T.C.
BİTLİS EREN ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI
YÜKSEK LİSANS TEZİ

NESNELERİN İNTERNETİ CİHAZLARDA DERİN ÖĞRENME KULLANILARAK EYLEM ALGILAMA

Ahmed Yaseen Bishree AL-ANI

ŞUBAT 2021

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI
YÜKSEK LİSANS TEZİ

NESNELERİN İNTERNETİ CİHAZLARDA DERİN ÖĞRENME KULLANILARAK EYLEM ALGILAMA

Hazırlayan
Ahmed Yaseen Bishree AL-ANI

Danışman
Prof. Dr. Sabir RÜSTEMLİ

Jüri Üyeleri
Prof. Dr. Sabir RÜSTEMLİ
Doç. Dr. Mehmet Nuri ALMALI
Dr. Öğr. Üyesi Kubilay DEMİR

ŞUBAT 2021

ONAY

Ahmed Yaseen Bishree AL-ANI tarafından hazırlanan "**Nesnelerin İnterneti Cihazlarında Derin Öğrenme Kullanılarak Eylem Algılama**" adlı tez çalışması 18/02/2021 tarihinde yapılan sınavla aşağıdaki jüri tarafından oybirliği/oyçokluğu ile Bitlis Eren Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Prof. Dr. Sabir RÜSTEMLİ (Danışman)	_____
Doç. Dr. Mehmet Nuri ALMALI (Üye)	_____
Dr. Öğr. Üyesi Kubilay DEMİR (Üye)	_____

Bu tezin kabulü, Lisansüstü Eğitim Enstitüsü Yönetim Kurulu'nun …/…/…gün ve …/… sayılı kararı ile onaylanmıştır.

Prof. Dr. Zeki ARGUNHAN
Enstitü Müdürü

BİTLİS EREN ÜNİVERSİTESİ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
YÜKSEK LİSANS / DOKTORA TEZ ÇALIŞMASI
ETİK BEYANI

Bitlis Eren Üniversitesi Lisansüstü Eğitim Enstitüsü tez yazım kılavuzuna göre hazırlamış olduğum "**Nesnelerin İnterneti Cihazlarda Derin Öğrenme Kullanılarak Eylem Algılama**" adlı tezimin özgün bir çalışma olduğunu, tez hazırlanırken tüm aşamalarda bilimsel etik ilkelerine uygun davrandığımı, tez kapsamında sunulan tüm verileri bilimsel etik ilkelerine uygun elde ettiğimi, tezde faydalandığım tüm eserlere atıf yaptığımı ve kaynaklar kısmında bu eserleri gösterdiğimi beyan ederim. 18/02/2021

Ahmed Yaseen Bishree AL-ANI
İmza

ÖZET

NESNELERİN İNTERNETİ CİHAZLARINDA DERİN ÖĞRENME KULLANILARAK EYLEM ALGILAMA

Ahmed Yaseen Bishree AL-ANI

Yüksek Lisans Tezi

Bitlis Eren Üniversitesi Lisansüstü Eğitim Enstitüsü
Elektrik Elektronik Mühendisliği Anabilim Dalı
Danışman: Prof. Dr. Sabir RÜSTEMLİ
Şubat 2021, 111 sayfa

Nesnelerin İnterneti (IoT) teknolojisi akıllı teknolojik cihazların birbiri ile iletişime geçip haberleşmesidir. Bununla birlikte Nesnelerin İnterneti (IoT) gelişimiyle beraber gün geçtikçe, akıllı uygulamaların ve birbirine bağlı olan cihazların sayısı artmaktadır. Derin Öğrenme (DL) yöntemi toplanan çok miktarda ham verinin işlenmesi, zekâ ve uygulama yeteneklerini daha da geliştirmek için gerekli hale gelmiş durumdadır. Araştırmacıların çoğunluğunun eylem algılama üzerine yoğunlaştığı görülmektedir. Deep learning kullanılarak IoT cihazlarında doğrudan eylem algılaması yaygın bir yöntem değildir. Derin Öğrenme uygulamaları yüksek CPU, RAM ve depolamaya ihtiyaç duyduğundan IoT cihazlarında standart Derin Öğrenme tekniklerinin kullanılması zordur.

Bu tez çalışmasında farklı olarak derin öğrenme tekniklerinin IoT cihazlarında kullanılması ile eylem algılama işlemi doğrudan kenar cihazında yapılmasın üzerine çalışılmıştır. Bunun için 3 farklı gerçek IoT cihazı üzerinde mini boyutlu Derin Öğrenme (DL-Lite) teknikleri uygulanmıştır. Bu tekniklerin IoT cihazlarında uygulanması sonucunda ortaya çıkan algılamada doğrululuk, gecikme ve cihazların sıcaklığı gibi parametrelere göre IoT cihazların ve mini Derin Öğrenme tekniklerinin kıyaslanması gerçekleştirilmiştir.

Anahtar Kelimeler: Eylem Algılama, IoT; Gömülü Cihazlar; Derin Öğrenme; Uç Hesaplama; Akıllı Şehir.

ABSTRACT

ACTION DETECTION IN IOT DEVICES USING DEEP LEARNING

Ahmed Yaseen Bishree ALANI

Master Thesis

Bitlis Eren University Graduate Education Institute
Department of Electrical and Electronic Engineering
Supervisor: Prof. Dr. Sabir RÜSTEMLİ
Feb 2021, 111 pages

Implementation of the Internet of Things (IoT) is becoming wide-spread, particularly in smart city applications. Due to the high amounts of raw data gathered by enormous of IoT devices, the Deep Learning (DL) method has become necessary to further develop intelligence and application capabilities. In particular visual action detection is one of the critical components of a smart city.

It is challenging to use standard Deep Learning techniques for action detection in IoT devices because Deep Learning applications need high CPU, RAM, and storage. To use the standard DL techniques in IoT devices some of DL models shrinked.

In this master's thesis, Deep Learning Lite and Micro techniques were applied on real IoT devices. Comparison of IoT devices and Deep Learning Lite and Micro techniques was performed in terms of parameters such as accuracy, delay, and temperature of the devices, applying these Techniques in IoT devices.

Keywords: IoT; Embedded devices; Deep Learning; smart city; Action Detection, Edge Computing.

TEŞEKKÜR

Her şeyden önce, başarılarım ve hayatım boyunca olacak tüm güzel şeyler için mahsusen ALLAH' a şükürler olsun.

Bu tez çalışması sırasında her türlü bilgi, teşvik ve deneyimleri ile yardımlarını esirgemeyen, üzerimde büyük emeği olan danışman hocam Elektrik Elektronik Mühendisliği Anabilim Dalı Başkanı Prof. Dr. Sabir RÜSTEMLİ ve diğer bölüm hocalarıma teşekkürlerimi sunarım.

Katkılarından dolayı her türlü maddi ve manevi desteklerini esirgemeyen en başta Öğr. Gör. Munip GEYLANİ' ye, Dr. Öğr Üyesi Baber BUTTİ ye, Bilgi İşlem Daire Başkanı İbrahim GÖK' e ve Raed AL-KATEEB 'e teşekkürlerimi borç bilirim.

Hayatımdaki ilk hocalarım olan Anneme ve Babama teşekkür ediyorum. Yine yanımda olan ve tez çalışması yaptığım süreç boyunca; tezimi bitirene kadar bana her zaman destek veren hayat arkadaşım ve eşim Asmaa' ya teşekkür ederim.

İÇİNDEKİLER

Sayfa

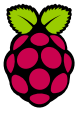
ÇİZELGELER DİZİNİ

ÇİZELGE **Sayfa**

ŞEKİLLER DİZİNİ

ŞEKİL **Sayfa**

SİMGELER DİZİNİ

	TensorFlow Lite
	TensorFlow
	Raspberry Pi
	Python
	C++
	Espressif
	OpenCV
	Arduino
	USB
	Kablosuz İşareti (Wi-Fi)
	SMS (Kısa Mesaj Hizmeti)
	Mail Service (e-posta hizmeti)
	MQTT Broker
	Nexmo SMS Gateway

KISALTMALAR DİZİNİ

AI	Artificial Intelligence (Yapay Zeka)
ANN	Artificial Neural Networks (Yapay Sinir Ağları)
ARM	Acorn RISC Machine
CNN	Convolutional Neural Networks (Evrişimli Sinir Ağları)
COCO	Common Objects In Context
CPU	Central Processing Unit (İşlemci)
CV	Computer Vision (Bilgisayar Görüşü)
DL	Deep Learning (Derin Öğrenme)
DNN	Deep Neural Networks (Derin Sinir Ağları)
DSP	Digital Signal Processors (Dijital Sinyal İşlemcileri)
FLOPS	Floating Point Operations Per Second (Saniyedeki Kayan Nokta İşlemleri)
FPS	Frames Per Second (Saniyedeki Kare Sayısı)
Gb	Gigabit (Gigabit)
GB	Gigabyte (Gigabayt)
GPU	Graphic Processing Unit (Grafik işleme Birimi)
HDD	Hard Disk Drive (Sabit Disk Sürücüsü)
HW	Hardware (Donanımsal)
IDE	Integrated Development Environment
IoT	Internet of Things (Nesnelerin İnternetini)
IPSO	IP for Smart Object (Akıllı Nesneler için IP)
IPv4	IP Version 4 (IP sürüm 4)
LSTM	Long Short-term Memory
M2M	Machine-to-Machine
mA	milliampere (miliamper)
mAP	Mean Average Precision (Ortalama Ortalama Hassasiyet)
MIT	Massachusetts Teknoloji Enstitüsü (Massachusetts Institute of Technology)
ML	Machine Learning (Makine öğrenme)
MQTT	Message Queuing Telemetry Transport
ms	Millisecond (Milisaniye)
NCS	Movidius NCS (Neural Compute Stick)
OS	Operating System (İşletim Sistemi)
RAM	Random-Access Memory

ReLU	Rectified Linear Units.
RNN	Recurrent Neural Network
RPi	Raspberry Pi
SD	Secure Digital (Hafıza Kartı)
SoM	System-on-Module
SMS	Short Message Service (Kısa Mesaj Hizmeti)
TF	TensorFlow
TFL	TensorFlow Lite
TFM	TensorFlow Mobile
TPU	Tensor Processing Unit (Evrensel Seri Veriyolu)
USB	Universal Serial Bus
VPU	Vision Processing Unit
YOLO	You Only Look Once

1. GİRİŞ

Bilgisayarla görme görevleri, insan görüşünü taklit edebilmek için bilgisayar sisteminin görsel dünyayı görüntülemesine ve tanımlamasına otomatik olarak izin verecek şekilde tasarlanmıştır.

Bilgisayarla görme ve derin öğrenme çalışmalarının ana temalarından biri, insanların birbirleriyle olan eylemlerini tanımlamaktır. İnsandan insana etkileşimde ve kişiler arası ilişkilerde, insan eylemlerini tanıma faktörü önemli bir rol oynar [1]. Eylemler; bireylerin kimliği, kişiliği ve psikolojik durumu hakkında bilgi içerdiği için bundan anlamlı veri çıkarmak zordur. Bu çalışma, gözetim, insan-bilgisayar etkileşimi gibi video izleme cihazları, insan eylemlerinin tespiti ve uyarı tetikleyiciler dâhil olmak üzere birçok kullanım için çoklu aktivite algılama durumlarını içermektedir [2] [3].

Bir Eylem Algılamanın farklı aktiviteleri, gözetim ve anormallik algılamadan, paylaşılan çalışma alanlarında insanlar ve robotik makineler arasında güvenli ve işbirliğine dayalı etkileşime kadar çok çeşitli uygulamalar için gereklidir [4]. Eylem Algılama, standart bir Bilgisayarla Görme problemidir ve üzerinde birçok araştırma yapıldığı görülmektedir. Eylem algılamada temel amaç, videoda gerçekleşen eylemleri belirlemek için videoyu analiz etmektir. Örneğin ayakta durma, koşma gibi bazı eylemler muhtemelen sadece tek bir çerçeve kullanılarak tanımlanabilirken; yürüme, koşma, eğilme ve düşme gibi daha karmaşık eylemleri doğru şekilde tanımlamak için tek çerçeveden daha fazlası gerekebilir [5]. Yerel geçici bilgi (local temporal information), bu tür eylemleri ayırt etmede önemli bir rol oynar. Daha fazlası için bazı kullanım durumlarında yerel geçici bilgiler yeterli değildir. İşlemi doğru bir şekilde tanımlamak veya videoyu sınıflandırmak için uzun süreli geçici bilgilere ihtiyaç duyulabilmektedir [6].

IoT cihazları derin öğrenme, yapay zeka algoritmalarının doğrudan veri toplayan bir cihaza uygulandığı bir metottur [7]. Bunun için dronlar, kameralar veya arttırılmış gerçeklik gözlükleri örnek verilebilir. Günümüz çözümlerinden farklı olarak IoT'de Derin Öğrenme, verileri bir sunucuya veya buluta aktarma ihtiyacını ortadan kaldırmaktadır [8].

IoT Cihazlarında eylem algılama tekniklerinin uygulanması birçok fayda sağlamaktadır. Örneğin IoT cihazları CPU, GPU, RAM ve Güç gibi donanım kaynaklarını daha az tüketmektedir. Bununla beraber birbirine bağlı çok sayıda fiziksel cihaz, birçok farklı bulut hizmetinden faydalanma imkânı, platformlar arası destek ortamı ve çeşitli iletişim protokolleri ile uzaktan izlenebilir ve kontrol edilebilir bir Cihaz Kontrol Sistemi desteği bu faydalar arasında gösterilebilir. IoT cihazlarda eylem algılama tekniklerinde insansız araçlar, akıllı ev sistemleri, akıllı şehir sistemleri, spor, sağlık ve trafik gibi birçok uygulama alanı bulunmaktadır [3].

Veri işleme hızı çok önemlidir ve milisaniyelik veri aktarım gecikmesi telafi zor durumlar ortaya koyabilmektedir [8] [13]. Ağ erişiminde, ağa istikrarlı bir şekilde erişebilmek ayrı bir sorundur. Böyle bir durum, örneğin, daha az nüfuslu alanlarda veya kasıtlı olarak sinyali engelleyen odalarda meydana gelebilir. Genellikle cihazı terk etmemesi gereken hassas verilerle çalışılmaktadır. Düşük güç tüketimi istenen ayrı bir özelliktir. IoT'de Derin Öğrenme tekniğinde, verilerin düzenli olarak buluta gönderilmesi gerekmediğinden, pil kullanımı otomatik olarak indirilmiş olur [11]. IoT uygulamalarında Derin Öğrenme için otonom araçları ve insansız hava araçlarını da kapsayan çok sayıda senaryo vardır [15] [16].

IoT Teknolojileri, büyük verilerle doğrudan başa çıkamayan farklı cihazlardan gelen ham verinin belirli bir kısmıyla ilgilenen ve cihaz kapasitesinin sınırlı olduğu teknolojilerdir. Davranışı tahmin etme ve sonrasında bu tahminlere dayanarak kendi kendine karar verme tekniklerinden bulmayı zordur. Örneğin, sadece IoT cihazında bulunan donanım ile gerçek zamanlı eylem algılama işlemi yapılması pek mümkün olmamaktadır. IoT cihazları IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge 2.3'da belirtildiği gibi sınırlı donanım kaynaklarına sahiptir. IoT cihazında kullanılabilen sınırlı sayıda Derin Öğrenme teknikleri mevcuttur [10]. Ayrıca yine IoT cihazlarıyla çalışabilen ve farklı video kamera türlerinde video akışını yakalayan az sayıda kamera türü vardır [6].

Gerçek zamanlı durumlarda video işlemenin genellikle video akışı kaynağından gelen çeşitli ve büyük bilgilerle başa çıkmak için uygulandığı unutulmamalıdır. Full HD, UHD gibi gelişmiş video kodekslerinin çok yüksek çerçeve hızları üretmesi sorun oluşturmaktadır. Esasen, videonun kalitesi ne kadar yüksekse, verileri işlemek o kadar uzun sürer [4] [5]. Buna göre, yüksek kaliteli videolar, Eylem algılama gibi gerçek zamanlı veya gerçek zamanlıya yakın veri işleminin gerekli olduğu projeler için büyük bir zorluk haline gelirken, bununla birlikte Derin Öğrenme tekniklerinden gelen bilgileri, kullanılan veri setleriyle karşılaştırması gerekmektedir [6] [7].

IoT cihazında, Makine Öğreniminin bile sınırları bulunmaktadır. Sınıflandırıcı aşamasını hazırlamak için Nesne Özelliği Çıkarma işlemindeki insan eyleminin yeni özellikler eklemesi veya diğerlerini sınıflandırması gerekecektir. Derin Öğrenme teknikleri IoT cihazları ise milyonlarca açıklamalı resimle başa çıkabilir. Bu ham verileri otomatik bir şekilde RNN veya CNN kullanarak bu bilgileri cihazın kendisinde yerel olarak depolayacak şekilde eğitecek; daha sonra derin öğrenme teknikleri bu bilgilerle doğrudan ilgilenebilecektir. Diğer bir deyişle, ortak temanın gerçek zamanlı karar verme ihtiyacı olduğu göz önünde bulundurulursa daha fazla işlem için buluta veri göndermek zaman kaybına neden olmaktadır [26].

IoT'de Derin Öğrenme, giyilebilir cihazlarda da kullanılmaktadır. Akıllı saatler ve bileklikler, yaşam parametrelerinin daha doğru ölçümlerini toplayabilmektedir. Yapay zekâ algoritmalarıyla birleştirilen bu bilgi, eğitimi planlamayı, sağlığı iyileştirmeyi ve yaşlılara bakmayı mümkün kılmaktadır [27].

Ayrıca IoT üzerinde Derin Öğrenme; titreşimleri, sıcaklığı ve Ultrasonları inceleyen sensör verilerine dayalı olarak gelecekteki makine arızalarının hızlı bir şekilde tahmin edilmesini sağlayan Endüstri 4.0 varsayımlarının yerine getirilmesine yardımcı olur. Aynı zamanda üretim süreçlerinin ve kalite kontrolün artan otomasyonunu destekler [28]. Örneğin, görüntü işleme algoritmaları ekipmana yakın nesneleri tanımlamak, sınıflandırmak ve izlemek için kullanılabilir [29].

IoT'de Derin Öğrenmeyi kullanmak için yukarıda belirtilen nedenlerden başka hayatımıza yeni giren akıllı evler, akıllı telefonlar, arttırılmış gerçeklik ve birçok teknolojinin sınırsız kullanım alanlarıyla beraber birçok uygulamasının yapılacağı aşikârdır [30] [31].

Gelişen teknolojiyle beraber Derin öğrenme algoritmaları gün geçtikçe küçülmektedir. Günümüzde Google Asistan ekibi, mikro-denetleyicilerde çalışacak kadar küçük boyutta kelimeleri algılayabilen ve yalnızca 14 kilobaytlık bir modele sahiptir [11]. Derin öğrenme test yöntemleri kullanılarak IoT üzerinde eylem algılama sistemlerindeki bazı sınırlamalar belirlenecek ve bir tasarım test çözümü uygulanacaktır. Dahası, uygulanan tasarım testi derin öğrenme çerçevesi, geliştiricilerin birim testleri yazmasına ve çalıştırmasına olanak tanıyacaktır [35].

Endüstride ve araştırma sektörlerinde IoT cihazlarında Eylem Algılama giderek daha gerekli hale gelmektedir. IoT cihazlarında Eylem Algılamanın geleceğini daha öngörülebilir ve yönetilebilir kılmak için bu alanda daha fazla araştırma yapılması gerekmektedir. Bu çalışma, IoT ve Gömülü cihazlarda Derin Öğrenmeyi kullanarak Eylem Algılama tekniklerinin incelenmesine katkıda bulunacağı düşünülmektedir [12]. Etkili, verimli ve güvenilir derin öğrenme modelleriyle desteklenen IoT uygulamalarında, eylem algılama oluşturma hakkında daha iyi bir imaja sahip olmak için hem endüstriyel etki alanı hem de araştırmacının çalışmaları için mevcut veya gelecekteki tasarımları iyileştirmek için kullanılabileceği öngörülmektedir [13] [14].

IoT yapısı altında sensörler sürekli olarak veri toplamaktadır. Bir video kameranın verilerini ağ üzerinden bir merkeze veya buluta taşımak yerine, bu sabit veri akışını sensörlerin kendisinde işlemek daha etkilidir. Bazı geliştiriciler bu gömülü aygıtları "uç" olarak adlandırmaktadır. Düşük güç tüketen, kaynakları kısıtlı olan uç cihazlarda, modeller ve platformlar daha verimli hale gelmekte ve daha hassas olarak yerleşik veri işlemeyi mümkün kılmaktadır [15].

Kameralar ve diğer cihazlar gibi akıllı sensörler, nesne algılama ve nesne izleme ve sınıflandırma gibi yakından ilgili etkinlikleri gerçekleştirmektedir. Sistem üzerinde bu ön işleme çalışmasının yapılması, istenmeyen trafiği ağdan uzak tutarak performansı arttırmaktadır. Bu IoT cihazının kendisi tarafından yapılabilecek çok önemli görevlerden biridir [16].

Bu tez çalışmasında, Derin Öğrenme tekniklerini kullanarak IoT ve gömülü cihazlarda Eylem Algılama incelenmesi yapılacak olup, aynı zamanda, Derin Öğrenme uygulama teknikleri için önerilen Optimizasyon çözümlerini geliştirerek IoT cihazlarında Eylem Algılamanın sonuç doğruluğu incelenecektir. Başka bir ifade ile bu tez çalışmasında temel olarak IoT cihazlarında Eylem Algılama geliştirebilmek için Derin Öğrenme modellerini araştırmak ve bu geliştirilen Eylem Algılama tekniği ile nesnelerin insansız gerçek zamanlı izlenmesi hedeflenmektedir.

1.1. Önceki Çalışmalar

Bu bölüm, yapılan literatür çalışma için gerekli olan temel kavramlara ve ilgili yayınlanmış çalışmalara daha yakından bir bakış içermektedir.

• **Yukui Luo** [17], en gelişmiş Derin Öğrenme çerçevelerinden biri olan Deep Convolutional Neural Network'ün OpenCL tabanlı bir uygulaması sunmuştur. Çerçeveler üç önemli katkı hedeflemiştir. Bunlar; gerçek zamanlı bir nesne tanıma sistemi, taşınabilir cihazlarda bile uygulanabilen düşük güç tüketimine sahip bir çerçeve ve çeşitli bilgi işlem cihazlarında çalışabilen bir çerçevedir. Çerçeve hızı, YOLO V2 kıyaslamasına dayalı olarak CUDA çerçevesiyle karşılaştırılarak değerlendirilmiştir.

• **Alpaydin** [18], değişken, gürültülü arka planlara sahip, kontrastı düşük olan, uzun menzilli görüntüler için oldukça verimli nesne tanıma sağlamak için Derin Evrişimli Sinir Ağları ile birlikte çalışan uyarlanabilir bulanık tabanlı bir ağ topolojisi önermiştir.

• **Daniel** [19], LİDAR verilerini ve RGBD nokta bulutlarını kullanarak doğru ve verimli nesne algılama elde etmek için bir 3D Evrişimli Sinir Ağı (CNN) mimarisi "VoxNet" i sunmuştur. Kamuya açık en son teknolojiye sahip kıyaslama yaklaşımları değerlendirilmiş ve yaklaşımlarının nesnelerini gerçek zamanlı olarak sınıflandırırken bu kriterlerin ötesinde doğru bir şekilde uygulandığını görmüşlerdir.

