgilerini içermemektedir. İniş alanlarını tespitini mümkün kılmak için UAP ve UAI resimleri belirlenen veri setindeki bazı resimlere gömülecek ve etiketlenecektir. Veri seti hazırlama sürecinde veri artırımı yöntemleri kullanılarak veri seti genişletilecektir. Özellik çıkartma (Feature Extraction) için önceden eğitilmiş ResNet50, RetinaNet, darknet53.conv.74 modelleri transfer öğrenme yöntemi ile modellere eklenecektir. Doğruluk oranını artırmak için görüntü iyileştirme yöntemleri veri setine uygulanacaktır. Farklı algoritmalara ait modeller ilk aşamada bu veri setleri üzerinden eğitilecektir. Eğitilen modeller örnek eğitim videosu kullanılarak test edilecektir. Tespit başarısının tatmin edici düzeyde olmaması durumunda kendi verilerimiz drone ile toplanacaktır. Bu veri seti elle etiketlenecektir ve önceki modelleri eğitmek için kullanılan işlemler tekrarlanacaktır. UAP ve UAI alanları üzerinde nesne olması durumunu tespit etmek için ayrıca bilgisayarlı görü yöntemlerinden faydalanılacaktır. Bu yöntemler görüntü işleme ile kusur tespiti ve yapay zekayla kusur tespiti olacaktır. Yağmurlu, sisli vb. durumlarda nesne tespit başarısını artırmak için görüntü iyileştirme yöntemleri <kullanılacaktır. Bunun için öncelikle görüntüde yağmur ve sis seviyesi tespit edilmesi gerekmektedir. Program nesne tespitine başlamadan önce duruma göre dinamik olarak görüntü iyileştirme yapacaktır. Nesnelerin kısmi gözüktüğü karelerde başarılı nesne tespiti için veri setine nesnelerin kısmi gözüktüğü örnekler eklenecektir.

5. Sonuçlar ve İnceleme



Çıktı katmanında, bir sınıflandırma alt ağı ve sınırlayıcı kutunun konumunu tahmin eden bir kutu alt ağı vardır. Araştırmamızda, veri kümemizde yeniden eğitilecek sınıf alt ağı dışında her şeyi olduğu gibi bırakarak RetinaNet'ten öğrenimi sağladık. Öncelikle modeli tanımlayan konfigürasyon dosyası kullanılacaktır. Model daha sonra kayıtlı kontrol noktaları ile başlatılacaktır. Bu şekilde, model mimarisi eğitilmiş ağırlıklardan ayrıştırılır. Elimizde bulunan görsellere açıklama eklemek için colab\_utils.annotate fonksiyonu kullanılıyor. Bu yöntem, yalnızca az sayıda görüntüyü etiketlememiz gerektiğinden yeterli olacağını düşündük. İlk aşama, bu veri kümeleri üzerinde farklı algoritmaların modellerini eğitecektir. Eğitilen model, örnek eğitim videoları ile test edilecektir. Tespit başarısız olursa, kendi verilerimiz dronlar tarafından toplanılacaktır. Bu veri seti manuel olarak etiketlenecek ve önceki modeli eğitmek için kullanılan süreç tekrarlanacaktır. UAP ve UAI bölgelerindeki nesnelerin varlığını tespit etmek için bilgisayarlı görme yöntemleri de kullanılacaktır. Bu yöntemler görüntü işleme ile hata tespiti ve yapay zeka ile hata tespiti olacaktır. Yağmurlu günler, sisli günler vb. Bu durumlarda, nesne algılamanın başarı oranını artırmak için görüntü iyileştirme yöntemleri kullanılacaktır. Bu sahte tahmin edici, bir TensorFlow kontrol noktası örneği olarak kullanılacaktır. Modelin yeniden eğitim aşamasına girilmesi: Önceden eğitilmiş retina modelinin eğitim değişkenleri arasında ince ayar yapılması gereken bazı değişkenler vardır ve değişkenlerin önceden eğitilmiş ağırlıkları ile olduğu gibi kalması gerekir. Hangi değişkenleri ince ayar yapmak istediğimizi kendi verilerimizle seçtik ve bir listeye kaydettik. Öğreticide kullanacağımız her görüntü için: önce onu ön işleme tabi tutuyoruz ve ardından sınıf ve sınırlayıcı kutular dahil olmak üzere bu görüntüler üzerinde tahminler yapmak için modeli kullanıyoruz. Arabanın etrafına çizdiğimiz sınırlayıcı kutuyu kullanarak, modelimizin gerçek etiketlere kıyasla ne kadar iyi performans gösterdiğini ölçmek için bir kayıp hesaplandı. Bu kayıpların gradyanları, ayarlamayı seçtiğimiz eğitilebilir ağırlıkların hangi yönde güncelleneceğini belirlemek için hesaplanır. Son olarak, ağırlıkları güncelleyerek model eğitilmiştir.

.