• **Lewis** [20] makalesinde, önceden işleme veya başka türlü çok maliyetli olan derinlemesine değerlendirmeler olmaksızın derin nesne tanımayı şekillendiren SimpleNet adlı bir DIY ağı önermiştir. Doğruluk, son teknolojiye göre nispeten daha az olsa da, SimpleNet, sonsuz sayıda parametre ile uygun kayıp fonksiyonlarından güç çekmeye çalışır. Buna karşılık diğer ağlar, gücü katmanların derinliklerinden alır. Au-thor, izleyicilere performans açısından tüm bu CNN modelleri hakkında derin bir fikir vermek için OverFeat, VGG16, Fast R-CNN ve YOLO gibi çeşitli CNN modellerini SimpleNet ile karşılaştırmıştır [42].

• **Girshick** [21], Hızlı R-CNN adlı Bölge tabanlı Evrişimli Sinir Ağı'nı sunmuştur. Bu ağ, düşük hesaplama hızıyla alım satım yaparken nesneleri yüksek doğrulukta tespit edebilir niteliktedir. Bu nedenle, ağın gerçek zamanlı nesne tespiti için uygun olmadığı düşünülür ve bunun aracılığıyla tanıma kabul edilebilir bir performans yanlışlığı sergiler.

• **Ren** [22], makalelerinde Hızlı R-CNN'nin Faster R-CNN olarak bilinen güncellenmiş bir sürümünü sunmuştur. Adından da anlaşılacağı gibi, Bölge tabanlı Evrişimli Sinir Ağının güncellenmiş sürümü, önceki sürümünden ve diğer birçok son teknolojili ağdan daha iyi hesaplama hızı ve doğruluğu göstermiştir. Bir Bölge Teklif Ağı (RPN) eklenerek, ağın hesaplama hızı, özellikler üreterek ve bunları son algılamayı yapmaktan sorumlu Tespit Ağı ile paylaşarak

artırmıştır. Daha hızlı R-CNN modelleri, gerçek zamanlı algılamalar yapabilir fakat boyut olarak daha küçük nesneleri algılamakta zorlanır.

• **Kim** [23], en son teknik yenilikleri kullanarak özellik çıkarma aşamasında değişiklikler yapmış ve PLANET olarak bilinen daha yeni bir ağ çalışması sunmuştur. Bu ağ, hesaplama maliyetini azaltırken, birden çok kategorideki nesneleri benzerleriyle birlikte aynı doğrulukla algılayabilir niteliktedir.

• **Dai** [24], nesne algılama söz konusu olduğunda son teknoloji olan mevcut ResNet'i benimserken, R-FCN adında tamamen evrişimli bir ağ kurdu. Nesne algılama doğruluğunu arttırma girişiminde, Hızlı R-CNN'deki tam olarak bağlı katmanların yerini, konuma duyarlı ve uzamsal bilgileri kodlayabilen bir dizi skor haritası almıştır. Sonuç olarak, R-FCN, Daha Hızlı R-CNN ile benzer doğrulukta olup, R-FCN'nin daha iyi hesaplama hızlarında olduğunu göstermiştir.

• **Kong** [25], birden çok çıktı katmanında algılama gerçekleştirerek nesneleri birden çok ölçekte algılayabilen HyperNet adlı bir ağ sunmuştur. Bu ağ, tarafından önerilen ve çeşitli ölçeklerdeki nesneleri tespit etmek için etkili bir çerçeve sağlayan MS-CNN'ye benzer.

• **Liu** [26], yüksek doğrulukta gerçek zamanlı performans sağlayabilen Single Shot multi-box Detector (SSD) adlı basit ve anlaşılır bir ağ sunarak, bu ağın bölgesel teklif yöntemini kullanmamaktadır. Bu ağda, nesne lokalizasyonu ve sınıflandırması, sınırlayıcı kutu regresyonunu gerçekleştirmek için çoklu-kutulu bir teknik kullanılırken, ağın tek bir ileri geçişinde gerçekleştirilir. Dolayısıyla SSD, uçtan uca hesaplamalar gerçekleştirebilir özelliktedir.

• **Redmon** [27], bildirisinde devrim niteliğindeki ağları YOLO'nun güncellenmiş bir versiyonu olan YOLOv3'ü sunmuştur. Bu model, Faster R-CNN, VGG-16, ResNet gibi diğer tüm son teknoloji ağları geride bırakmıştır. Hesaplama hızı ve doğruluğu açısından, diğer ağların yapamadığı yüksek hassasiyeti korurken gerçek zamanlı tespitler ve izleme yapmak için ideal bir ağ haline getirilmiştir. YOLOv3, üç farklı ölçekteki nesneleri etkili bir şekilde algılayabildiği için küçük boyutlu nesneleri algılayabilir.

• **Mehdi** [12], IoT veri madenciliği yaklaşımlarına odaklanan derin öğrenme yöntemi hakkında bir anket sunmuştur. IoT altyapısı ve hizmetleri için farklı sınıflandırma, kümeleme ve sık model madenciliği algoritmalarını ele aldı fakat bu çalışmada, çalışmanın odak noktası olan IoT üzerindeki DL yaklaşımlarıyla Eylem Algılama dikkate alınmamıştır.

• **He Li** [28], makalesinde, uç hesaplamada IoT için çeşitli derin öğrenme modelleri için esnek bir model formüle etmek için yol sunmuştur. Ayrıca, uç bilgi işlem modelinin hizmet kapasitesini optimize etmek için verimli bir çevrimiçi algoritma tasarlamıştır. Ancak eylem algılamayı veya gömülü cihazı dikkate almamışlardır.

• **Hong** [29], Edge computing’2 olarak sistemin çekirdeğini kullanmıştır. Öncelikle aktif kameralardan görüntüleri alır ve ardından bu görüntüleri izleyiciden geçirir. EdgeServer, izleme sürecinden sonra yalnızca sıkıştırılmış bilgileri Cloud IoT Core'a gönderir ve DL bölümünü bulut hizmetine uygulamak için yerel web ara yüzü verilerini sağlamıştır.

• **Shreshth** [30], hızlı hareket eden araçları tespit etmek için akıllı bir gözetim mimarisini modellemiştir. Elde edilen mimari modellemede video, akıllı telefonlar, akıllı tabletler, arabalardaki bilgisayarlar ve hesaplama özelliklerine sahip diğer yerinde cihazlar gibi sis hesaplama düğümlerinde işlenir. Ancak gözetim mimari modelleri, araçları yalnızca gerçek zamanlı olarak algılayabilir niteliktedir. Fakat birden fazla hedef nesneyi algılayamaz.

• **Faisal** [31], Gerçek Zamanlı IoT izleme sistemi için web tabanlı geliştirilmiş bir uygulama önermiştir. IoT'de, Raspberry Pi gibi cihazlar, sensörleri izlemek ve aktüatörleri uzaktan kontrol etmek için kullanılabilir niteliktedir. Bulut hizmetleriyle MQTT bağlantısını kullanmıştır. Gömülü cihazları düşünmeden veya bir Lite derin öğrenme kitaplığı kullanmadan IoT'yi akıllı evler için bir ev izleme sistemi olarak kullanmakla kendilerini sınırladılar.

• **Srinivasan** [32], gömülü cihaz veya farklı IoT İşletim Sistemleri dikkate alınmadan, IoT cihazına yerel olarak uygulanacak TensorFlow Lite derin öğrenme kitaplığını kullanan bir nesne algılama sistemi önermiştir. Ayrıca IoT bağlantı protokolünün MQTT olarak kullanımı yoktur.

• **Jure** [33], bir web uygulaması geliştirmeye benzer; Bulut hizmetlerine IoT bağlantısı için MQTT aracısını kullanmıştır.

• **Ruimin** çalışmasında [34], sistem kamera düğümleri, IoT cihazları, hücresel veri iletim modülleri ve merkezi bir sunucudan oluşan IoT cihazları Raspberry Pi 3B'dir. Genel tasarım, hesaplama yükü ile veri aktarım hacmi arasındaki dengeyi ve derin öğrenmeyi kullanarak sistemin güvenilirliğini ve ölçeklenebilirliğini dikkate alır.

1.1.1. Tezin Literatüre Katkısı

Yapılan araştırma dâhilinde, literatürde IoT cihazları ile kullanabilen Derin Öğrenme-Lite modellerinin arasındaki bağlantının incelmediğini gördük eylem algılama ile IoT'deki Derin Öğrenme-Lite yöntemlerinin uygulamaları arasındaki belirli ilişkiyi araştırmaya yönelik bir makale bulunmamaktadır. IoT cihazlarına gelince, bazı araştırmacılar deneysel kısım için IoT ve gömülü cihazlar olarak Raspberry Pi, ve ESP32 kullandı [35]. Çok az çalışma, IoT ortamlarında kullanılan ortak veri madenciliği, makine öğrenimi ve Derin Öğrenme yöntemlerini sunar. TensorFlow Lite'ın gerçek zamanlı nesne algılama ve tanıma için en uygun Derin Öğrenme algoritması olduğu söylenebilir. Toplanan bilgilerden, nesne algılama sistemlerinin araştırma ve

geliştirmelerinin çoğunun ya IoT olmayan cihazlarda ya da düşük maliyetli gözetim sistemlerinde, park sistemlerinde ve araba gibi araçlarda uygulandığı çok açıktır [9]. Aynı zamanda, nesnelerin tespiti ve izlenmesi için en iyi Derin Öğrenme modelini belirleyerek IoT ve gömülü cihazlarda Derin Öğrenme modellerini kullanarak mevcut son teknoloji Eylem Algılama sistemini değerlendirmek için sadece küçük bir araştırma yapılmış olup gerçek durum ortamlarındaki eylemler üzerine bugüne kadar bu çalışma alanında çok az araştırma yapılmıştır. Bu nedenle Derin Öğrenme API'si, eylem algılama süreçlerini hızla geliştiren ve dağıtan güçlü bir araç haline gelmiştir. Bu nedenle, bu araştırma çalışmasında verilerin IoT, Gömülü ve Mikrodenetleyici cihazlarında eğitilmesi amaçlanmıştır. Bu tez, bu algoritmaların performansını değerlendirmek ve gelecekteki yeniliklere doğru bir adım olarak IoT üzerinde gerçek zamanlı Eylem Algılamayı tanımak için DL-Lite modelleri olmuştur [36].

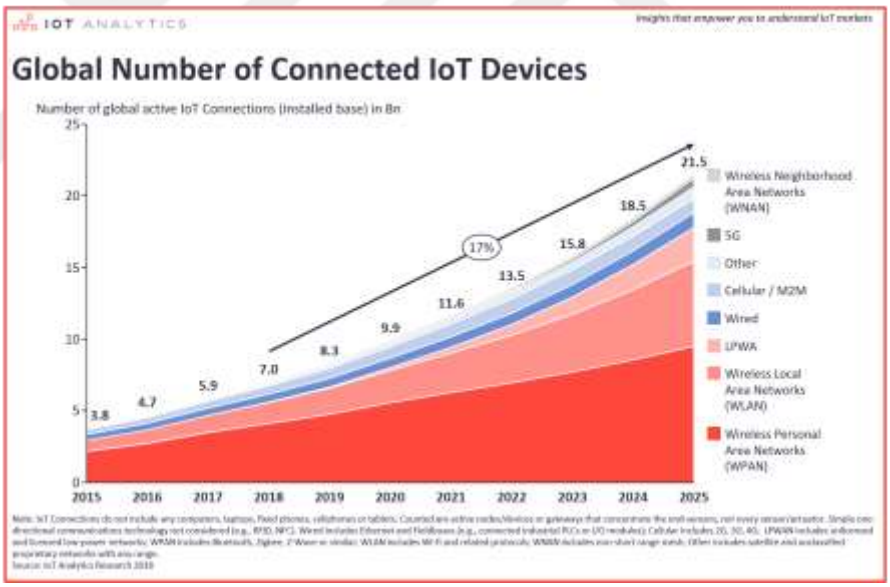
Literatür taraması, IoT üzerinde Eylem tespiti ile kullanılan Derin Öğrenme modellerinin sınırlaması ve bunların etkinliği hakkında derinlemesine bir anlayışla bilgi edinmek için seçildiğinden, tanımlanan modellerden en uygun ve verimli yöntem seçilebilir. Bu tezde, Derin Öğrenme Lite ve normal modelleri seçtik. IoT cihazları için tam desteğe sahip, eylem algılama modeli için kullanılan son teknoloji Derin Öğrenmeye göre seçilmiştir [37][38].

1.2. IoT Cihazları

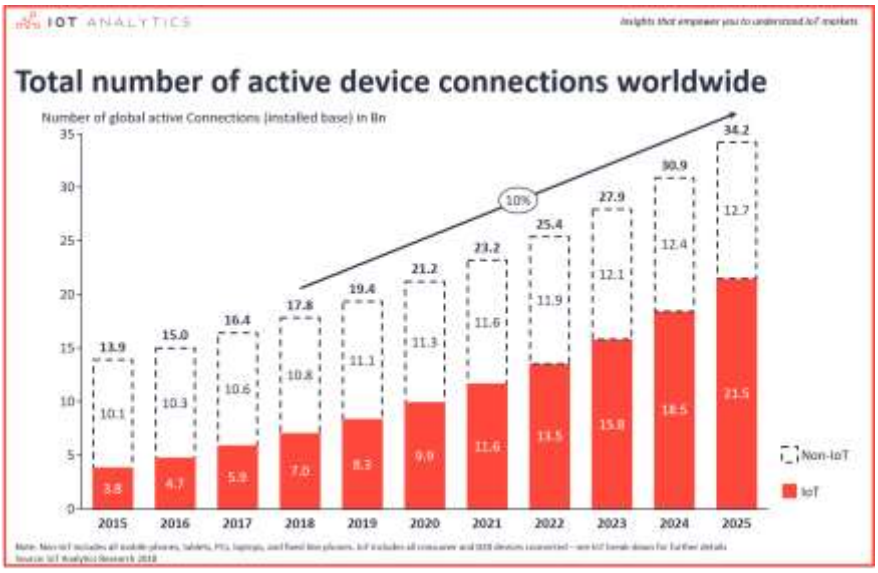
Nesnelerin İnterneti (IoT), internet üzerinden diğer cihazlara ve sistemlere bağlanmak ve veri alışverişi yapmak için sensörler, yazılımlar ve diğer teknolojilerle gömülü fiziksel nesnelerin ağını tanımlar. Bu cihazlar, sıradan ev eşyalarından sofistike endüstriyel aletlere kadar çeşitlilik gösterebilmektedir [39].

Şekil 1.1 ve Şekil 1.2'de gösterildiği gibi IoT ANALYTICS'in paylaştığı verilere göre 2025 yılına kadar İnternet'e bağlı 21,5 milyar "IoT" cihazı olması öngörülmektedir. Bu IoT'lar akıllı telefonlar ve bilgisayarlar gibi genel amaçlı cihazlar değil, Otomatik satış makineleri, jet motorları, kendi kendine giden arabalar, gözetim sistemi kameraları gibi işlev nesneleri ve şekil 2'de gösterildiği gibi çeşitli cihazlardan oluşmaktadır [9][40].

IoT, birçok sektörü dijital işletmeye dönüştürerek ve yeni iş modellerini teşvik ederek ekonomi üzerinde üretkenliği artırmak, çalışanların ve müşterilerin etkileşimini artırmak gibi büyük bir etkiye sahip olacaktır.

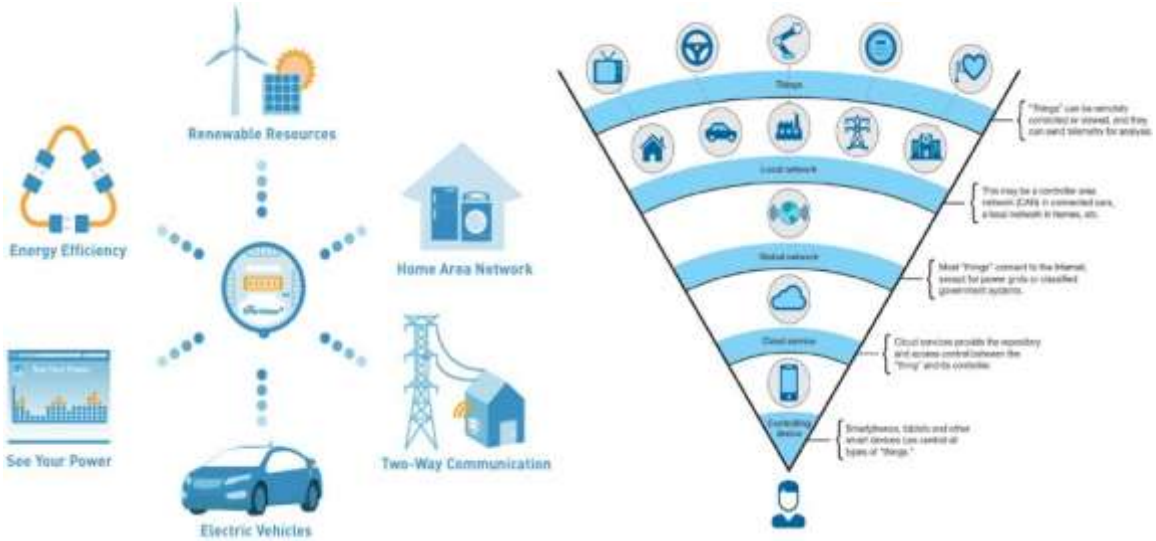


Şekil 1.1 Bağlı IoT Cihazlarının Küresel Sayısı [41].



Şekil 1.2 Dünya çapındaki toplam aktif cihaz bağlantısı sayısı [41].

Nesnelerin İnterneti, her ay milyonlarca yeni sensör ve cihazın çevrim içi hale gelmesiyle hızla büyümektedir. Nesnelerin İnternetinin, ucuz, düşük güçlü bileşenlerle güçlendirilmiş yaygın internet bağlantısı hem kurumsal hem de tüketici tarafından büyük ilgi görmektedir. Akıllı ekmek kızartma makinelerinden akıllı şehirlere, tedarik zincirlerindeki RFID etiketlerinden tıbbi izleme implantlarına, öğrenen termostatlardan sürücüsüz arabalara kadar birçok yeni cihaz bu teknolojiyi kullanmaktadır [38]. Kullanım alanlarının çeşitliliğiyle beraber, dünyadaki insan nüfusundan daha fazla İnternet'e bağlı cihaza sahip olmaya nasıl geçildiği sorusu akıllara gelmektedir. Şekil 1.3'de tarihsel gelişimiyle beraber IoT'un nereden geldiği ve gelecekte nereye gideceğine dair genel bir fikir verilmektedir [2].



Şekil 1.3 IoT İçin Bazı Modeller [42].

1.2.1. IoT'de Seçilen Önemli Olayların Zaman Çizelgesi

1969 yılında modern internetin öncüsü olan ARPANET, ABD Savunma İleri Araştırma Projeleri Ajansı DARPA tarafından geliştirilmiş ve hizmete sunulmuştur. Bu, Nesnelerin İnternetinin "İnternet" kısmının temelidir.

1980'li yıllarda ARPANET, ticari sağlayıcılar tarafından halka açılmış ve insanlar dilediği sürece bir şeyler arasında bağlantı kurmalarına olanak sağlamıştır.

1982 yılında Carnegie Mellon Üniversitesi'ndeki programcılar, bir Coca-Cola otomat makinesini İnternet'e bağlayarak, satın almadan önce makinede soğuk gazlı içecekler olup olmadığını kontrol etmelerine olanak tanımışlardır. Bu uygulama genellikle ilk IoT cihazlarından biri olarak anılmaktadır.

1990 yılında John Romkey, bir meydan okumaya yanıt olarak ekmek kızartma makinesini İnternet'e bağlamış ve başarılı bir şekilde açıp kapatarak bizi modern IoT cihazları olarak düşündüğümüz şeye daha da yaklaştırmıştır.

1993 yılında Cambridge Üniversitesi'ndeki mühendisler, İnternet'i aletler ve yiyeceklerle birleştirme geleneğini sürdürerek, bir kahve makinesinin durumunu dakikada üç kez çeken ve durumunun çalışanlar tarafından uzaktan izlenmesine olanak tanıyan dünyanın ilk web kamerası olarak tanımlayabileceğimiz bir sistem geliştirmişlerdir.

1995 yılında ABD hükümeti tarafından yürütülen GPS uydu programının ilk sürümü tamamlanmış ve birçok IoT cihazı için en hayati bileşenlerden biri olan konum özelliğinin sağlanmasına öncü olmuştur.

1998 yılında IPv4 standardı yerine IPv6 standardı taslak bir standart haline gelmiştir. Böylece 32 bit IPv4 yalnızca 4,3 milyar civarında cihaz için IP dağıtabiliyor iken 128 bit IPv6, 340 undesilyona (36 sıfırla 340) kadar benzersiz IP adresi dağıtmaya imkân sağlamıştır.

MIT'in Auto-ID Labs başkanı Kevin Ashton 1999 yılında, RFID izleme teknolojisinin potansiyelini göstermek için Proctor & Gamble yöneticilerine yaptığı bir sunumda IoT ifadesini kullanmıştır. Bu ifadenin ilk kez burada kullanıldığı düşünülmektedir.

2000 yılında LG İnternet Buzdolabı adını verdiği ekran ve algılayıcılarla dolapta neler olduğunu takip edebilen bir ürün piyasaya sürmüştür. Ancak ürün aşırı pahalı olması nedeniyle fazla ilgi görmemiştir.

"Nesnelerin İnterneti" ifadesi kitap başlıklarında ortaya çıkmaya başlar ve görünümleri yaratılır.

2007 yılında ilk iPhone geliştirilmesiyle halkın dünya ve İnternet bağlantılı cihazlarla etkileşime girmesi için yepyeni bir yol sunulmuştur.

2008 yılında ilk uluslararası IoT konferansı İsviçre'nin Zürih şehrinde yapılmıştır. Aynı zamanda İnternet'e bağlı cihazların sayısı dünyadaki insan sayısını geçecek şekilde arttırılmıştır.

2009 yılında Google, sürücüsüz araba testlerine başlamıştır ve St. Jude Tıp Merkezi İnternet bağlantılı kalp pillerini piyasaya sürmüştür. Ayrıca, IoT'nin büyük bir parçası olması muhtemel olan blok zinciri teknolojilerinin öncüsü olan Bitcoin bu yıl çalışmaya başlamıştır [43].

2010 yılında Çin hükümeti IoT'yi kilit bir teknoloji olarak adlandırıyor ve uzun vadeli kalkınma planlarının bir parçası olduğunu duyuruyor. Aynı yıl Nest, alışkanlıklarınızı öğrenen ve evinizin sıcaklığını otomatik olarak ayarlayarak "akıllı ev" konseptini ön plana çıkaran akıllı bir termostatı piyasaya sürmüştür.

2011 yılında pazar araştırma şirketi Gartner, bir teknolojinin popülerliğini gerçek kullanışlılığına karşı ölçmek için kullanılan bir grafik olan "heyecan döngüsüne" IoT'yi eklemiştir.

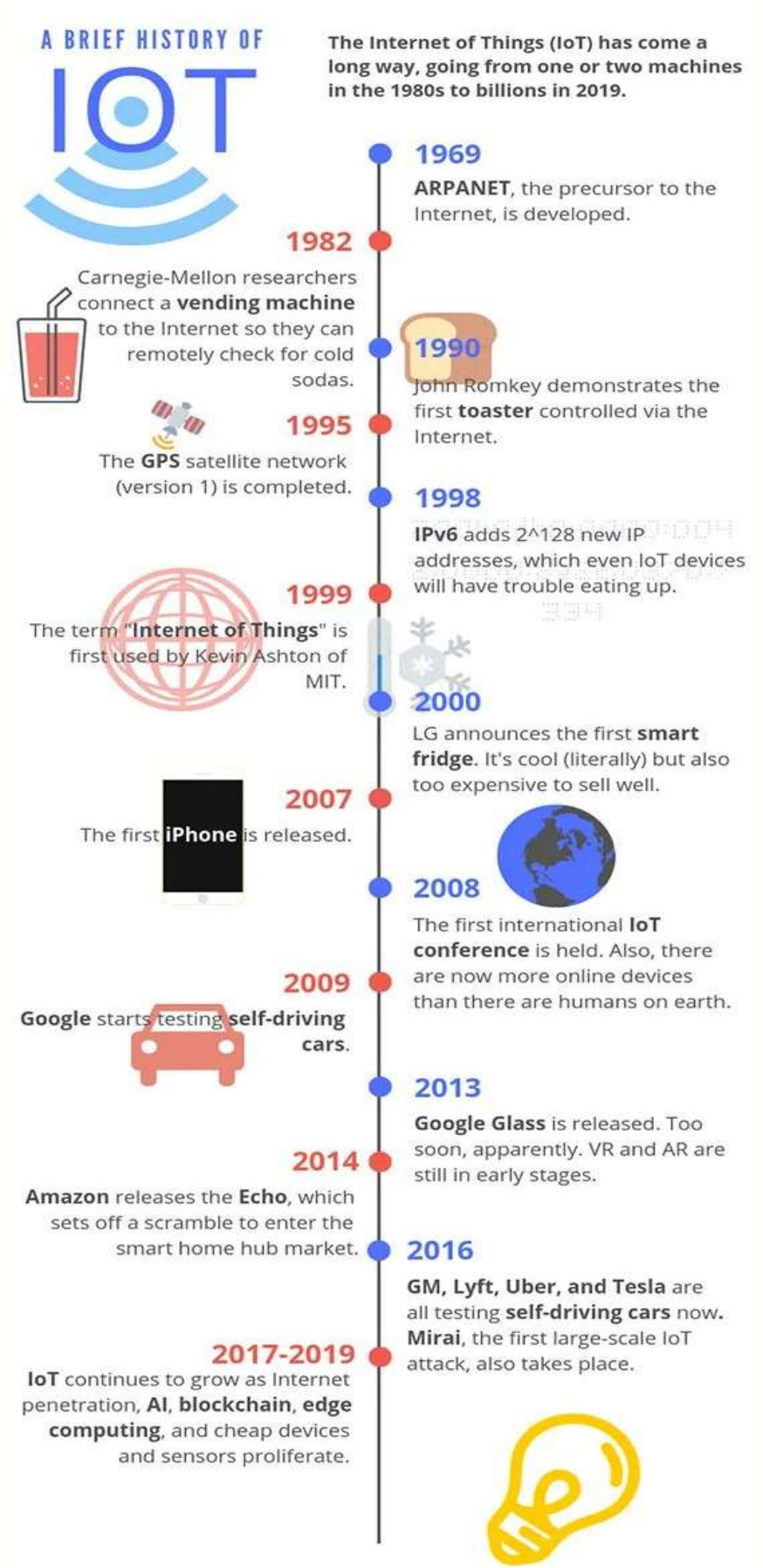
2013 yılında IoT ve giyilebilir teknolojide devrim niteliğinde ve muhtemelen zamanın ötesinde bir adım olan Google Glass'ı piyasaya sürmüştür.

2014 yılında Amazon, Echo'yu piyasaya sürerek IoT cihazlarının akıllı ev merkezi pazarına girmesinin yolunu açmıştır.

2016 yılında General Motors, Lyft, Tesla ve Uber, sürücüsüz arabaları test edilmiştir. Mirai botnet'in, üreticisinin varsayılan oturum açma bilgileriyle IoT cihazlarına saldırması, onları devralması ve DDoS popüler web sitelerinde kullanmasıyla ilk büyük IoT kötü amaçlı yazılım saldırısı da bu yıl gerçekleşmiştir.

2017- 2019 yılları arasında IoT cihazları daha ucuz, daha kolay ve daha geniş kabul görerek sektörün her yerinde küçük inovasyon dalgalarına yol açmıştır. Otonom araçlar gelişmeye devam etmiş, BitCoin ve yapay zeka (AI), IoT platformlarına entegre edilmeye başlanmış; artan akıllı telefon ve geniş bant penetrasyonu IoT'yi gelecek için önemli bir teknoloji haline getirmiştir [28].

IoT zaman akışı Şekil 1.4'de gösterilmiştir [43].

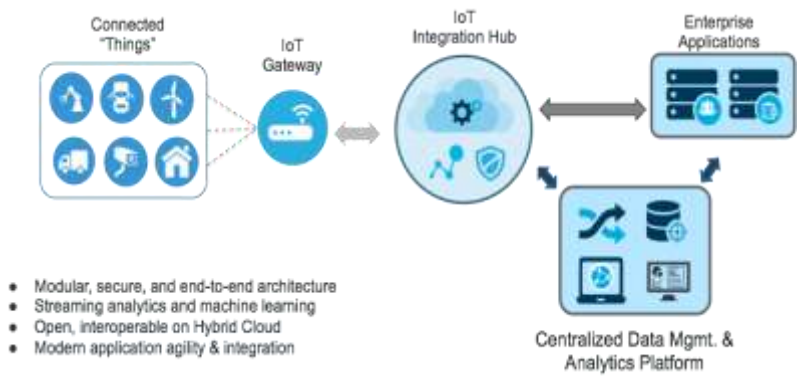


Şekil 1.4 IoT Zaman Akşı [43].

1.2.2. IoT'nin Geleceği

Gartner Hype Döngüsünde, önümüzdeki birkaç yıl içerisinde beklentiler yeniden ayarlanmaya çalışılacaktır. Bu çalışma uzun vadeli olsa da IoT muhtemelen yeni bir norm olacaktır. Evde olunca bile akıllı telefondan durumu kontrol etmek oldukça kolay olacaktır. Tedarik zincirindeki her ögenin gerçek zamanlı bir resmini elde etmek de oldukça kolaydır. AI ve blockchain gibi teknolojiler, cihazları daha bağımsız ve daha iyi bir ağa sahip hale getirmek için giderek daha fazla kullanılmaya başlanmıştır. "Uç bilişim" teriminin yükselişi, büyük ölçüde, IoT cihazlarının yaygınlaşmasının buluta uzun gidiş-dönüş yolculuğu ve yerel kullanıcılar için geri dönüşü elverişsiz hale getirdiği gerçeğinden kaynaklanmaktadır. Kitlesel donanımın benimsenmesini gerektiren şeyler biraz zaman alabilir. Dolayısıyla Nesnelerin İnternetine geçiş kademeli olacaktır. Kademeli olmasının nedeni ise çoğu faaliyetin çevrimiçi olarak kullanılması ve çalıştırılması, ortaya çıkacak olan gizlilik ve güvenlik sorunlarını anlamak açısından araştırmacılara biraz daha zaman kazandıracaktır. [58]

IoT (uç bilişim, edge computing), dağıtılmış bir bilgi işlem paradigmasıdır. Merkezi bir bilgi işlem paradigmasının aksine, tüm verileri merkezi sunucuya toplamaz ve bunun yerine bunları işleme koymaz; verileri ve bilgi işlem iş yükünü dağıtılmış cihaz düğümlerine veya yerel sunuculara dağıtır ve bunlara “kenar sunucuları” denir. IoT, bilgisayar veri depolamasını ve işlemeyi gerekli işlemin yapılacağı konuma yakınlaştırır. IoTde, hesaplama büyük ölçüde uç sunucularda gerçekleştirilmektedir [13]. IoT, uygulamaları, verileri ve bilgi işlem hizmetlerini merkezi noktalardan kullanıcıya daha yakın konumlara kaydırmaktadır. Sonuç olarak, IoT ağları daha az gecikme ve daha fazla güvenlik sunabilmektedir. Ayrıca, merkezileştirilmiş veri merkezi ile iletişim azalmaktadır. Bulut bilişimin aksine IoT, ağın ucunda merkezi olmayan veri işlemeyi ifade etmektedir. IoT Şekil 1.5'te gösterildiği gibi geleneksel IoT-bulut bilişim modelinde bir ara katman sunmaktadır.

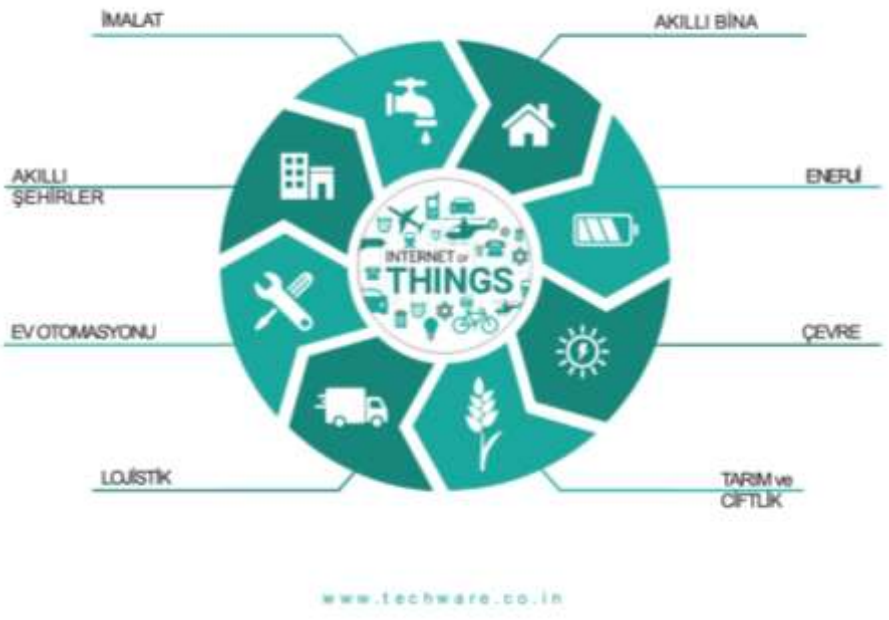


Şekil 1.5 Nesnelerin İnterneti Mimarisi [35].

Bulut tabanlı IoT mimarisi IoT cihazları ve bulut arka ucu olmak üzere iki katmandan oluşmaktadır. IoT tabanlı IoT mimarisinde ise ortada kenar katmanı olmak üzere üç katmandan oluşmaktadır. Üç katmanın orta katmanı olan kenar katmanı, bulut ve IoT ağları arasında köprü oluşturmada ve ara yüz oluşturmada önemli bir rol oymaktadır. Bu katmandaki bir kenar öğesi, herhangi bir küçük boyutlu veya orta boyutlu bilgi işlem varlığı olabilir. Bu öğe, üç katmandan herhangi birine dağıtılan uygulamalara depolama, bilgi işlem, ağ kaynağı ve ağ denetimi sağlamayı amaçlamaktadır. Bu varlığın formu, düşük güçlü düğümlerden hücresel baz istasyonlarına kadar farklı durumlara göre farklılık gösterir. Bu varlıklar bulut servis sağlayıcıları, IoT ağı kullanıcıları veya bir Telekom operatörü tarafından sahip olunabilir ve işletilebilir. Bu durum, Telekom operatörünün veya bulut hizmeti sağlayıcılarının ne kadar ve ne şekilde hizmet sağlamak istediğine bağlıdır [14].

Kenar ağlar ile bulut arasında bir orta katman kullanmak, modern ağ altyapılarında yaygın bir uygulamadır. Geleneksel orta katman işlevleri esas olarak yönlendirme, ağ bağlantısı ve ağ odaklı işlemleri içerir. İşlevselliklerin bir parçası olarak IoT giderek daha popüler hale gelmekte ve daha fazla uygulamada kullanılmaktadır [15]. Özellikle IoT ekosistemleri için, IoT ağları belirli bir kullanım durumu için özel bir altyapıya ihtiyaç duyar, ancak buluttaki kullanım durumundan bağımsızdır. IoT bulut ve IoT ağlarını orta katmandan ayırarak bu talepleri karşılayabilir ve bulutla ara yüz için standartlaştırılmış bir çıktı elde etmek için verileri soyutlayabilir.

IoT uygulamaları Şekil 1.6'da verilmiştir [44].



Şekil 1.6 IoT uygulamaları [44].

Kenar katmanının varlığı, zorlu IoT hizmetlerinin performans gereksinimlerini esas olarak gecikme ve güvenlik açısından karşılayabilir [16]. Kenar sunucusu yalnızca IoT ağından üretilen verileri işlemekle kalmaz, çalışma zamanında IoT ağının sağlanmasına da yardımcı olabilir. Bunlar, bir IoT ağı için önemli iyileştirmelerdir. Bu iyileştirme ihtiyacının iki sebebi vardır. Birincisi, IoT düğümleri düşük iletişim, hesaplama ve pil kaynağına sahiptir. Bu nedenle, ağ yapılandırma uygulamalarını her zaman gerçekleştiremezler. İkincisi ise kenar sunucuların yardımıyla IoT ağları daha iyi performansa sahip olmaktadır.

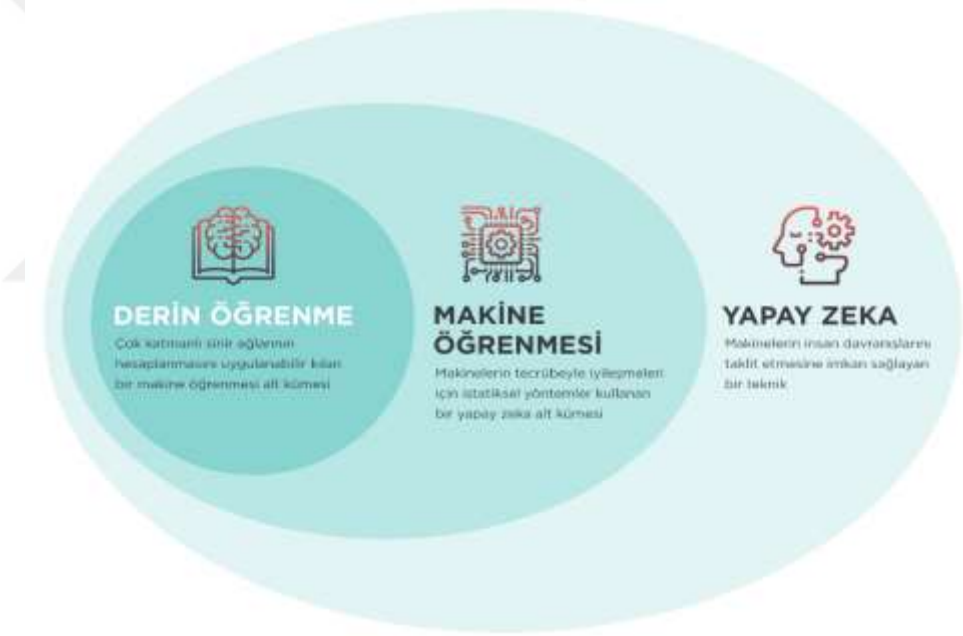
Bulut tarafından gerçek zamanlı uygulamalar için mesafe, verilere erişme ve getirme gecikmesini çok yüksek hale getirmektedir. Ayrıca omurga iletişim masrafı çok yüksek olmaktadır. Sonuç olarak dinamik ağlarda IoT yapılandırması önemlidir ve en uygulanabilir yol olarak karşımıza çıkmaktadır [45].

1.3. Derin Öğrenme

Derin öğrenme, ham veri girdisinden aşamalı olarak üst düzey özellikleri çıkarmak için birden çok katman kullanan bir makine öğrenmesi algoritmaları sınıfıdır[11]. Örneğin, görüntü işlemede düşük katmanlar kenarları belirleyebilirken; daha yüksek katmanlar, rakamlar, harfler veya yüzler gibi insanla ilgili kavramları belirleyebilir.

Derin sinir ağları, derin inanç ağları, tekrarlayan sinir ağları ve evrişimli sinir ağları gibi derin öğrenme mimarileri, bilgisayarla görme, makine görüşü, konuşma tanıma, doğal dil işleme, ses tanıma, sosyal ağ filtreleme, makine çevirisi, biyoinformatik, ilaç tasarımı, tıbbi görüntü analizi, malzeme denetimi ve masa oyunu programları gibi alanlara uygulanmıştır. Bu uygulamalarda insanlarla yarışır hatta bazı durumlarda insanüstü sonuçlar elde edilmiştir [46].

Şekil 1.7'de Yapay zekâ, makine öğrenimi ve derin öğrenmenin ilişkileri gösterilmiştir [44].

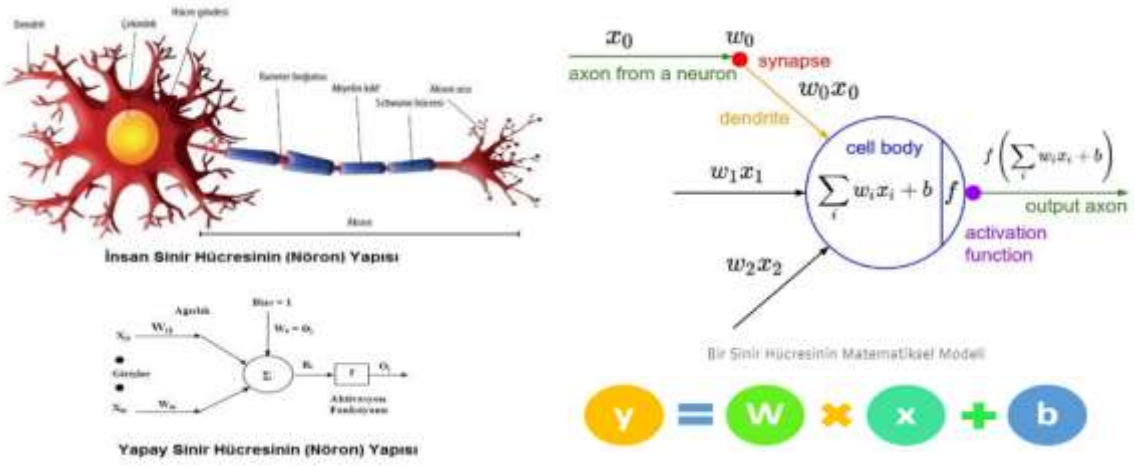


Şekil 1.7 Yapay Zekâ [44].

Derin Öğrenme Yapay Derin Sinir Ağı kullanımını içeren bir Makine Öğrenme yöntemidir. İnsan beyninin, sinyalleri göndererek ve alarak bilgiyi işleyen sinir hücreleri veya nöronlardan oluşması gibi, derin öğrenmedeki derin sinir ağı, birbirleriyle iletişim kuran ve işlem bilgisi olan 'nöronların' katmanlarından oluşur [37].

Derin Öğrenmedeki "derin", ağ içindeki katmanların sayısını gösterir. Yani daha fazla katman sayısı, daha derin olan ağdır.

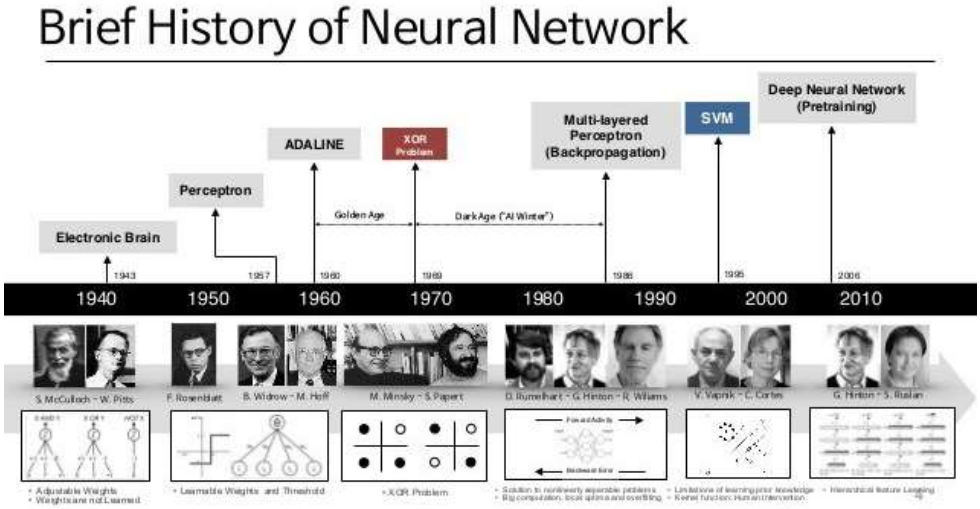
Yapay Zekâ Sinir Hücresinin (Nöron) Yapası Şekil 1.8'de verilmştir [47]



Şekil 1.8 Yapay Zekâ Sinir Hücresinin (Nöron) Yapası [47]

Derin öğrenmedeki "derin" sıfatı, ağdaki birden çok katmanın kullanımından gelmektedir. İlk çalışmalar, doğrusal bir algının evrensel bir sınıflandırıcı olamayacağını ve daha sonra, sınırsız genişlikte bir gizli katmana sahip polinom olmayan aktivasyon işlevine sahip bir ağın böyle olabileceğini göstermiştir. Derin öğrenme, hafif koşullar altında teorik evrenselliği korurken, pratik uygulamaya ve optimize edilmiş uygulamaya izin veren sınırsız sayıda sınırlı boyutlu katmanlarla ilgilenen modern bir varyasyondur. Derin öğrenmede, katmanların heterojen olmasıyla birlikte; verimlilik, eğitilebilirlik ve anlaşılabilirlik adına biyolojik olarak bilgilendirilmiş bağlantı modellerinden büyük ölçüde sapmasına izin verilir, bu da "yapılandırılmış" kısımdır [47].

Şekil 1.9'da Sinir Ağlarının Kısa Tarihi verilmiştir [48].

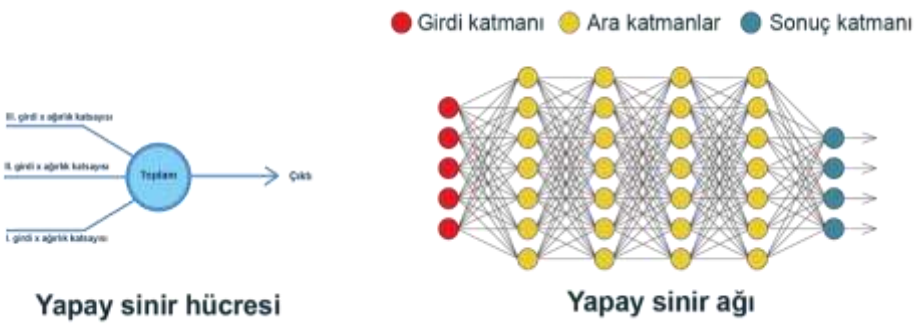


Şekil 1.9 Sinir Ağlarının Kısa Tarihi [48].

1.3.1. Neural Network (NN)

Yapay sinir ağı olarak da adlandırılan bir sinir ağı, Şekil 1.10'de gösterildiği gibi birçok basit işlem biriminden oluşan bir yazılım sistemidir. Bu birimler çok basittir ve sadece basit işlemlere sahiptir. Örneğin, bir birim, ağırlığı ve önyargılı doğrusal bir işlev veya sigmoid, düzeltilmiş doğrusal birim, hiperbolik tanjant işlevleri gibi doğrusal olmayan hesaplamalar olabilir. Ancak bu birimlerin büyük bir kısmı organize edildiğinde, sinir ağları karmaşık işleri bitirebilir [24]. Fikir, insan beyninden ve insanın öğrenme sürecinden esinlenmiştir. İnsan beyni, son derece karmaşık, doğrusal olmayan ve paralel hesaplamalar gerçekleştirmek için muazzam miktarda nöron kullanır. Örneğin etkileşimler ve eğitim yoluyla öğrenen bebekler bu nöronları bilgi edinmek için kademeli bir şekilde eğitilerek öğrenirler [49].

Bir sinir ağında, işlem notu, insan beynindeki nöronun eşdeğer bileşenidir. İnsan beynindeki nöronları bağlama şeklini simüle eden düğümler, benzer veya farklı işlevleri gerçekleştirmek için birkaç katmana bölünür ve her katman arasındaki düğümler, değişken bağlantı ağırlıklarıyla bağlanır. Ağı eğitmek, bu ağırlıkları eğitmektir. Model, belirli bir probleme göre uygun şekilde seçilirse, yeterli miktarda eğitimden sonra, model bir noktada birleşir ve eğitim veri setine göre tahminlerde bulunur [25]. Donanım hesaplama kapasitesinin gelişmesiyle birlikte, üretim ve araştırmada kullanılan makine öğrenmesinde sinir ağları en başarılı yöntemler haline gelmiştir. Sinir ağları ile pek çok araştırma yapılmakta ve pek çok sorun göreve özel herhangi bir kural ile programlama yapmadan çözülmektedir. Bu, yeterli tarihsel veri varsa, karmaşık sorunları çözmek için sinir ağlarının kullanımını kolaylaştırır ve kullanışlı hale getirir. İleri Beslemeli Sinir Ağları (Feedforward Neural Networks), düğümler arasındaki bağlantıların bir döngü oluşturmadığı temel ve en basit sinir ağlarıdır. Şekil 1.10'de giriş katmanı, gizli katman ve çıktı katmanı olmak üzere üç katmandan oluşan bir ileri beslemeli sinir ağı gösterilmektedir. Her katmanın beş işleme birimi vardır ve bu ileri beslemeli sinir ağı için girdi boyutu beş ve çıktı boyutu beştir [50].



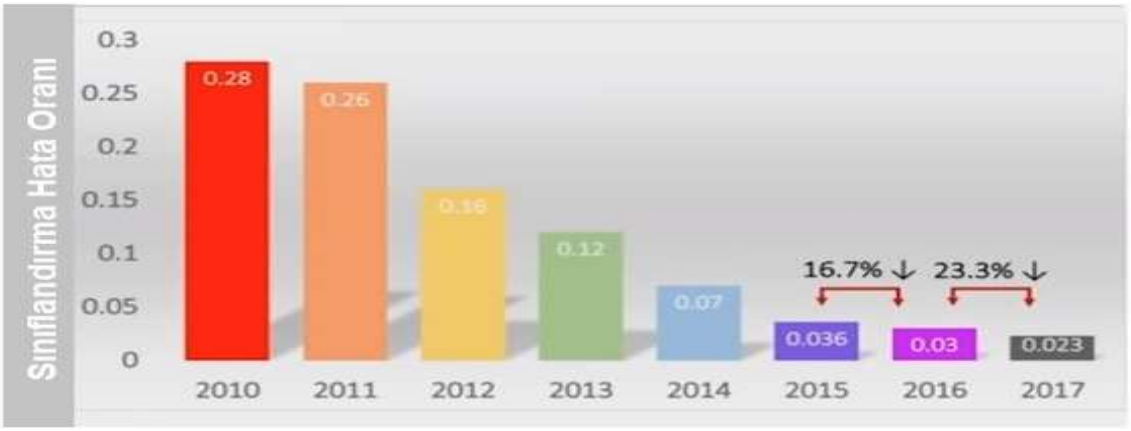
Şekil 1.10 Yapay Zekâ sinir ağı [48].

Yapay Sinir Ağlarının farklı yükseltilmiş sınıfları da bulunmaktadır. Örneğin, Evrişimli (Convolutional) Sinir Ağları (CNN'ler) bunlardan biridir. CNN'ler bir görüntünün kısmi alanlarında kalıp bulmada başarılı olduklarından, örüntü tanımada büyük başarı elde edilmektedir [1].

Tekrarlayan sinir ağları, konuşma tanıma, el yazısı tanıma, dil modelleme, çeviri ve görüntü altyazısında yaygın olarak kullanılırken, dâhili durumlarını (bellek) girdi dizilerini, özellikle de zamansal dizileri işlemek için kullanabilirler [19].

1.3.2. Derin Öğrenme Hata Oranı

Bir insanın gördüğü görüntüyü akılda tutma yeteneği %95'tir. Makineler 2015 yılına kadar bu algılama yeteneğine yaklaşamamıştır. 2010 yılında %75 algılama yeteneği başarılı sayılabilecek bir düzey olurken; sonrasında makinelerin çok büyük bir gelişimi olmuş ve Derin öğrenme ile beraber 2015 yılında insanların %95 algısına geçilmiş, 2017 yılında ise %98'lere kadar ilerlemiştir. Artık gördüklerini ya da duyduklarını insanlardan daha iyi anlayabilme seviyesine ulaşmışlardır. Hata oranının yıllara göre gelişimi Şekil 1.11'de gösterilmiştir [51].



Şekil 1.11 2015 Yılında Makinelerin Algılama Yeteneği [48].

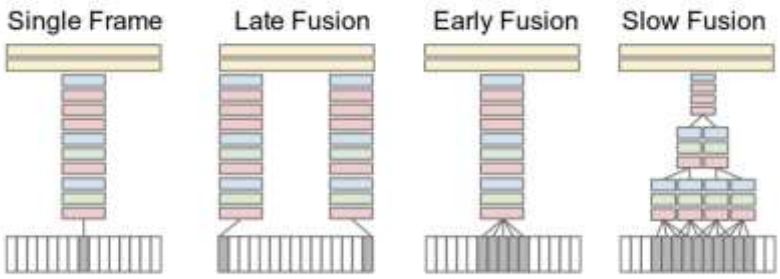
1.4. Derin Öğrenme İle Eylem Algılama Mimarileri

Eylem tahmini, ilk çerçevelere bakarak eylemin erken tanınmasıdır ve ideal olarak çerçevelerin % 50'sinin altında olması yeterli olmaktadır. Artık hem izole hem de sürekli videolar üzerinde çalışmalar yapılmaktadır. İdeal olarak, bir eylem tahmin yöntemi gerçek zamanlı olarak çalışmalı ve eylemlerin başlangıç ve bitiş noktalarının tespitini içeren yakın gelecekteki eylemleri 2-3 saniyeden önce tahmin etmelidir. Eylem algılamada kullanılan veri kümesi kullanılabilir, ancak insanların birkaç saniye önceden tahmin edebileceği yeni veri kümeleri de gereklidir. Örneğin sosyal robotlar için etkileşim senaryoları, kendi kendine giden arabalar, diğer sürücülerin ve yayaların dönüşünü tahmin etme vb. örnek gösterilebilir [5][52].

1.4.1. 2014'ten 2019'a Kadar En Son Teknolojiye Bir Bakış

Derin Öğrenme Ortaya Çıkması; 2014'ten sonra, derin öğrenme mimarileri UCF101, Sports-1M ve HMDB51 gibi dönüm noktası niteliğindeki video işlem tanıma veri kümelerinde son teknoloji performansıyla galip gelmiştir. 2014 yılında, yayımlanan iki önemli makale video tanımada derin öğrenmenin başlangıcı sayılabilir. Karpathy ve arkadaşları tarafından Evrişimli Sinir Ağları ile Büyük Ölçekli Video Sınıflandırması [2] ve Simonyan ile Zisserman tarafından yazılan Videolarda Eylem Tanıma için İki Akımlı Evrişimli Ağlar [3], eylem tanımada tek akış ve iki akış ağının popülerliğini artırmıştır.

Karpathy ve arkadaşları zamansal verilerin tek bir akışlı 2D evrişimli sinir ağı ile nasıl birleştirileceğini araştırmış ve Şekil 1.12'de önerilen mimarileri test etmişlerdir.



Şekil 1.12 Tek akışlı ağ mimarileri [53].

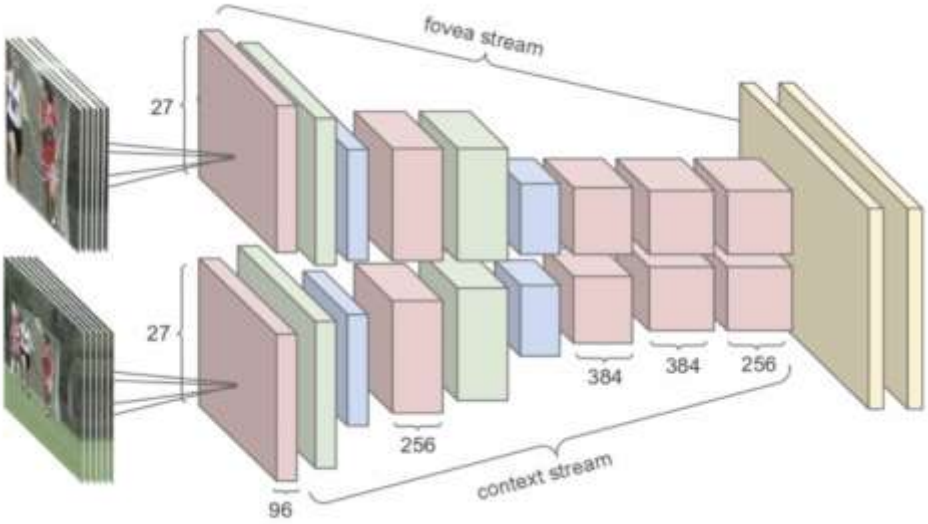
Şekil 1.12' de kırmızı, yeşil ve mavi kutular sırasıyla evrişimli, normalleştirme ve havuzlama katmanlarını göstermektedir. Yavaş Füzyon modelinde, gösterilen sütunlar parametreleri paylaşır.

Tek çerçeve ağı (single frame network), esasen geçici özellikleri olmayan bir görüntü sınıflandırma ağıdır. Geç füzyon (Late fusion), birbirinden uzak iki çerçeve kullanır ve iki çerçeveden işlenen özellikleri düzleştirerek yoğun şekilde bağlı katmanlarda derin hiyerarşik özellikleri birleştirir. Erken füzyon (Early fusion), kareleri kanallar olarak yığınlar ve tüm kare yığını üzerinde 2D Convolution aracılığıyla video tanımlayıcılarını öğrenir. Yavaş füzyon, bir çerçeve yığınından özellikleri hiyerarşik bir şekilde birleştirmeye çalışır, böylece ağ derinleştikçe, daha geçici özellikler öğrenilir. Sonuçları Çizelge 1.1'de özetlenmiştir.

Çizelge 1.1 Sports-1M test setinin 200.000 videosunun sonuçları [54].

Model	Clip Hit@1	Video Hit@1	Video Hit@5
Feature Histograms + Neural Net	-	55.3	-
Single-Frame	41.1	59.3	77.7
Single-Frame + Multires	**42.4**	**60.0**	**78.5**
Single-Frame Fovea Only	30.0	49.9	72.8
Single-Frame Context Only	38.1	56.0	77.2
Early Fusion	38.9	57.7	76.8
Late Fusion	40.7	59.3	78.7
Slow Fusion	**41.9**	**60.9**	**80.2**
CNN Average (Single+Early+Late+Slow)	41.4	63.9	82.4

Tek akış stratejisi, büyük ölçekli görüntü veri kümelerinde eğitilmiş modellerden aktarımla öğrenmeyi kullanabilme yönüyle güçlüdür. Ayrıca bu mimaride RGB görüntü verileri doğrudan kullanıldığından, görüntülerin optik akış için önceden işlenmesine gerek kalmamaktadır. Bu, tek akışlı ağları gerçek zamanlı işlem için bir aday yapmaktadır. Yazarlar önerdikleri füzyon mimarilerinde derin bir 2D CNN'den parametre sayısının önemli ölçüde arttığını ve bunu hafifletmek için çok çözünürlüklü bir akış kullanmayı önermektedir. Şekil 1.13'de tüm videodan düşük çözünürlüklü bir bağlam akışı ile birlikte videonun bir orta kesimine yüksek çözünürlüklü bir fovea akışını yerleştiren bir video gösterilmektedir[53].



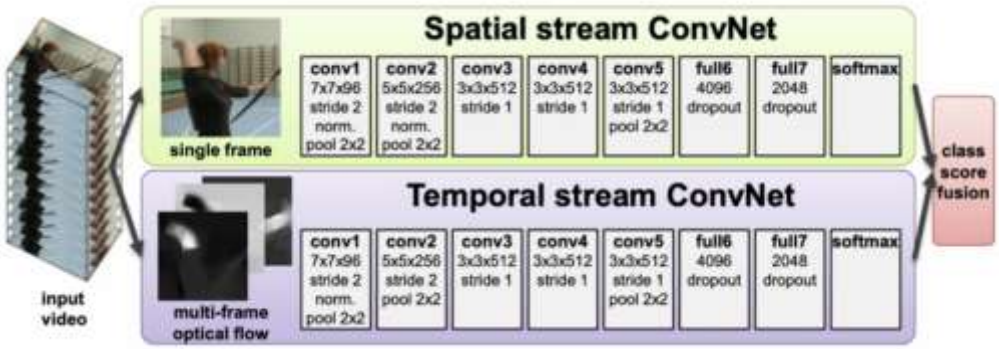
Şekil 1.13 Çok çözünürlüklü akış [53].

Şekil 1.13' de alternatif evrişim (kırmızı), normalleştirme (yeşil) ve havuzlama (mavi) katmanlarından oluşan iki akışa sahiptir. Her iki akış da birbirine tamamen bağlı iki katmana (sarı) birleşir.

Ampirik olarak, bu, parametre sayısını büyük ölçüde tek çerçeve mimarisiyle karşılaştırılabilir bir büyüklük sırasına düşürmüştür.

Çizelge 1.1'de yayınlanan sonuçlardan, önerilen mimarilerin yalnızca tek bir çerçeveyi etkin bir şekilde kullanarak performansı artırmada başarısız olduğu görülmektedir. Bu modellerin zayıf yönü, hareket özelliklerini iyi yakalamamalarıdır. Ancak bu çalışma, transfer öğrenmenin eylem tanıma için çok yararlı olduğunu ortaya koymuştur. Sports-1M'de önceden eğitilmiş ve ardından en üstteki 3 katman üzerinde hassas bir şekilde ayarlanmış modeller, UCF101'de sıfırdan eğitilmiş bir modele kıyasla UCF101 veri kümesinde doğruluğu % 20'nin üzerinde arttırmaktadır [2].

2014 yılında; Simonyan ve Zisserman, mekansal ve zamansal özellikleri Şekil 1.14'teki gibi ayrı ayrı işleyen iki akışlı bir mimari önermiştir [3][53].



Şekil 1.14 İki akışlı Ağ [55].

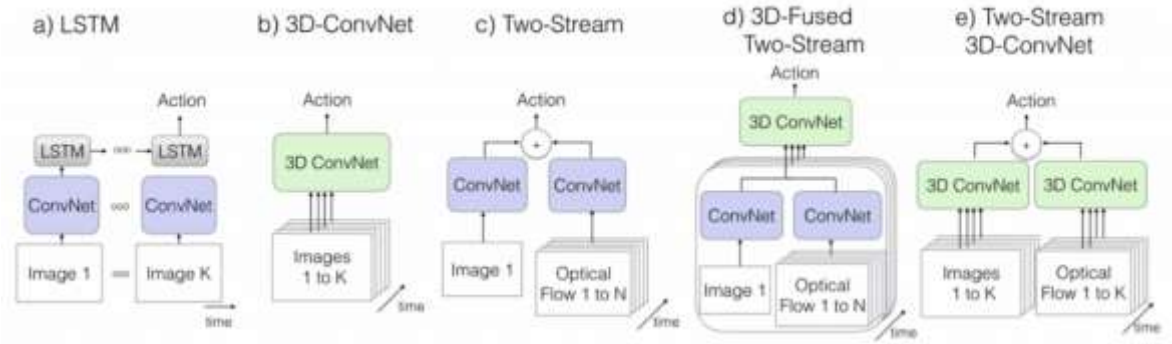
Video için tek bir çerçeve bir 2D evrişim ağına aktarılırken, önceden işlenmiş çok kareli optik akış ayrı bir 2D evrişim ağına aktarılır. Her akış bir tahmin oluşturur ve bunların füzyonu sınıf puanını belirler. Bu mimarinin dezavantajı, optik akışın ayrı olarak hesaplanması gerektiğinden ve her iki akışın da ayrı ayrı eğitilmesi gerektiğinden uçtan uca eğitilebilir olmamasıdır. Uzamsal akış, büyük görüntü veri kümelerinden bilgi alabilirken, zamansal akış bir video veri kümesinde eğitilmelidir. Bu şekilde, transfer öğrenimi bu mimari için tamamen geçerli değildir. Ayrıca, optik akışı hesaplamak için gerekli olan ön işlem, bu algoritmanın gerçek zamanlı yeteneklere sahip olmasını zorlaştırır. Çizelge 1.2 Bölünmede ortalama ConvNet doğrulukları [55].'de de görüldüğü gibi, IDT gibi zamanın en son tekniklerine uyma yeteneği, bu yaklaşımın güçlü yanıdır [55].

Çizelge 1.2 Bölünmede ortalama ConvNet doğrulukları [55].

Method	UCF-101	HMDB-51
Improved dense trajectories (IDT) [26, 27]	85.9%	57.2%
IDT with higher-dimensional encodings [20]	**87.9%**	61.1%
IDT with stacked Fisher encoding [21] (based on Deep Fisher Net [23])	-	**66.8%**
Spatio-temporal HMAX network [11, 16]	-	22.8%
"Slow fusion" spatio-temporal ConvNet [14]	65.4%	-
Spatial stream ConvNet	73.0%	40.5%
Temporal stream ConvNet	83.7%	54.6%
Two-stream model (fusion by averaging)	86.9%	58.0%
Two-stream model (fusion by SVM)	**88.0%**	**59.4%**

Bu teknik, video sınıflandırması için derin öğrenmeye yönelik daha fazla araştırmanın kapısını açmıştır. Bu araştırma, evrişimli derin ağların bazı hareket özelliklerini etkili bir şekilde yakalayabildiğini ve bunu, eylem sınıfları için doğru tahminler oluşturmak için uzamsal özelliklerle birleştirebileceğini göstermiştir.

2014'ten 2019 yılına kadar geliştirilen derin öğrenme mimarileri, büyük ölçüde Şekil 1.15'te gösterilen mimariler etrafındaki varyasyonları takip etmiştir.



Şekil 1.15 Eylem tanıma için çeşitli mimariler [56]

Şekil 1.15'te K, bir videodaki toplam kare sayısını, N ise videonun komşu karelerinin bir alt kümesini belirtir.

Sırasıyla bir LSTM ve bir 3D ConvNet kullanan ilk iki yaklaşım a) ve b), uçtan uca eğitilebilir ve gerçek zamanlı yetenekli olmanın gücünü paylaşır. Bunun nedeni, optik akışa güvenmemeleri ve bunun yerine bu bilgiyi kodlayan özellikleri öğrenmeleri gerektiğidir. Bu, ağın zamansal özellikleri doğrudan uçtan uca eğitimde öğrenmesine olanak tanır. c) ve e) yaklaşımları, ham veriler üzerinde optik akış hesaplamaları gerektirdikleri için gerçek zamanlı yetenekli veya uçtan-uca eğitilebilir değildir. b), d) ve e) yaklaşımları 3B evrişimleri kullanır. Bu, geleneksel 2D ConvNets'ten çok daha fazla parametre oluşturur. UCF101 veri kümesi için eğitilmiş tek bir 3B evrişimli sinir ağı, 2B durumda sadece 5M + parametrelere kıyasla 33M + parametrelere sahip olabilir [4]. Sports-1M'de eğitilmiş 3D ConvNet modelleri yaklaşık 2 ay sürdüğü için bu, eğitim maliyetini önemli ölçüde etkiler. Bu, video verileri için doğru mimariyi aramayı zorlaştırır. Çok sayıda parametre, aşırı uyum riski de yaratır.

Videolar için LSTM mimarisi, Donahue ve arkadaşları tarafından 2014 tarihli Long-term Recurrent Convolutional Networks for Visual Recognition and Description makalesinde popüler hale getirilmiştir [5]. Mimari, LRCN olarak bilinmektedir. Kodlayıcı-kod çözücü mimarisinin doğrudan bir uzantısıdır. Ancak sadece video gösterimleri içindir. LRCN ağının gücü, çeşitli uzunluklardaki dizileri yönetebilmesinden gelir. Ayrıca, resim yazısı ve video açıklaması gibi diğer video görevlerine de uyarlanabilir. LRCN'nin, zamanında en son teknolojiyi yenememiş olması zayıflık olarak görülse de .

Çizelge 1.3'te belirtildiği gibi, tek çerçeve mimarileri üzerinde iyileştirmeler sağlamıştır [55].

Çizelge 1.3 Etkinlik tanıma: RGB ve akış girişleriyle UCF101 [25] veri kümesinde etkinlik tanıma için tek çerçeve modellerini LRCN ağlarıyla karşılaştırma [57].

Model	Single Input Type		Weighted Average	
	RGB	Flow	$1/2, 1/2$	$1/3, 2/3$
Single frame	67.37	74.37	75.46	78.94
LRCN-fc_6	**68.20**	**77.28**	80.90	**82.34**

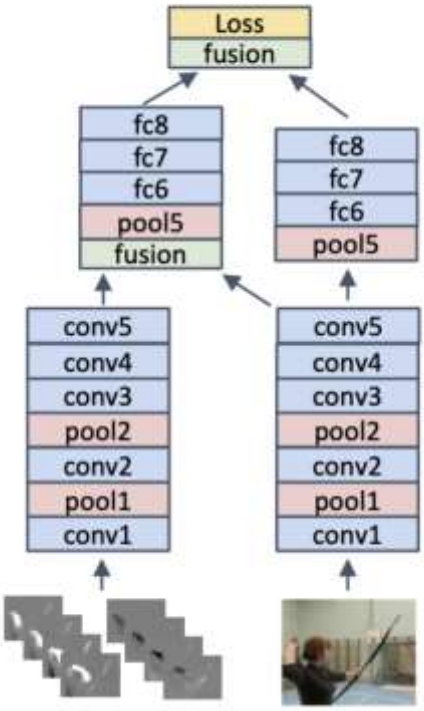
Uzamsal özelliklerin zamansal modellemesi, gizli bir tekrarlayan katmanın öğrenmesi için zordur. Ampirik olarak, RGB modellerine daha fazla gizli birim eklemek son 256 gizli birimi iyileştirmemiştir. Bununla birlikte, Flow girişini kullanırken daha fazla gizli birim eklemek, 256 birimden 1024 birime % 1,7'lik bir doğruluk artışı sağlamıştır. Bu, LRCN'nin zor bir zaman öğrenme optik akışına veya benzer bir hareket doğal temsiline sahip olduğunu gösterir, Çizelge 1.4'de UCF101'de eylem tanıma sonuçları verilmiştir [58].

Çizelge 1.4 UCF101'de eylem tanıma sonuçları. C3D, 2015 yılında ana hatlar ve son teknoloji yöntemlerle karşılaştırılmıştır [59].

Method	Accuracy (%)
Imagenet + linear SVM	68.8
iDT w/ BoW + linear SVM	76.2
Deep networks [18]	65.4
Spatial stream network [36]	72.6
LRCN [6]	71.1
LSTM composite model [39]	75.8
C3D (1 net) + linear SVM	82.3
C3D (3 nets) + linear SVM	**85.2**
iDT w/ Fisher vector [31]	87.9
Temporal stream network [36]	83.7
Two-stream networks [36]	88.0
LRCN [6]	82.9
LSTM composite model [39]	84.3
Conv. pooling on long clips [29]	88.2
LSTM on long clips [29]	88.6
Multi-skip feature stacking [25]	89.1
C3D (3 nets) + iDT + linear SVM	**90.4**

3D ConvNets; Du Tran ve arkadaşları tarafından 2015 yılında yazılan 3D Convolutional Networks ile Learning SpatIoTemporal Features [6] araştırma makalesinde son teknoloji ürünü olarak öne sürülmüştür. Bu makale, 3x3x3 çekirdekli 3B evrişim ağının (C3D) uzay-zamansal özelliklerini öğrenmede en etkili olduğunu ortaya koymaktadır. İlginç bir şekilde, çözülmeler, ağın ilk birkaç kare için uzamsal görünümü öğrendiğini ve ardından bir video çekiminin sonraki karelerinde belirgin hareketin izlediğini ortaya koymuştur. Bu mimari, C3D 313 fps'ye kadar işlerken birçok videonun gerçek zamanlı olarak işlenebilmesi açısından güçlüdür. Bu ağ tarafından

üretilen video tanımlayıcılar da kompakt ve ayırt edicidir. Çünkü konvolüsyonlar tarafından üretilen özellikler PCA aracılığıyla 10 boyuta yansıtılabilir ve yine de UCF101 veri setinde % 52,8 doğruluk elde edilebilir. Her iki ağ kulesinin tutulduğu iki katmanda füzyon mimarisi Şekil 1.16'de verilmiştir [60].



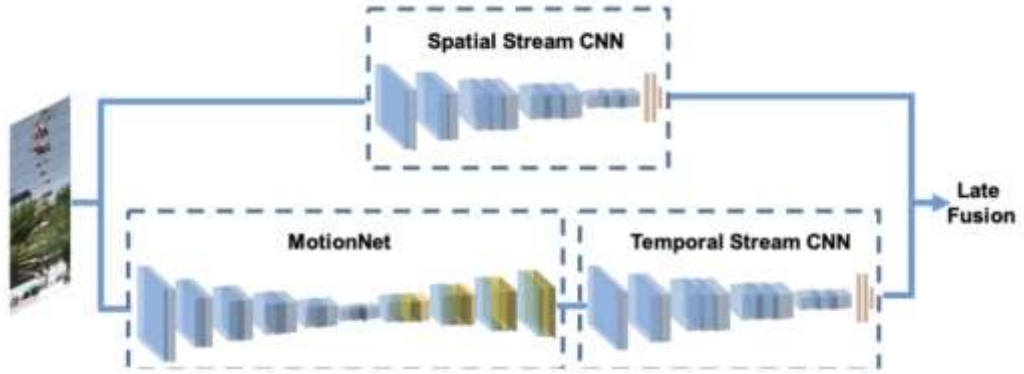
Şekil 1.16 Her iki ağ kulesinin tutulduğu iki katmanda füzyon mimarisi (conv5'ten ve fc8'den sonra), biri hibrit uzay-zamansal ağ ve diğeri tamamen uzamsal bir ağ [61].

2016'da; çalışmaların odak noktası iki akış ağına geri döndüğü görülmektedir. Video Eylem Tanıma için Evrişimli İki Akışlı Ağ Füzyonunda, Zisserman ve arkadaşları [7] da uzamsal ve zamansal verilerin akışlar arasında nasıl etkili bir şekilde birleştirileceğini ve uzun vadeli geçici bağımlılıkları idare edebilecek çok seviyeli kayıp yaratmayı ele almışlardır. Buradaki motive edici fikir, görüntünün farklı bölümlerindeki saçları taramak ve dişleri fırçalamak gibi benzer hareketleri ayırt etmek için, ağın bir piksel konumunda uzamsal özellikler ve hareket özelliklerinin bir kombinasyonunu almasını gerektirir [61].

Teorik olarak, yoğun şekilde bağlanmış katmanlardan önce akışları birleştiren yöntemler bunu başarabilir. Önerilen mimaride yazarlar, Şekil 1.17'te gösterildiği gibi, iki akışı iki konumda birleştirir. Bu ağ, farklı alt ağlarda daha iyi yakalanan hareket ve uzamsal özellikler ve son teknoloji IDT ve C3D yaklaşımlarını geride bırakır.

Çok seviyeli kayıp, son füzyon katmanında bir uzay-zamansal kayıp ve geçici ağın çıktısından oluşan ayrı bir zamansal kayıptan oluşur. Bu durum, araştırmacıların uzay-zamansal özellikler oluşturmasına ve uzun vadeli zamansal bağımlılıkları modellemesine izin vermiştir. Bu yöntem hala orijinal iki akış ağının zayıflıklarından muzdariptir. Ancak gerçek dünyadaki önyargılarımıza daha iyi hizmet eden gelişmiş bir mimari nedeniyle daha iyi performans gösterir [62].

2017 yılında; Zhu ve arkadaşları MotionNet [8] adlı optik akışı öğrenen gizli bir akış sunarak iki akışlı ağları bir adım ileri götürmüşlerdir. Bu uçtan uca yaklaşım, araştırmacıların optik akışı açıkça hesaplamayı atlamasına izin vermiştir. Bu durum, Şekil 1.17'de gibi iki akış yaklaşımının artık gerçek zamanlı olabileceği ve yanlış tahminlerden kaynaklanan hataların daha optimum optik akış özellikleri için MotionNet'e de yayılabileceği anlamına gelmektedir [60].



Şekil 1.17 MotionNet, girdi olarak ardışık video karelerini alır ve hareketi tahmin eder. Daha sonra zamansal akış CNN, hareket bilgisini eylem etiketlerine yansıtmayı öğrenir [60].

Araştırmacılar, gizli iki akışlı CNN'nin gizli olmayan yaklaşımlara benzer bir doğrulukta performans gösterdiğini, ancak Çizelge 1.5'te görüldüğü gibi artık saniyede 10 kata kadar daha fazla kare işleyebildiğini keşfetmişlerdir. Bu, iki akış yöntemi için gerçek zamanlı yetenekler sağlamaktadır.

Çizelge 1.5 İki akışlı yaklaşımlar ve bunların UCF101'deki doğruluğu [60].

Method	Accuracy (%)	fps
Two-Stream CNNs [19]	88.0	14.3
Very Deep Two-Stream CNNs [24]	**90.9**	**12.8**
Hidden Two-Stream CNNs (a)	87.50	120.48
Hidden Two-Stream CNNs (b)	87.99	120.48
Hidden Two-Stream CNNs (c)	**89.82**	**120.48**

MotionNet alt ağı genişletilebilir ve optik akış hesaplamasının gerekli olduğu diğer derin öğrenme yöntemlerine uygulanabilir. Bu özellik gerçek zamanlı olarak başka yaklaşımlar yapılmasına izin verdiği için önemlidir.

2017'de Zisserman ve arkadaşlarının, Quo Vadis, Action Recognition Yeni Model ve Kinetik Veri Seti, başlıklı çalışması C3D'yi iki akış ağından öğrendikleriyle birleştirerek bir adım daha ileri götürmüştür [4]. Araştırmacılar, yeni bir iki akışlı şişirilmiş 3D ConvNet (I3D) önermektedir. 2D ConvNets'ten gelen filtreler ve havuzlama çekirdekleri 3D'ye genişletilerek onlara ekstra bir zamansal boyut kazandırılır. Bu durumda araştırmacıların 2D sınıflandırma için başarılı mimariler almasını ve bunları 3D'ye uygulamasını sağlar. Araştırmacılar ayrıca ImageNet gibi büyük görüntü veri kümeleri üzerinde eğitilmiş 2D ConvNet modellerinden gelen parametrelerle bu 3D filtreleri önyükleme yapma gücüne sahiptir.

UCF-101 ve HMDB-51 test setlerinde (her ikisinin 1'ini ayırarak) performansını Çizelge 1.6'da gösterilmiştir.

Çizelge 1.6 ImageNet önceden eğitilmiş ağırlıklar olmadan başlayan mimariler için UCF-101 ve HMDB-51 test setlerinde (her ikisinin 1'ini ayırarak) performansını gösterir [56].

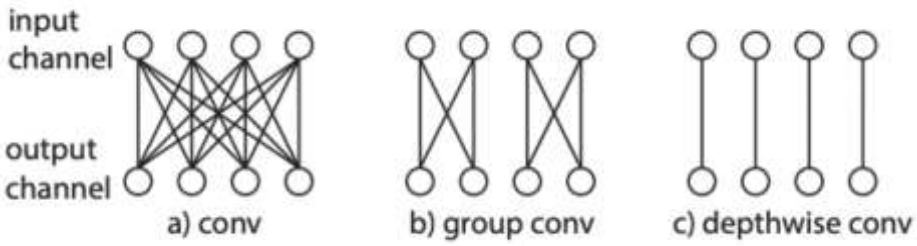
Architecture	UCF-101		
	Original	Fixed	Full-FT
(a) LSTM	81.0 / 54.2	88.1 / 82.6	91.0 / 86.8
(b) 3D-ConvNet	– / 51.6	– / 76.0	– / 79.9
(c) Two-Stream	91.2 / 83.6	93.9 / 93.3	94.2 / 93.8
(d) 3D-Fused	89.3 / 69.5	94.3 / 89.8	94.2 / 91.5
(e) Two-Stream I3D	93.4 / 88.8	97.7 / 97.4	98.0 / 97.6

İki akışlı bir mimaride sıralı RGB çerçevelerinde ve sıralı optik akış çerçevelerinde 3D ConvNets kullanmak, araştırmacıların UCF101'de son teknolojiyi geçmesini sağlamıştır. Araştırmacılar, Kinetics veri setinin kullanılmasıyla transfer öğrenmenin açık önemini ortaya koymuştur. Bununla birlikte kullandıkları model mimarisi uçtan uca eğitilebilir nitelikte olmamakla birlikte, gerçek zamanlı yeteneklere de sahip değildir.

2017'den 2018'e kadar; derin arttırmanın öğrenme üzerinde birçok ilerleme kaydettiği, 3DResNet ve sözde kalıntı C3D (P3D) gibi yeni mimarilere yol açmıştır [9]. Son teknoloji üzerinde de etkileri olmuştur.

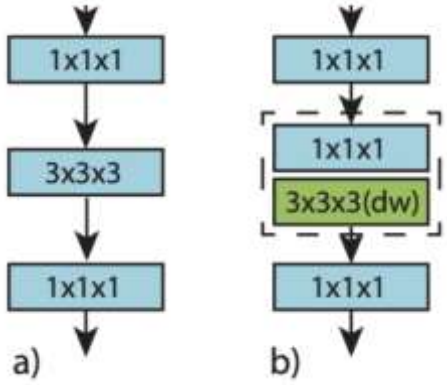
Son olarak, Haziran 2019'da Du Tran ve arkadaşları Kanalla Ayrılmış Evrişimli Ağlarla Video Sınıflandırmasında eylem tanıma görevi için kanalla ayrılmış evrişimli ağları (CSN) önermektedir [10]. Araştırmacılar, Xception ve MobileNet modellerinde büyük başarı elde eden grup evrişimi ve derinlemesine evrişim fikirlerini geliştirmişlerdir.

Gruıp evrişimleri Şekil 1.18'de verilmiştir.



Şekil 1.18 (a) Sadece bir gruba sahip konvansiyonel bir evrişim. **(b)** 2 gruplu bir grup evrişimi. **(c)** Grup sayısının giriş / çıkış filtrelerinin sayısıyla bir evrişim [56].

Temel olarak, grup evrişimleri, tam olarak bağlanmayarak düzenlileştirme ve daha az hesaplama sağlamaktadır. Derinlik-bilge evrişimler, Şekil 1.19'de görüldüğü gibi, giriş ve çıkış kanallarının grup sayısına eşit olduğu grup evrişimlerinin uç durumudur.



Şekil 1.19 (a) Standart bir ResNet darboğaz bloğu. **(b)** Bir etkileşim korumalı darboğaz bloğu [60].

Araştırmacılar, 3x3x3 evrişim çekirdeklerini iki farklı katmana ayırmayı önermektedir. İlk katman, yerel kanal etkileşimi için 1x1x1'lik bir evrişimdir ve ikinci katman, yerel uzay-zamansal etkileşimler için 3x3x3'lük derinlikte bir evrişimdir. Araştırmacılar, bu blokları kullanarak ağdaki parametre sayısını önemli ölçüde azaltır ve güçlü bir düzenlilik biçimi sunmaktadır. Kanalla ayrılmış bloklar, ağın farklı katmanlardaki uzamsal ve uzamsal-zamansal özellikleri yerel olarak öğrenmesine izin vermektedir.

Sports-1M'deki son teknoloji mimarilerle karşılaştırılması Çizelge 1.7'de verilmiştir.

Çizelge 1.7 Sports-1M'deki son teknoloji mimarilerle karşılaştırılması [63]

Method	input	video@1	video@5
C3D [30]	RGB	61.1	85.2
P3D [24]	RGB	66.4	87.4
Conv pool [40]	RGB+OF	71.7	90.4
R(2+1)D [31]	RGB	73.0	91.5
R(2+1)D [31]	RGB+OF	73.3	91.9
ir-CSN-101	RGB	74.8	92.6
ip-CSN-101	RGB	74.9	92.6
ir-CSN-152	RGB	**75.5**	**92.7**
ip-CSN-152	RGB	**75.5**	**92.8**

Çizelge 1.7'de gösterildiği gibi CSN, Sports-1M veri kümesinde R (2 + 1) D, C3D ve P3D gibi son teknoloji RGB yöntemlerini geliştirmektedir. Ağ, çıkarım sırasında da 2–4 kat daha hızlıdır. Model ayrıca sıfırdan eğitilmiştir. Tablodaki modellerin geri kalanı, ImageNet veya Kinetics veri setinde önceden eğitilmiştir. Bu yeni mimari, önceki faktörlü ağları geliştirirken, aşırı uyumu azaltır, olağanüstü derecede hızlıdır ve kıyaslama veri kümelerinde son teknoloji ürünü "doğruluk" üretir.

Eylem tanıma için en son teknoloji (Ağustos 2019), kanalla ayrılmış ağdır. Bu ağ, mekânsal ve mekânsal-zamansal özellikleri kendi farklı katmanlarında etkili bir şekilde yakalar. Kanalla ayrılmış evrişim blokları, bu özellikleri ayrı bir şekilde öğrenir, ancak bunları tüm evrişim aşamalarında yerel olarak birleştirir. Bu durum, zamansal ve uzamsal iki akış ağlarının yavaş füzyonunu gerçekleştirme ihtiyacını azaltır. Ağın aynı zamanda, ağın iki boyut arasında karıştırılan özellikleri öğrenmeye karar verebildiği C3D'deki gibi uzamsal veya zamansal özellikleri öğrenme arasında karar vermesi gerekmez. Bu ağ, 2B uzamsal dilimlerin doğal bir görüntü oluşturması gerektiği yönündeki önyargıyı etkili bir şekilde yakalar; burada, zamansal yöndeki 2B bir dilim farklı zamansal özelliklere sahiptir ve doğal manifolda düşmez. Bu şekilde, araştırmacılar, her bir yönü işlemek için iki ayrı katman oluşturarak bu önyargıyı güçlendirirler. Kanal ayrımı, eylem tanımada ileriye doğru atılan önemli bir adımdır ve sıfırdan eğitildiğinde bile son teknoloji sonuçlarını geride bırakmıştır. Aynı zamanda gerçek zamanlı çıkarım yapılabilir. Bu özellikleriyle, CSN'lerin mevcut son teknolojilerde öne çıkmasını sağlamaktadır.

1.4.2. Derin Öğrenmede Amaçlanan Doğrultular

Derin öğrenmenin, eylem tanıma için videoları işleme yöntemimizde devrim yarattığı görülmektedir. Derin öğrenme literatürü, geliştirilmiş Yoğun Yörüngeleri kullanmakla başlayan uzun bir yol kat etmiştir. Kardeş görüntü sınıflandırma probleminden birçok öğrenim, eylem tanıma için derin ağların ilerletilmesinde kullanılmıştır. Spesifik olarak, evrişim katmanlarının, havuz katmanlarının, toplu normalizasyonun ve artık bağlantıların kullanımı 2B alandan ödünç alınmış ve 3B olarak büyük bir başarıyla uygulanmıştır. Uzamsal akış kullanan birçok model, kapsamlı görüntü veri kümeleri üzerinde önceden eğitilmiştir. Optik akış, iki akış ağı ve füzyon ağları gibi erken dönem derin video mimarilerindeki zamansal özellikleri temsil etmede de önemli bir role sahiptir. Optik akış, sonraki karelerde hareketin tüm pikseller için yoğun olarak hesaplanmış akış vektörleri olarak tanımlanabileceğine nasıl inandığımıza dair matematiksel bir tanımdır. Başlangıçta ağlar, optik akışı kullanarak performansı arttırmıştır. Ancak bu, ağları uçtan uca eğitilemez hale getirmiş ve gerçek zamanlı yetenekleri sınırlamıştır. Modern derin öğrenmede, optik akışın ötesine geçilmiş ve bunun yerine geçici yerleştirmeleri yerel olarak öğrenebilen ve uçtan uca eğitilebilir ağlar tasarlanmaktadır [64].

Eylem tanımanın kendine özgü karmaşıklıklarıyla gerçekten benzersiz bir sorun olduğu ortadadır. İlk ihtilaf kaynağı, 3B evrişimlerle ilişkili yüksek hesaplama ve bellek maliyetidir. Bazı modellerin modern GPU'larda Sports-1M üzerinde eğitim alması 2 aydan fazla sürer. İkinci ihtilaf kaynağı, video mimarisi araması için standart bir kıyaslama olmamasıdır [11]. Sports-1M ve UCF101 ile yüksek düzeyde ilişkilidir. Yanlış etiket atama, bir videonun bir bölümü için eğitilmek üzere seçildiğinde yaygındır. Ancak videonun başka bir bölümünde olabileceği için aslında gerçek eylemi içermeyebilir. Son ihtilaf kaynağı ise, bir video derin sinir ağı tasarlamanın önemsiz olmasıdır. Katman seçimi, girdinin nasıl önceden işleneceği ve zamansal boyutun nasıl modelleneceği açık bir sorundur. Zaman hiyerarşisiyle verilen çalışma ve gelişmeler temelde bu sorunları deneysel bir şekilde ele almaya ve videolardaki zamansal modellemeyi çözen yeni mimariler önerme üzerinedir [65].

2. MATERYAL VE YÖNTEM

Bu bölümde, yapılmış deneylerden yola çıkarak ortaya atılan prosedürü, mantığı ve yöntem seçimini açıklamak ve geliştirmek amacındayız. Yapılmış olan deneylerin amacı, bu hesaplama açısından sınırlı ortamda derin öğrenme modelini çalıştırarak, eylem algılama algoritmalarının performansını deneyimlemek ve uygulanabilirliğini değerlendirmektir. Göz önünde bulundurulan derin öğrenme çerçeveleri, TensorFlow, TensorFlow Lite ve TensorFlow Lite Micro'dur. Dolayısıyla bu çerçeveler son teknoloji performansını sağlayarak, yaygın olarak tercih edilir. Faz I'de, araçların ve tekniklerin incelemesi sunulmuş, çözüm tasarımı ve veri toplama önerileri paylaşılmıştır. Ardından Faz II'de, belirlenen donanım, yazılım ve önceden eğitilmiş model için yapılan uygulama çalışmaları geliştirilerek açıklanmış ve önerilen model test edilmiştir. En son Faz III'te önerilen model çalıştırılmış, veriler toplanarak gerekli analizler yapılmıştır.

Nihayetinde, istenen sonuç, makul bir sürede çıkarım yapma denetimi altında mevcut buluşsal yöntem tabanlı modelden daha iyi performans gösteren bir veya daha fazla ağı belirlemektedir.

2.1. Çalışmada Kullanılan Araç ve Teknoloji Modelleri

Çalışmadaki kullanılan araç ve teknoloji modelleri Çizelge 2.1'de olarak göstermektedir.

Çizelge 2.1 Çalışmada Kullanılan Araç ve Teknoloji Modelleri

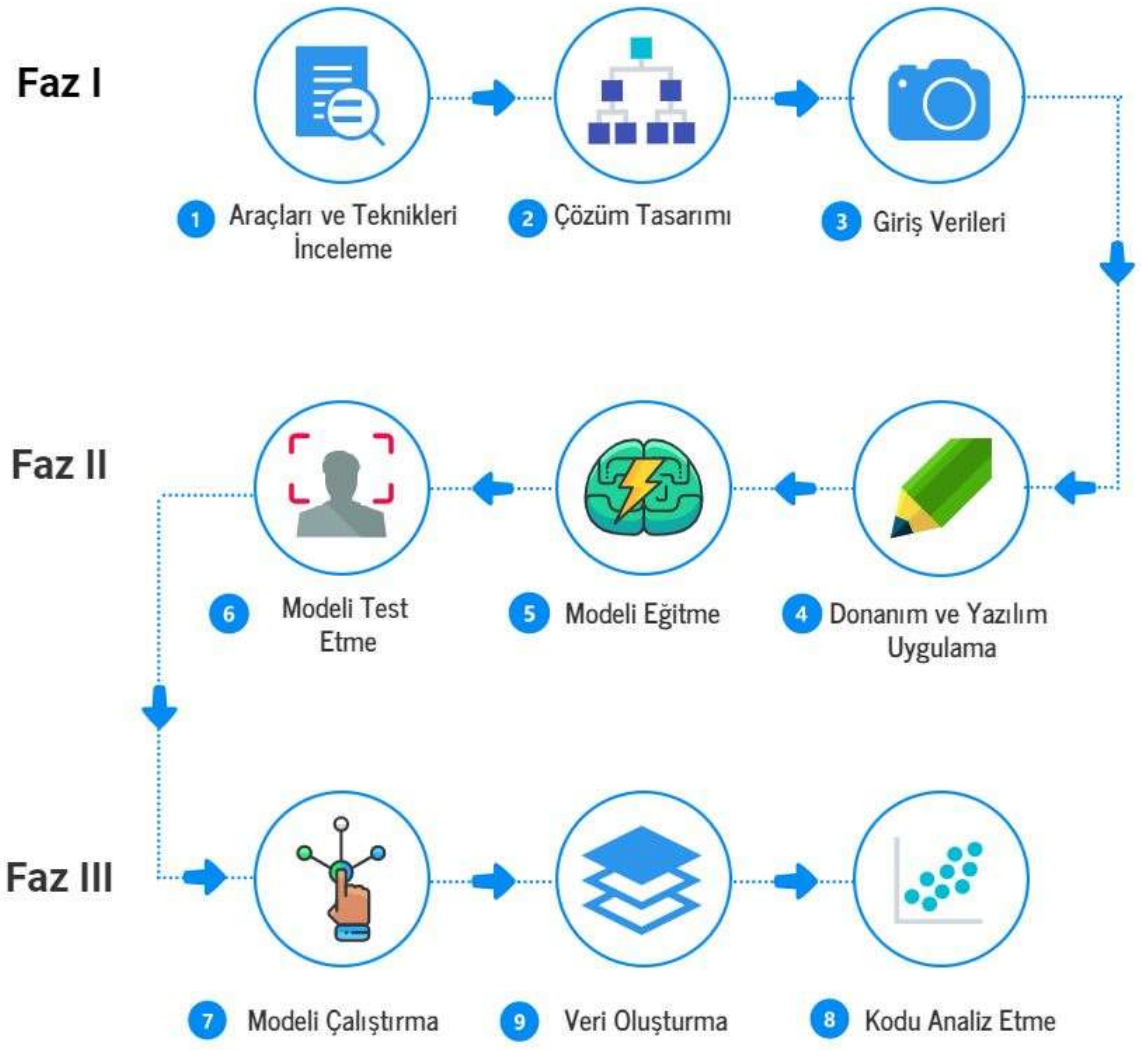
No.	Araç ile Teknoloji Modelleri
1.	ESP32-CAM Embedded Device
2.	Raspberry Pi 3B IoT cihaz
3.	Raspberry Pi 4B IoT cihaz
4.	Movidius Intel® Movidius™ NCS
5.	Coral TPU google USB stick
6.	Arduino UNO IoT cihaz
7.	Thermometer IR Sensor
8.	I2C LCD Screen
9.	VMware WorkStation virtual machine as IoT cihaz emulator
10.	Android Studio Emulator
11.	Android 8.1 (Oreo), API level 27
12.	Android SDK Tools, Revision 26.1.1
13.	Python IDE NetBeans as a Python Editor
14.	Arduino IDE 1.8.13
15.	Balena Etcher v1.5.111 OS image writer
16.	TensorFlow
17.	TensorFlow Lite
18.	TensorFlow Lite for Microcontroller
19.	OpenCV 4 computer vision library
20.	SSD MobileNet
21.	COCO Dataset

2.2. Deneysel Tasarım

IoT cihazlarda derin öğrenme metodu kullanılarak eylem algılama için oluşturulacak algoritma aşağıdaki gibi üç faza ayrılmıştır. Her bir faz ayrı ayrı açıklanmaktadır.

- **Faz I:** Seçilen tekniklerin İncelemesi, girdi verilerinin, çözüm tasarımı.
- **Faz II:** Kodların uygulanması, CNN'nin eğitilmesi, modelin test edilmesi
- **Faz III:** Modelin çalıştırılması, sonuçların elde edilmesi ve kod analizi

Yapılan çalışma Şekil 2.1'de gibidir:

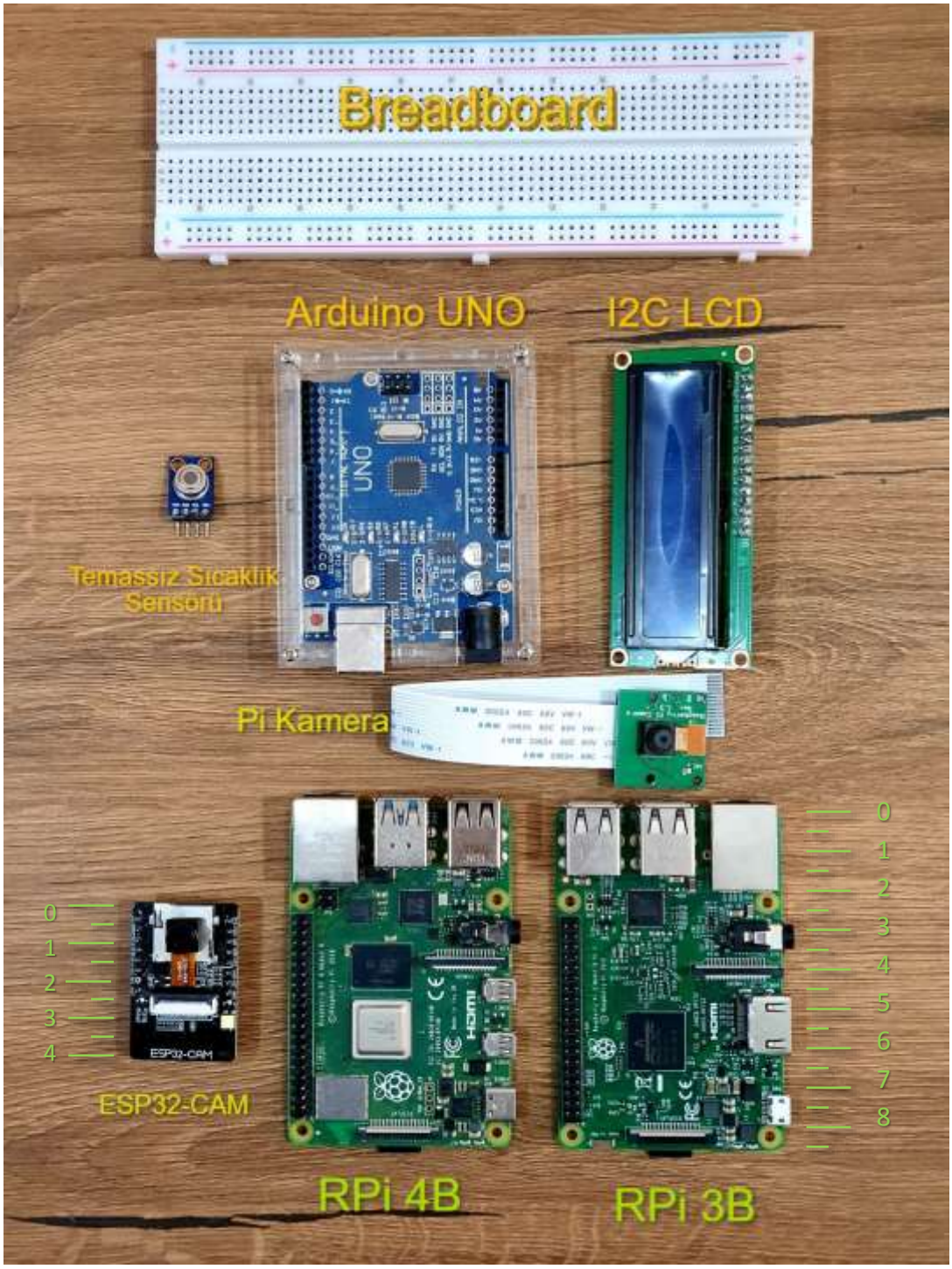


Şekil 2.1 Metodoloji fazların akış şeması

2.3. Faz I: Seçilen Tekniklerin İncelemesi, Girdi Verilerinin, Çözüm Tasarımı

2.3.1. Seçilen Yazılım ve Kullanılan Teknik İncelenmesi

Burada, kullanılacak teknik ve araçlar gözden geçirilmiştir. Şekil 2.2'de kullanılan birincil IoT cihazları gösterilmektedir.



Şekil 2.2 Kullanılan IoT Cihazları

VMware Workstation Pro; Windows veya Linux PC üzerinde aynı anda birden fazla sanal makine, OCI kapsayıcı ve Kubernetes kümesi çalıştırılmasına olanak tanır. Kod geliştirmede, çözüm mimarisinde, uygulama testlerinde, ürün tanıtımlarında ve daha fazlasında kullanılmak üzere, tam özellikli ve güvenli bir şekilde izole edilmiş Linux ve Windows VM'leri ve diğer masaüstü, sunucu ve bulut ortamları, yapılandırılabilir sanal ağ ve ağ durumu simülasyonu imkânı tanır [66].

Python; modern, öğrenmesi kolay, nesne yönelimli bir programlama dilidir. Güçlü bir yerleşik veri türleri kümesine ve kullanımı kolay denetim yapılarına sahiptir. Python yorumlanmış bir dil olduğundan, en kolay şekilde yalnızca etkileşimli oturumlara bakarak ve bunları açıklayarak gözden geçirir [59] [60].

C ile C ++; Bjarne Stroustrup tarafından C programlama dilinin bir uzantısı olarak oluşturulmuş genel amaçlı bir programlama dilidir.

C, statik tip sistemle yapılandırılmış programlamayı, sözcüksel değişken kapsamını ve özyinelemeyi destekleyen genel amaçlı, prosedürel bir bilgisayar programlama dilidir. C, belirli makine komutlarıyla verimli bir şekilde eşleşen yapılar sağlar [63] [64].

TensorFlow; el yazısıyla yazılmış rakam sınıflandırması, görüntü tanıma, kelime gömme, tekrarlayan sinir ağları, makine çevirisi için diziden diziye modeller, doğal dil işleme ve PDE (kısmi diferansiyel denklem) tabanlı simülasyonlar için derin sinir ağlarını eğitebilir ve çalıştırabilir. CPU'ları ve NVidia GPU'ları destekler. Ubuntu Linux, macOS, Android, iOS ve (eskisinden daha iyi) Windows üzerinde çalışır. Esnek bir şekilde Eğitim için kullanılan aynı modellerle üretim tahminini ölçekli olarak desteklemeye devam edebilir. Ayrıca Otomatik farklılaştırma yapar, TensorBoard'da model görselleştirme aracına sahiptir ve (R ve Scala programcıları) hala Python dilinden kullanım için en iyi desteği sunmaktadır [65].

TensorFlow'un mobil; IoT ve yerleşik cihazlar için Lite çözümüdür. TensorFlow her zaman birçok platformda çalışmıştır, ancak Derin Öğrenme modellerinin benimsenmesi son birkaç yılda katlanarak arttığından, bunların mobil ve gömülü cihazlarda kullanılması gerekmektedir. TensorFlow Lite, cihazdaki makine öğrenimi modellerinin düşük gecikmeli çıkarımını sağlar. Dahası, geliştiricilerin TensorFlow modellerini mobil, gömülü ve IoT cihazlarında çalıştırmalarına yardımcı olan bir dizi araçtır. Düşük gecikme süresi ve küçük bir ikili boyut ile cihaz üzerinde makine öğrenimi çıkarımını mümkün kılar [27] [59].

Mikrodenetleyiciler için TensorFlow Lite; yalnızca birkaç kilobayt bellek içeren mikro denetleyiciler ve diğer cihazlarda makine öğrenimi modellerini çalıştırmak için tasarlanmıştır. Çekirdek çalışma zamanı, bir Arm Cortex M3'e 16 KB'ye sığar ve birçok temel modeli

çalıştırabilir. İşletim sistemi desteği, herhangi bir standart C veya C ++ kitaplığı veya dinamik bellek ayırma gerektirmez [65] [27] [33].

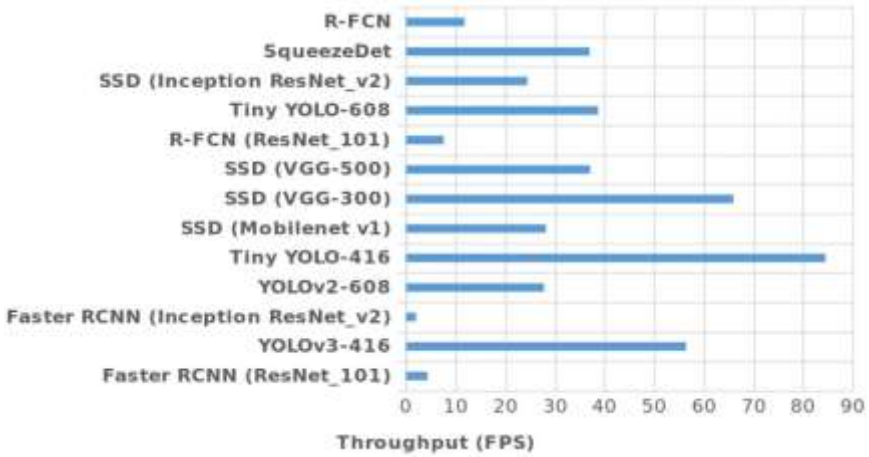
Mikrodenetleyiciler; için TensorFlow Lite, C ++ 11 ile yazılmıştır ve 32 bitlik bir platform gerektirir. Arm Cortex-M Serisi mimarisine dayalı birçok işlemci ile kapsamlı bir şekilde test edilmiş ve ESP32 dâhil olmak üzere diğer mimarilere taşınmıştır. Çerçeve (framework) bir Arduino kütüphanesi olarak mevcuttur. Ayrıca Mbed gibi geliştirme ortamları için projeler üretebilir. Açık kaynaklıdır ve herhangi bir C ++ 11 projesine dâhil edilebilir [67].

Geliştirme panoları, Çizelge 2.2'de gösterildiği gibi Mikrodenetleyiciler için TensorFlow Lite tarafından desteklenen cihazlarda.

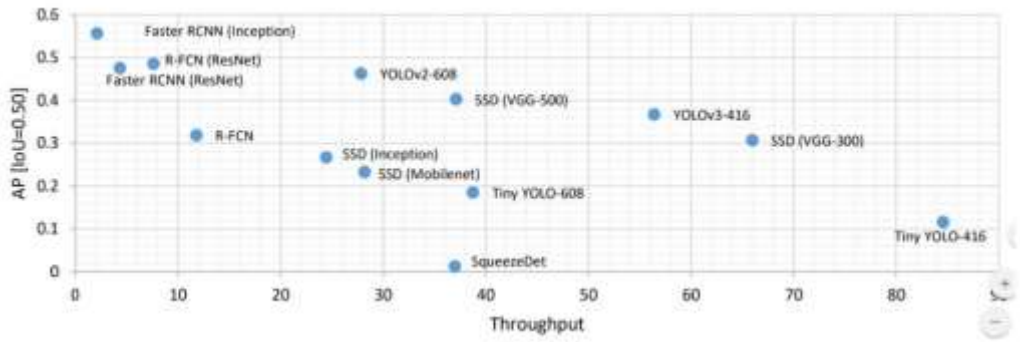
Çizelge 2.2 Tensorflow Lite Tarafindan Desteklenen Cihazlar [67]

No	TensorFlow Lite Tarafından Desteklenen Cihazları
1	Arduino Nano 33 BLE Sense
2	SparkFun Edge
3	STM32F746 Discovery kit
4	Adafruit EdgeBadge
5	Adafruit TensorFlow Lite for Microcontrollers Kit
6	Adafruit Circuit Playground Bluefruit
7	Espressif ESP32-DevKitC
8	Espressif ESP-EYE
9	Wio Terminal: ATSAMD51
10	Himax WE-I Plus EVB Endpoint AI Development Board
11	Synopsys DesignWare ARC EM Software Development Platform

MS COCO; veri kümesi (Lin ve arkadaşları, 2014), görüntü başına beş farklı açıklama içeren 123.287 görüntüden oluşmaktadır. Bu veri kümesindeki görüntüler, 80 nesne kategorisi için açıklanmıştır. Yani bu kategorilerden birinin tüm örneklerin etrafındaki sınırlayıcı kutuların tüm görüntüler için kullanılabilir olduğu anlamına gelmektedir. MS COCO veri kümesi, standart değerlendirme sunucusu tarafından kolaylaştırılan bir şey olan, görüntü tanımlaması için yaygın olarak kullanılmaktadır [14]. MS COCO'nun uzantıları, soruların ve yanıtların eklenmesi de dâhil olmak üzere şu anda geliştirme aşamasındadır [26]. Çeşitli modellerde sürekli çıkarım hızı Şekil 2.3'de, Aktarım hızındaki Ortalama Hassasiyet Şekil 2.4'de verilmiştir [68]



Şekil 2.3 Çeşitli modellerde sürekli çıkarım hızı [68].

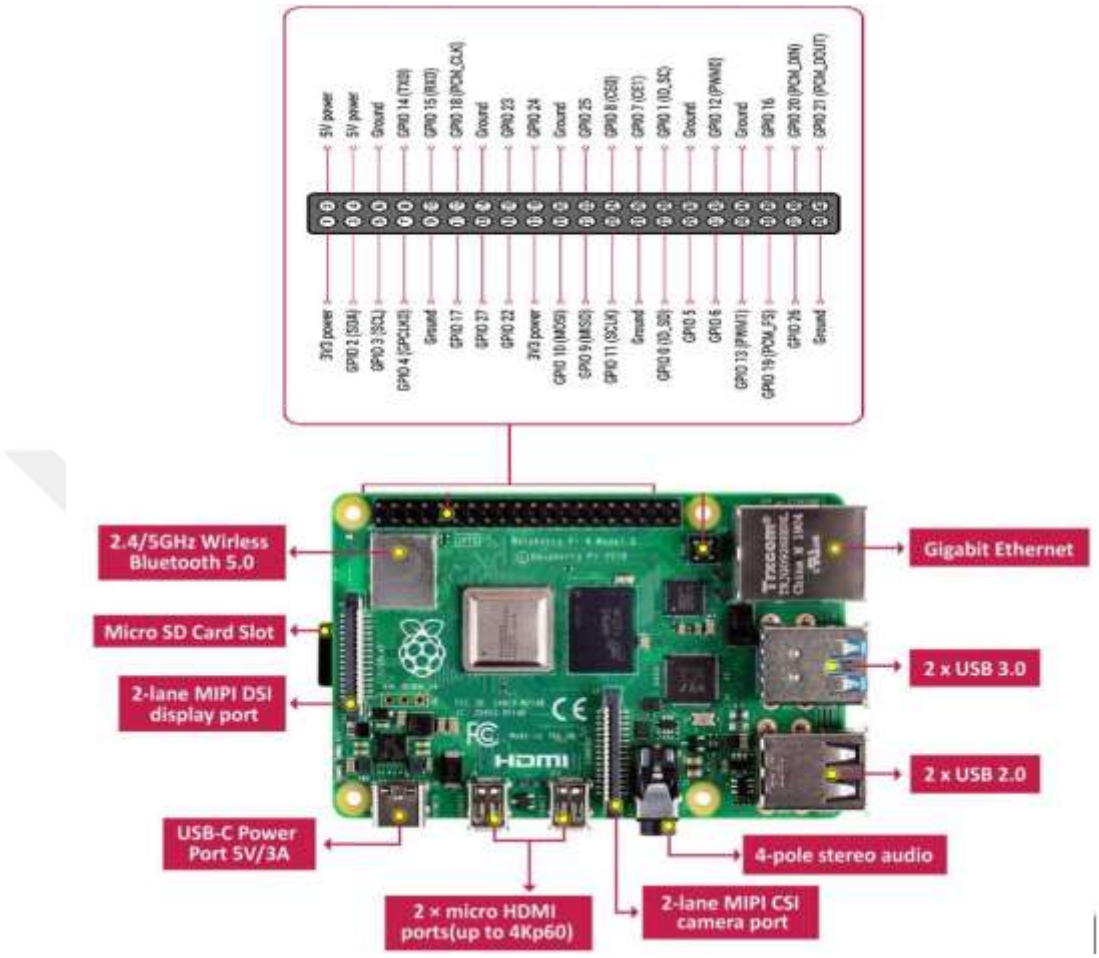


Şekil 2.4 Aktarım hızındaki Ortalama Hassasiyet [IoU = 0,5] [68].

OpenCV; C ++, C, Python ve Java arayüzlerine sahip olan ve hesaplama verimliliği için Windows, Linux, Mac OS, iOS ve Android'i destekleyen bir araçtır. Optimize edilmiş C / C ++ ile yazılmış gerçek zamanlı uygulamalara güçlü bir odaklanma ile kitaplık, çok çekirdekli işlemden yararlanabilir ve aynı şekilde temelde yatan heterojen bilgi işlem platformunun donanım hızlandırmasından da yararlanılabilir [69].

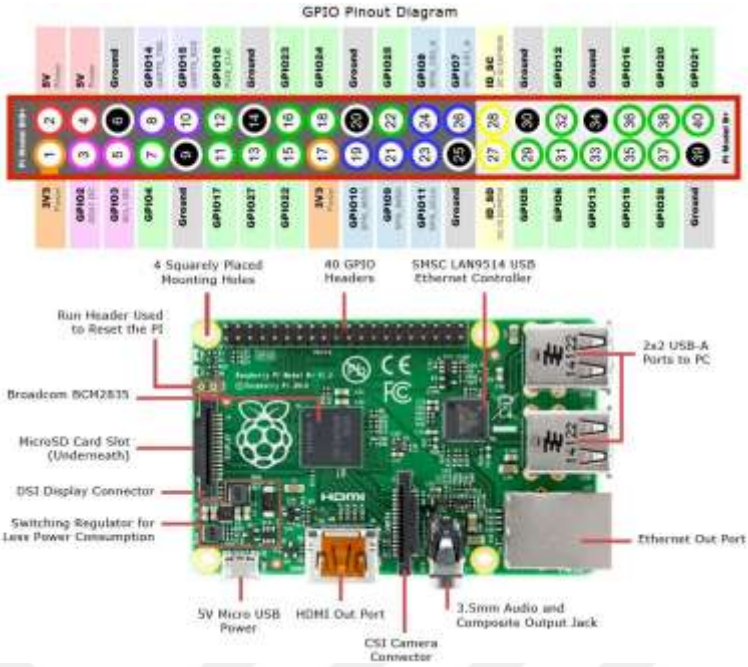
Raspberry Pi4 B İnceleme; 1,5 GHz 64 bit dört çekirdekli ARM Cortex-A72 işlemci, yerleşik 802.11ac Wi-Fi, Bluetooth 5, FULL Gb Ethernet (işlem hacmi sınırlı değil), iki USB 2.0 bağlantı noktası, iki USB ile Haziran 2019'da piyasaya sürülmüştür. 3.0 bağlantı noktası ve 4K çözünürlüğe kadar bir çift mikro HDMI (HDMI Tip D) bağlantı noktası aracılığıyla çift monitör desteği mevcuttur. Pi 4 ayrıca, uygun bir PSU ile kullanıldığında aşağı akış çevre birimlerine ek güç sağlanmasına olanak tanıyan bir USB-C bağlantı noktası üzerinden de çalıştırılabilir. İlk Raspberry Pi 4 kartının, Apple MacBook'larda kullanıldığı gibi e-işaretli üçüncü taraf USB kablolarının onu yanlış tanımladığı ve güç sağlamayı reddettiği bir tasarım kusuru vardır [68] [69]. Tom's Hardware 14 farklı kabloyu test etmiş ve bunlardan 11'inin Pi'yi sorunsuz bir şekilde açtığını

ve çalıştırdığını ortaya koymuştur [30]. Tasarım kusuru, 2019'un sonlarında yayınlanan Anakart'ın 1.2 revizyonunda düzeltilmiştir [28]. Raspberry 4 mimarisi Şekil 2.5'de verilmiştir.



Şekil 2.5 Raspberry 4 mimarisi [70]

Raspberry Pi3 B İnceleme; Şubat 2016'da 1,2 GHz 64 bit dört çekirdekli işlemci, yerleşik 802.11n Wi-Fi, Bluetooth ve USB önyükleme özellikleriyle piyasaya sürülmüştür [26]. Pi Günü 2018'de Raspberry Pi 3 Model B +, daha hızlı 1,4 GHz işlemci ve üç kat daha hızlı gigabit Ethernet (dâhili USB 2.0 bağlantısıyla yaklaşık 300 Mbps ile sınırlı verim) veya 2,4 / 5 GHz çift bant 802.11ac Wi-Fi ile piyasaya sürülmüştür (100 Mbps). Ayrıca Ethernet üzerinden Güç (PoE), USB önyüklemesi ve ağ önyüklemesi özellikleri mevcuttur. Çizelge 2.3'de Pi 3 cihazın teknik özellikleri gösterilmiştir [71]. Raspberry 3 mimarisi Şekil 2.6'de verilmiştir.



Şekil 2.6 Raspberry 3 mimarisi [41].

ESP32-CAM İnceleme; ESP32, entegre Wi-Fi ve çift modlu Bluetooth ile çipli bir mikro denetleyicide düşük maliyetli, düşük güç tüketen bir sistemdir. ESP32 serisi, hem çift çekirdekli hem de tek çekirdekli varyasyonlarda bir Tensilica Xtensa LX6 mikroişlemci kullanır ve dâhili anten anahtarları, RF balon güç amplifikatörü, düşük gürültülü alıcı amplifikatör, filtreler ve güç yönetimi modülleri içerir. ESP32, Şangay merkezli bir Çin şirketi olan Espressif Systems tarafından geliştirilmiş ve TSMC tarafından 40 nm prosesleri kullanılarak üretilmiştir. ESP8266 mikro denetleyicisinin halefidir. Şekil 2.7'de cihazın teknik özellikleri gösterilmiştir [72].

Şekil 2.7 ESP32-CAM mimarisi [72].

Google Coral USB İnceleme; Google Coral, uç uygulamalar için genel amaçlı bir makine öğrenimi platformudur. Bulutta eğitilmiş TensorFlow Lite modellerini çalıştırabilir. Google'ın Debian çeşidi olan Mendel Linux'a dayanmaktadır [73].

Nesne algılama, Google Coral için tipik bir uygulamadır. Video akışlarındaki nesneleri algılayan önceden eğitilmiş bir makine öğrenimi modeliniz varsa modelinizi Coral Edge TPU'ya dağıtabilir ve giriş olarak yerel bir video kamera kullanabilirsiniz. TPU, videoyu buluta aktarmak zorunda kalmadan nesneleri yerel olarak algılamaya başlayacaktır [74].

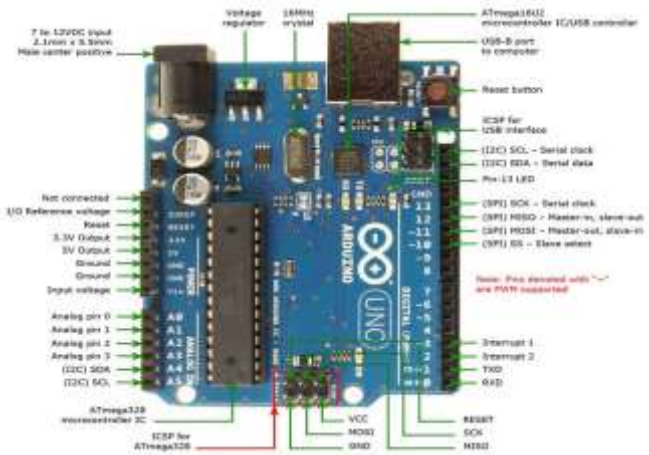
Coral Edge TPU yongası birkaç paket halinde mevcuttur. System-on-Module (SoM) içeren ve geliştirme için kullanımı kolay bağımsız geliştirme kartı bulunmaktadır. Buna alternatif olarak, bir USB, PCIe veya M.2 konektörü aracılığıyla bir PC'ye bağlı ayrı bir TPU hızlandırıcı cihazı kullanılabilir. Bir SoM ayrıca Şekil 2.8'da gösterildiği gibi özel donanıma entegre etmek için ayrı olarak da kullanılabilmektedir [75].

Şekil 2.8 Google Edge TPU coprocessor [73]

Movidius Intel® İnceleme; Intel® Movidius ™ VPU'lar zorlu bilgisayar görüşü ve son teknoloji yapay zekâ iş yüklerini verimli bir şekilde sağlar. VPU teknolojisi, Şekil 2.9'da gösterildiği gibi görsel perakende, güvenlik, koruma ve endüstriyel otomasyon gibi alanlarda derin sinir ağlarına ve bilgisayarla görmeye dayalı uygulamalara sahip akıllı kameralara, uç sunuculara ve AI (yapay zekâ) cihazlarına izin verir.

Şekil 2.9 Intel® Movidius™ Vision Processing Units (VPUs) [85]

Arduino-UNO İnceleme; Arduino Uno, Microchip ATmega328P mikrodenetleyicisine dayanan ve Arduino tarafından geliştirilen açık kaynaklı bir mikrodenetleyici kartıdır [2,3]. Kart, çeşitli genişletme kartları (kalkanlar) ve diğer devreler ile arayüzlenebilen dijital ve analog giriş/çıkış (I/ O) pimleri setleriyle donatılmıştır [1]. Anakartta 14 dijital I / O pinleri (altı PWM çıkışı kapasiteli), altı analog I/O pini ve bir B tipi USB kablosu aracılığıyla Arduino IDE (Entegre Geliştirme Ortamı) ile programlanabilir [4]. USB kablosuyla veya harici bir 9 voltluk pil ile çalıştırılabilmekle birlikte; 7 ile 20 volt arasındaki gerilimleri da kabul eder. Bu teknik Arduino Nano ve Leonardo'ya benzer [5,6]. Donanım referans tasarımı, Creative Commons Attribution-Share-Alike 2.5 lisansı altında dağıtılır ve Arduino web sitesinde mevcuttur. Donanımın bazı sürümleri için yerleşim ve üretim dosyaları da mevcuttur. Arduino-UNO Şekil 2.10'da gösterilmiştir.



Şekil 2.10 Arduino-UNO [76].

IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge 2.3 IoT Cihazların özelikleri [72] [76] [77].

Özellikler	RPi 3B	RPi 4B	ESP32-CAM	Arduino Uno
İmalatçı.	Raspberry Pi Foundation, UK	Raspberry Pi Foundation, UK	Espressif Sistemler, a Şangay merkezli Çin	Birçok
OS	Raspberry Pi OS (32-bit) Çekirdek sürümü 5.4	Raspberry Pi OS (32-bit) Çekirdek sürümü 5.4	Embedded Sistem	Embedded Sistem
CPU (İşlemci)	ARM Cortex-A53 (1.4 GHz)	ARM Cortex-A53 (1.4 GHz)	240 MHz 32-bit LX6 microprocessor	Mikroçip AVR (8-bit) 16MHz saat hızı
GPU	Broadcom VideoCore IV @ 400 MHz	Broadcom VideoCore IV @ 400 MHz	Hayır	Hayır
RAM	1GB LPDDR2	2GB LPDDR2	520 KB SRAM plus 4 MB PSRAM	SRAM 32k Flash Memory
Storage	23 GB miniSD	23 GB miniSD	4 MB	Flash, EEPROM
Kamera	Raspberry Pi Kamera V2.1 8 Megapiksel 1080p30 3.04 mm	Raspberry Pi Kamera V2.1 8 Megapiksel 1080p30 3.04 mm	OV2640 2 Megapiksel sahiptir Dizi boyutu UXGA 1622×1200 Image transfer rate of 15 - 60 fps	Hayır
Ethernet	Evet	Evet	Hayır	Evet
Wireless	Evet	Evet	Evet	Evet
Bluetooth	Evet	Evet	Evet	Evet
USB	4× USB2.0	2× USB2.0 2× USB3.0	Hayır	1xUSB2.0
Güç kaynağı	+5 Volt	+5 Volt	+5 Volt	+5 Volt
Maliyet	Yaklaşık. 380 TL	Yaklaşık. 395 TL	Yaklaşık. 120 TL	Yaklaşık. 180 TL

Arduino IDE İncelemesi; Arduino entegre geliştirme ortamı (IDE), İşleme ve Kablolama arasında bir çapraz platform uygulamasıdır. Yazılım, Arduino donanımı ile çalışır. Usta veya yazılım geliştirmeye aşina olmayan yeni gelen çapraz platformlara programlamayı tanıtır [28].

Arduino IDE'nin ana faydaları, şirket içi bir uygulama ve çevrimiçi bir düzenleyici, doğrudan çizim, kart modülü seçenekleri ve entegre kitaplıklar olarak işlev görme becerisinde görülebilir. Özellikle, kullanıcıların sistemin kullanımındaki avantajları şunlardır:

Kart Modülü Seçenekleri; araç olarak kullanıcıların kullanmak istedikleri kartı seçebilecekleri bir kart yönetim modülü ile donatılmıştır. Başka bir panele ihtiyaç duyulursa, açılır menüden sorunsuz bir şekilde başka bir seçenek seçebilirler. PORT verileri, panoda her değişiklik yapıldığında veya yeni bir pano seçildiğinde otomatik olarak güncellenir.

Doğrudan Eskiz; Arduino IDE, kullanıcıların kendi metin düzenleyicisinin içinden eskizler oluşturmasına olanak tanır. İşlem basit ve anlaşılırdır. Dahası, metin düzenleyicinin daha etkileşimli bir deneyimi teşvik eden ek özellikleri vardır [70].

Python ve C++ IDE NetBeans; NetBeans (Apache); Windows, Linux, Solaris ve Mac için açık kaynaklı ve ödüllü bir IDE (entegre geliştirme ortamı) uygulamasıdır. NetBeans IDE, programcıların Java tabanlı web uygulamalarını, mobil uygulamaları ve masaüstlerini hızla geliştirmelerine olanak tanıyan güçlü bir Java uygulama çerçevesi platformu sağlar. C / C ++ programlama için en iyi IDE'lerden biridir ve ayrıca PHP programcıları için hayati araçlar sağlar [71].

IDE; PHP, C / C ++, XML, HTML, Groovy, Grails, Ajax, Javadoc, JavaFX ve JSP, Ruby ve Ruby on Rails gibi birçok dili destekleyen ilk düzenleyicidir.

Düzenleyici, zengin özelliklere sahiptir ve çok çeşitli araçlar, şablonlar ve örnekler sunar. Topluluk tarafından geliştirilen eklentiler kullanılarak oldukça genişletilebilirken; bu da onu yazılım geliştirmeye çok daha uygun hale getirir [72].

Netbeans IDE; aşağıda belirtilen özelliklerle beraber uygulama geliştirilmesini yepyeni bir düzeye taşır. Bu düzeyler aşağıdaki gibidir:

• Hızlı kullanıcı arayüzü geliştirme için sürükle ve bırak GUI tasarım aracı.

• Kod şablonları ve yeniden düzenleme araçlarıyla zengin özelliklere sahip bir kod düzenleyici.

• GIT ve mercurial gibi entegrasyon araçları.

• En son Java teknolojilerini destekleme.

• Zengin bir topluluk eklentileri seti.

Dışsal Girdi verileri ve hazırlık; Eylem algılamada en önemli parametre, video karesinin boyutudur. Çerçevenin boyutu önemlidir çünkü çerçeve ne kadar küçükse, doğruluk o kadar yüksek olur. Dolayısıyla bu şekilde nesneleri tespit etmek kolaylaşır. Çerçevenin boyutu artarsa, videonun kalitesi düşer ve bulanıklaşır. Bulanık videoda, nesneleri yüksek doğrulukla tespit etmek oldukça zordur. 300x300 piksel video çerçeve boyutlarını kullanarak üç farklı IoT cihazı ile

deneyler yapılmıştır. Deneysel analiz yukarıdaki üç parametre dikkate alınarak yapılmıştır. TensorFlow ve TensorFlow Lite kitaplıklarını kullanarak bir derin öğrenme algoritmasına dayalı eylem tespiti gerçekleştirilmiştir.

Bu çalışma ile ilgi alanına giren görüntüleri ve videoları toplamak için ücretsiz ve halka açık çevrimiçi kaynaklar kullanılmıştır. Ayrıca test modeli gereksinimlerini karşılamak için iç mekan görüntüleri ve videolar 3 metre mesafeden toplanmıştır. Toplanan veriler kişi ve araç resimlerini içermektedir. Video çerçeve boyutu, IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge 2.3'te gösterildiği gibi bağlı kameranın desteklenen aralığı içinde herhangi bir boyuta ayarlanabilir niteliktedir. IoT Raspberry Pi kurulumu için video çerçeve boyutu 300x300 piksel olarak seçilmiş ve ESP32-CAM'de olduğu gibi video canlı akışı için 320x240 çerçeve boyutu seçilmiştir.

Veri Kümesi Tasarımı; Girdi verileri, veri toplamada uyarıcı olarak kullanılan görüntüler için bir takım gereklilikler vardır. Kolay değerlendirme için görüntülerin bazı teknik gereksinimleri karşılaması gerekir. Bu nedenle, gerekli veri seti, görseller üzerinde, tasvir edilen nesneleri ve görüntüdeki konumlarını listeleyen açıklamalar içermelidir. Bu konum bilgisi sınırlayıcı kutular şeklinde ya da segmentasyon maskeleri şeklinde olabilir. Nesnelerin sabit konumlarının karmaşık olmayan bir şekilde karşılaştırılmasına olanak sağlamak için, görüntülerin ideal olarak aynı boyutta veya en azından aynı en-boy oranına sahip olması gerekir. Resimler çok küçük olmamalıdır. Resim boyutları ortalama olarak en az 400 piksel olmalıdır.

Bu ilk gereksinimler, açıklamalı görüntüleri ücretsiz olarak kullanılabilen birkaç veri kümesi seçeneği bırakmıştır. Kalan seçeneklerden, görüntülerin içeriği göreve en uygun olduğu düşünüldüğü için Microsoft'un COCO (bağlamdaki ortak nesneler) veri kümesi [35] seçilmiş ve kullanılmıştır.

COCO veri kümesi, yalnızca nispeten düz bir arka planda ortalanmış tek bir nesneyi gösteren çok sayıda görüntü içeren ImageNet gibi diğer görüntü veri kümelerinin aksine, nesneleri doğal ortamlarda göstermek için toplanmıştır. Böylece sözde fotoğrafçı önyargısından kaçınmanın ve zengin bağlamla doğal sahneleri göstermenin, öğrenme algoritmalarının, belirli bir nesnenin nerede bulunabileceği ve diğer nesnelerin genellikle yakınlarda görünmesi gibi nesneler hakkında bağlamsal bilgi toplanmasına olanak sağlayacaktır. Ayrıca, öğrenme algoritmalarının sadece bir görüntünün merkezine doğru odaklanmasını engellemelidir.

2014'te; piyasaya sürülen COCO'nun doğrulama setinden biri, modeli eğitmek ve diğeri ise oluşturulan belirginlik haritalarının faydasını değerlendirmek için iki alt görüntü alt kümesi alınmıştır. Sabitleme verileri yalnızca eğitim görüntü seti için kaydedilmiştir.

Çizelge 2.4'de alt kümeler hakkında istatistiksel bilgiler verilmiştir. İki veri kümesi tamamen ayrıktır. Eğitim setinin, görüntüleri daha çeşitli hale getirmek ve sabitleme veri setinin daha sonra genişletilmesine izin vermek için istatistiklere dâhil edilmeyen iki ek nesne kategorisi içerdiğine dikkat etmek gereklidir. Bu durum, mevcut değerlerin hesaplamasını dikkate almak için değerlendirme kümesine kıyasla görüntü başına ortalama hedef sayısını ve farklı hedef kategorilerini azaltır [73].

Çizelge 2.4 alt kümeler hakkında istatistiksel bilgiler [31] [78].

	Training Set	**Evaluation Set**
Ortalama hedef örnek sayısı	0.938	**1.049**
Ortalama farklı nesne kategorileri sayısı	0.743	**1.086**
Sunum için ortalama ölçekleme faktörü	**1.687**	**1.687**

(a) Görüntü istatistikleri

	ESP32-CAM	**RPi3**	**RPi4**
Ortalama sınırlayıcı kutu boyutu	5.1 %	22.27 %	52.96 %
Minimum sınırlayıcı kutu boyutu	0.007 %	0.1 %	1.03 %
Maksimum sınırlayıcı kutu boyutu	85.1 %	93.83 %	100 %
Ortalama bölümleme alanı	4.02 %	14 %	32.43 %
Görüntü başına ortalama örnek sayısı	0.275	0.408	0.255
Hedefi içeren görüntü sayısı	94	110	93

(b) Eğitim setinin hedef istatistikleri

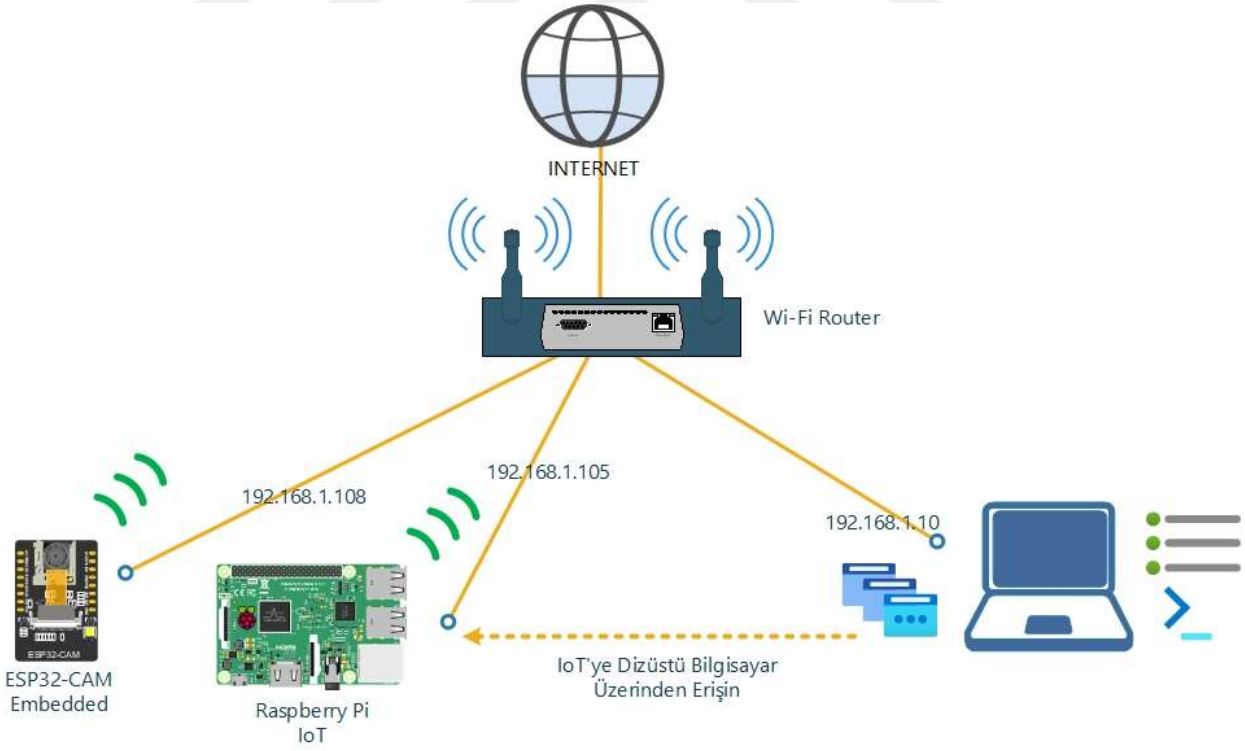
	ESP32-CAM	**RPi3**	**RPi4**
Ortalama sınırlayıcı kutu boyutu	4.17 %	22.65 %	50.98 %
Minimum sınırlayıcı kutu boyutu	0.012 %	0.05 %	0.2 %
Maksimum sınırlayıcı kutu boyutu	98.04 %	99.48 %	100 %
Ortalama bölümleme alanı	3.18 %	14.39 %	32.06 %
Görüntü başına ortalama örnek sayısı	0.345	0.391	0.313
Hedefi içeren görüntü sayısı	229	243	236

(c) Değerlendirme setinin hedef istatistikleri

2.3.2. Çözüm Tasarımı

2.3.2.1. Ağ tasarımı

Raspberry Pi ve ESP32-CAM'e bağlanmak ve üzerinde çalışmak, hem Raspberry Pi hem de ESP32-CAM'den yerel olarak hata ayıklamak için, doğrudan İnternet bağlantısı olan bir ağ yapısına ihtiyaç duyulur. Sonuç olarak, dizüstü bilgisayardan Raspberry Pi ve ESP32-CAM'e bağlanırken her iki cihazda da İnternet bağlantısına sahip olan yerel bir ağa ihtiyaç vardır. Ayrıca, Raspberry Pi'den son kullanıcı uygulamasına mesaj gönderecek bir sunucu kurulmalıdır. Bu tez çalışmasında, bir IoT gerçek zamanlı sistem olarak önerilebilecek gerçek bir senaryo için eylem tespitini gözden geçirmeyi amaçladığından, bir sağlayıcıdan sunucu satın almak bir çözüm yolu değildir. Bu nedenle, IoT cihazlarımızın bir istemci gibi davranması gerekecek ve sunucu bir Google SMTP sunucusu olacak; aynı zamanda, MQTT mesajları için MQTT aracısı olarak Thingspeak kullanılmıştır. Raspberry Pi'ye dışarıdan erişim, yerel ağın yönlendiricisindeki port yönlendirme ile sağlanmıştır. Aşağıdaki Şekil 2.11'da projenin hem IoT hem de Gömülü cihazlar için ağ kurulumu Şekil 2.11'de gösterilmiştir.



Şekil 2.11 IoT Ağ tasarımı

2.3.2.2. Önerilen Çözüm Tasarımı:

Önerilen tasarım, Şekil 2.12'te gösterildiği gibi bir derin öğrenme algoritması kullanarak eylem algılama tekniği sınırlamalarını gözden geçirmektedir.

Teknik açıdan değerlendirildiğinde, yaygın olarak bulunabilen Raspberry Pi (iki model RPi 3B, RPi 4B için) ve ESP32-CAM olmak üzere iki farklı düşük güçlü ve düşük maliyetli IoT cihazı vardır. Her iki cihaz da farklı konfigürasyonlarda bulunabilir durumdadır. Bu çalışmada kullanılan IoT cihazların özellikleri IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge **2.3** IoT Cihazların özelikleri [72] [76] [77].

Ayrıca Intel MovidiusTM Neural Compute Stick (NCS) ve Google Coral Edge TPU çipinin IoT cihazlarına eklenmesi aşağıda açıklanmaktadır.

NCS, derin öğrenme ağlarını çalıştırma, video kodlama / kod çözme ve görüntü işleme görevlerini gerçekleştirme konusunda uzmanlaşmış, özel bir MyriadTM 2 Vision İşleme Ünitesi (VPU) içeren düşük güçlü bir USB bellektir. İki DNN çerçevesini (TensorFlow ve Caffe) destekler.

Coral Edge TPU, düşük güçlü cihazlar için yüksek performanslı makine öğrenimi çıkarımı sağlayan, Google tarafından tasarlanmış küçük bir ASIC'dir. Tek bir Edge TPU, yalnızca 2 Watt güç kullanarak saniyede 4 trilyon (sabit nokta) işlem (4 TOPS) gerçekleştirebilir. Başka bir deyişle, watt başına 2 TOPS elde edilebilir ve yalnızca TensorFlow Lite'ı destekler.

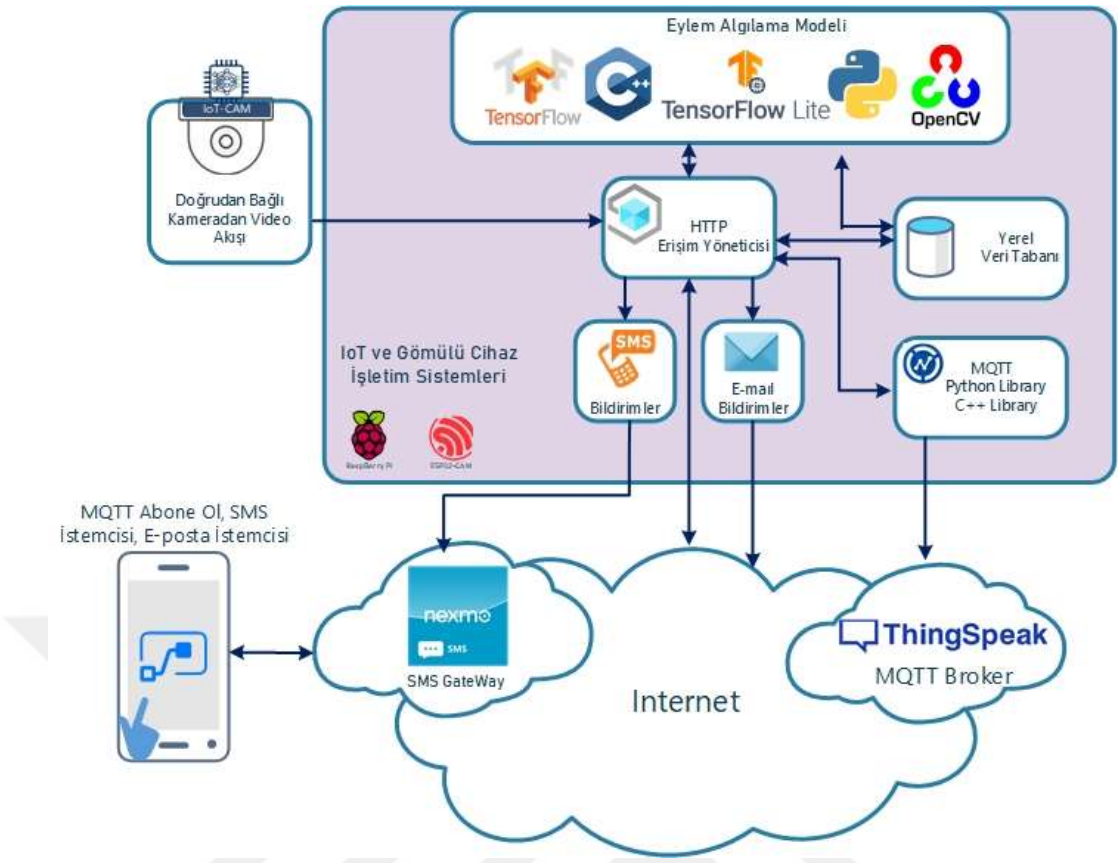
Bu tez çalışmasında aynı zamanda IoT cihazının bulut hizmetlerine bağlanmadan yerel olarak tahminler yapabildiği alternatif bir paradigma önerilmektedir. IoT bulut hizmetleri ikincil bir seçenek olarak sunulabilir. Dolayısıyla bu süreç, geleneksel paradigmanın ötesinde birçok senaryoyu mümkün kılmaktadır. Gecikme, bant genişliği, gizlilik ve enerji endişeleri nedeniyle verileri buluta aktarmak mümkün olmamaktadır. Örneğin, ormana yerleştirilmiş bir mikrodenetleyiciyi, itfaiye istasyonundan yardım istemek için yaklaşan yangın konusunda uyaran, sahadaysa söndürmeye başlayan sistemler için, yerel olarak tahminlerde bulunmanın önemli olduğu birçok senaryo oluşturulabilir. Yerel olarak tahminlerde bulunmak, cihazın bulut bağlantısından bağımsız olarak her yerde çalışmasına izin verir. Ayrıca, yerel tahminlerle uyarılar, tüm sensör okumalarının öncelikle buluta iletilmesini gerektirdiğine kıyasla daha hızlı bir şekilde yükseltilebilir. Bir talimatı yürütmek için gereken enerji, bir bayt iletmek için gereken enerjiden çok daha düşük olabileceğinden, yerel olarak tahminlerde bulunmak pil ömrünü önemli ölçüde uzatır. Böylece tekrarlanan orman yangın felaketlerinden kaçınılır. Dolayısıyla gelecekte oluşabilen yangın riski önlenebilir hale getirilir. Son olarak, insanlar bu tür hassas verileri,

özellikle de gerçek zamanlı video akışından gelen verileri bazı özel alanlarda buluta iletmeye istekli olmayabilir. Bu özellikler, gözetim sistemlerindeki implantlar, sağlık hizmetleri sistemleri, bağlantısız çiftliklerde hassas tarım, görme engelliler için akıllı gözlükler vb. dâhil olmak üzere birçok farklı senaryolarla arttırılabilir özelliktedir [74].

Önerilen Çözümün Tasarımı Şekil 2.12'de gösterilmiştir.

Şekil 2.12 Önerilen Çözümün Tasarımı

IoT cihaz işletim sistemi tasarımın içinde nasıl çalışır; IoT Cihazları, Şekil 2.13'de de gösterildiği gibi çeşitli hizmetlerin düzenlenmesinde kullanılmaktadır. Sistem, doğrudan bağlı bir kameradan yerel Gerçek Zamanlı Akış videosunu destekler. Örneğin, güvenlik kamerası akışları Motion RBG akışlarına dönüştürülür. Kodlamadan sonra görüntüler işlenmeye hazırdır. Gömülü cihaz, intranet'e bağlı farklı IoT cihazları için güvenli aktarım sağlayarak Message Queuing Telemetri Aktarımı (MQTT) protokol aracısı olarak da görev yapar. Ayrıca, bulut ağında abone görevi görür.



Şekil 2.13 IoT cihazı işletim sistemi süreci

Veri toplama hizmetlerine ek olarak, gömülü cihaz, eylem algılama olaylarını, video sensörü girişlerini ve röle çıkışlarını kaydetmek için kullanılan yerel bir veritabanı içerir. İnternet bağlantısı varken eylem algılaması gerçekleştirildiğinde, bu olayları SMS yoluyla veya e-posta ile kullanıcılara bildirmek için tetikleyici uygulanır. Bu olaylar 10 saniye MPEG-4 videoları içerir ve mesajlaşma kodu bloğu kullanılarak etkinleştirilebilir veya devre dışı bırakılabilir özelliktedir. Son olarak, kameralar, Amazon Web Services (Amazon.com, Inc., Seattle, ABD), gibi çevrimiçi bulut platformlarında barındırılan harici bir MQTT aracısı kullanılarak herhangi bir yerden uzaktan izlenebilir / kontrol edilebilir düzeydedir [69]. Donanım ve Yazılım uygulama adımı bitirildikten sonra model eğitimi için bir sonraki faza geçilmiştir.

2.3.3. Giriş Verileri

Veri Kümesi Ortamı; Bu alt bölümde, kodlamada kullanılan veri seti açıklanmaktadır. Deneysel ihtiyaçlara yönelik olan veri seti, MS-COCO, kıyaslama paketi ve simülasyon teknolojileri kullanılmıştır.

Genel kayıp işlevi şu şekilde tanımlanır; Derin Öğrenmeyi kullanarak Eylem Algılamayı çalıştırmak için, gerçek zamanlı olarak tahmin edilen davranışın doğruluğu ile en zorlu görevlerden biri olan ilgili veri setine ihtiyaç duyulur. Önceden tanımlanmış bir sınırlayıcı kutu (önceki), IoU oranına göre kesin referans nesneleriyle eşleştirilir. Özellik haritasının her bir öğesi, kendisiyle ilişkilendirilmiş bir dizi varsayılan kutuya sahiptir. Temel doğruluk kutusu olan 0,5 veya daha büyük IoU'ya sahip herhangi bir varsayılan kutu eşleşme olarak kabul edilir. Hesaplanan sınırlayıcı kutudaki bir nesnenin varlığında ağın ne kadar güvenli olduğunu ölçen güven kaybı dâhil, her kutu için SSD ağı iki kritik bileşeni hesaplar. Kategorik çapraz entropi ve konum kaybını kullanarak, ağların sınırlayıcı kutularının eğitim verilerine dayanarak gerçek olanlardan ne kadar uzakta olduğunu tahmin ettiğini hesaplar.

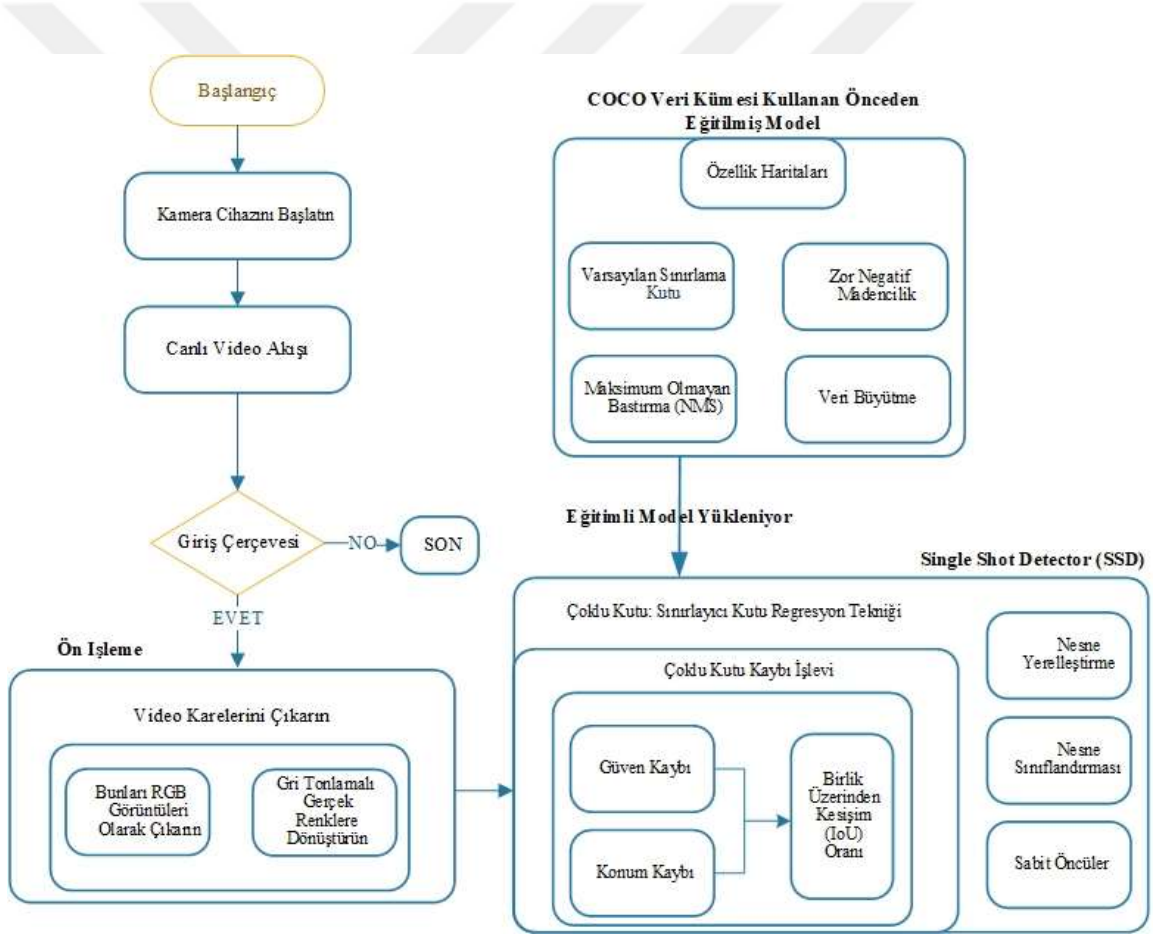
Genel kayıp işlevi formülü [79] şu şekilde tanımlanır:

$$\boldsymbol{L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right)}$$

Burada N, eşleşen varsayılan kutuların sayısıdır. 300 ve 512 girişli standart SSD'nin diğer varyantları ile MobileNet ve Inception modelleri bu araştırmada uygulanmış ve test edilmiştir.

Mutlak performansı maksimize etmeyi değil, daha çok transfer sonuçlarını iyi bilinen bir mimari üzerinde inceleme hedef alındığı unutulmamalıdır. Elde edilen sonuçların karşılaştırılabilir, genişletilebilir ve çok sayıda araştırmacı için faydalı olması için TensorFlow'un referans uygulaması kullanılmıştır [85]. Araştırma çalışması esas olarak Derin Öğrenme algoritmalarını kullanan Eylem Algılama tekniklerine odaklanmaktadır. Bu nedenle önceden eğitilmiş görüntüleri aktararak öğrenme kullanılmıştır. Ücretsiz MS-COCO veri setinde hazır bulunan nesnelerin ağırlıkları kullanılmıştır. Şekil 2.14'te, derin öğrenme algoritmasına (SSD) dayalı nesne algılamanın ayrıntılı sistem akış diyagramını temsil etmektedir [48]. Canlı video akışındaki nesneleri sınıflandırmak için COCO veri setine dayalı önceden eğitilmiş bir model kullanılmıştır. SSD'de, varsayılan sınırlayıcı kutular, nesnelerin çoğunun yakalanması için farklı ölçeklerde değişiklik gösterir. Özellik haritaları, evrişimli blokların sonuçlarıdır. Özellik haritaları, nesne algılama şansını arttırır. Verileri eğitirken, sınırlayıcı kutular düşük Birlik Kesişimine (IoU) sahip olabilirler. Bu durum da olumsuz bir eğitim örneği anlamına gelir. Ayrıca Negatif örnekler

de gereklidir. Çünkü ağın öğrenilmesi gerekir. Bu süreç zor negatif madencilik olarak adlandırılır. Birçok derin öğrenme uygulamasının, ağa çeşitli boyutlardaki nesneleri algılamayı öğretmesi gerekir. Görüntü çerçeveleri farklı IoU oranlarında test edilir. Veri büyütmede, nesnelerin benzer olasılıkla solda ve sağda görünmesini sağlamak için görüntü çerçeveleri yatay olarak çevrilir. SSD'nin ileri geçişi sırasında çok sayıda sınırlayıcı kutu oluşturulur. Gürültü sınırlayıcı kutuların kaldırılması oldukça önemlidir. Bu nedenle, 0,45 IoU'dan daha az olan sınırlayıcı kutular, videoda düşük doğrulukla tespit edilen nesnelerin etiketlenmesine gerek olmadığından göz ardı edilebilir. Bu teknik, maksimum olmayan baskılama (NMS) olarak bilinir. SSD'de kullanılan sınırlayıcı kutu tekniği, Szegedy'nin Multibox'taki çalışmasından esinlenmiştir (Szegedy ve diğerleri, 2014). Şekil 2.14'te, COCO veri kümesiyle derin öğrenme algoritmasına (SSD) dayalı nesne algılamanın ayrıntılı sistem akış diyagramını temsil etmektedir.



Şekil 2.14 Nesne algılama mekanizması ayrıntılı akış şeması

Güven kaybı, ağın bir nesneyi ne kadar doğru algıladığı anlamına gelirken, konum kaybı, ağın öngörülen sınırlayıcı kutusunun eğitim setinin temel gerçeklerden ne kadar uzakta olduğunu ölçer. Her özellik haritası hücresi, farklı boyutlarda bir dizi sınırlayıcı kutu ile ilişkilendirilir. Bu

öncelikler, temel gerçeğin 0,5'in üzerinde olduğu için IoU'ları nedeniyle seçilir. Bu, Sabit öncelikler olarak adlandırılır. Birleşimin kesişimi, birleşim alanına göre toplam örtüşme alanı olarak hesaplanır. IoU 0,5'ten büyükse nesne algılama mükemmel kabul edilir. Çoklu kutu tekniği, konum ve güven kayıplarını en aza indirmiştir. Tek seferde, nesne yerelleştirme ve sınıflandırma, ağın tek bir ileri geçişinde yapılır.

2.4. Faz II: Donanım Ve Yazılım Uygulama, CNN'nin Eğitilmesi, Modelin Test Edilmesi

2.4.1. Donanım Uygulama

2.4.1.1. IoT Donanımları Uygulama

Bu tez çalışmasında Windows ve Linux işletim sistemiyle çalışan iki bilgisayar kullanılmıştır. Windows dizüstü bilgisayarda IoT sanal ortamı oluşturmak için kullanılmış ve Linux dizüstü bilgisayar ise IoT cihazlarına bağlanma ve kod uygulaması için kullanılmıştır. Çizelge 2.5 Çalışmada kullanılan bilgisayarların özellikleri verilmiştir.

Çizelge 2.5 Çalışmada kullanılan bilgisayarların özellikleri

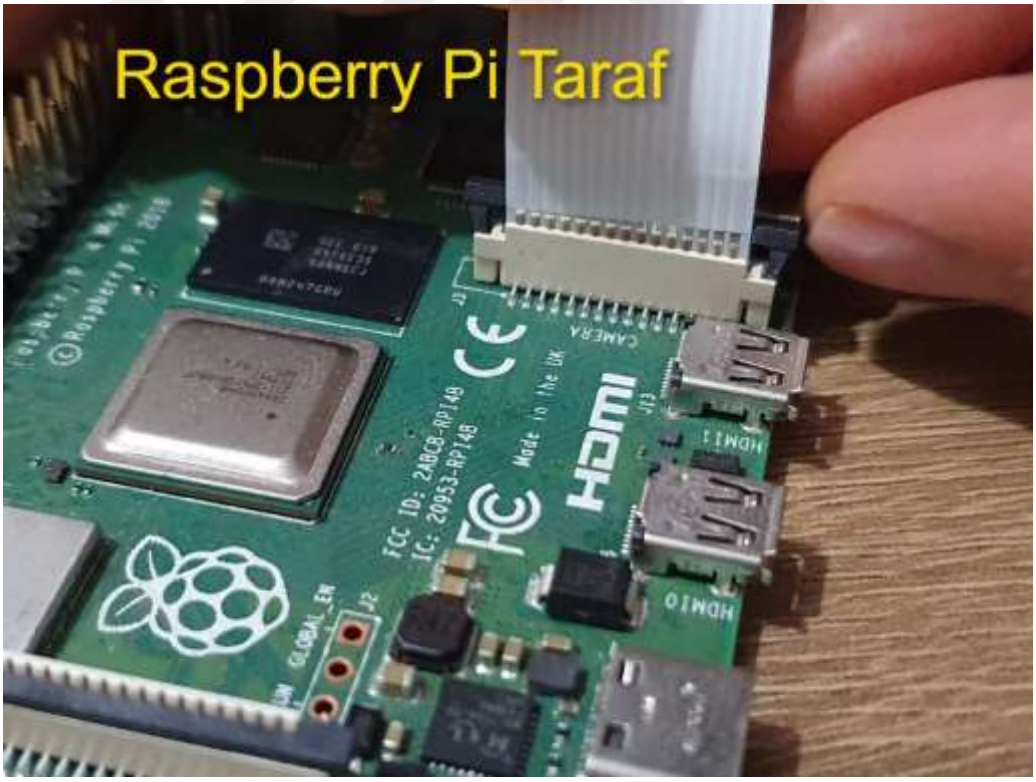
Parçalar	**Dizüstü 01**	**Dizüstü 02**
OS	Linux Ubuntu 20 LTS	Window 10
CPU	i7	i7
GPU	1GB	1GB
RAM	16GB	12GB
HDD	500 GB SSD	128 GB SSD + 1TB HDD

2.4.1.2. IoT cihazı Raspberry Pi 3B ve Pi 4B'nin kurulması

Boarda Güç Verilmesi; Raspberry Pi 3B ve Pi 4B'nin 2A ile 3.5A sağlayabilen bir 5V mikro USB güç kaynağı kullanılarak çalıştırılmasını önerirken, karşılaştırma testleri veya ağır bir iş yükü çalıştırılması durumunda 4A güç kaynağı kullanılmasını önermektedir [68] [75]. Buradan USB güç kaynağıyla düşük gerilim sorunları yaşanması durumunda, normal bir DC kaynağı kullanılarak cihazın çalıştırılabileceği anlaşılmaktadır.

Raspberry Pi Kamera V2.1'i Kurma; Bu adımda, Raspberry Pi Camera V2.1'in Rasberry Pi kartına eklenmesi işlemi yapılmıştır. Bu işlemler Şekil 2.15 ile Şekil 2.16'de gösterilmiştir.

Şekil 2.15 Pi Kamera Kurulumu Kamera Tarafından Gösterimi



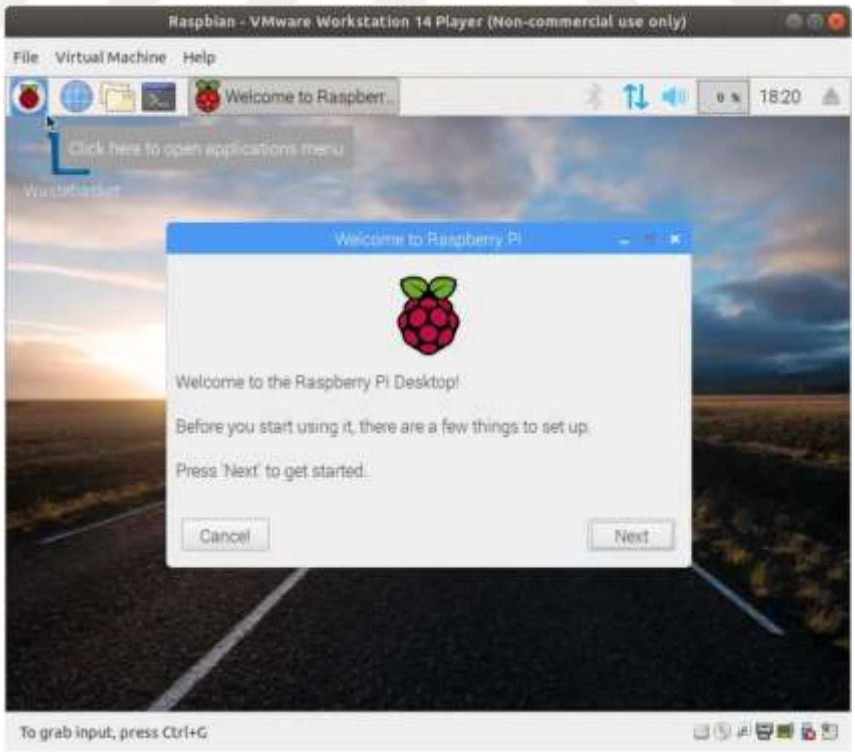
Şekil 2.16 Pi Kamera Kurulumu RPi 3 Tarafından Gösterimi

İşletim sistemi görüntüsünün indirilmesi; Raspberry Pi için IoT cihazı olarak seçilen işletim sistemi IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge 2.3'de belirtildiği gibi Raspberry Pi OS'dir. Sürüm oluşturma için sistem, Raspberry Pi OS sürüm 5.4 olan mevcut yeni sürüm kullanılmıştır. Kullanılan tüm teknikler, Raspberry Pi OS bağlantı noktası desteklidir. En son Raspberry Pi OS Installer ISO'yu indirilmiş ve iş istasyonu ana bilgisayarının yerel sabit diskine kaydedilmiştir. Önerilen işletim sistemleri için resmi görüntüler Raspberry Pi web sitesi indirme sayfasından indirilebilir. SD Karta İşletim Kartının Yazılması, kullanılan işletim sistemine bağlıdır.

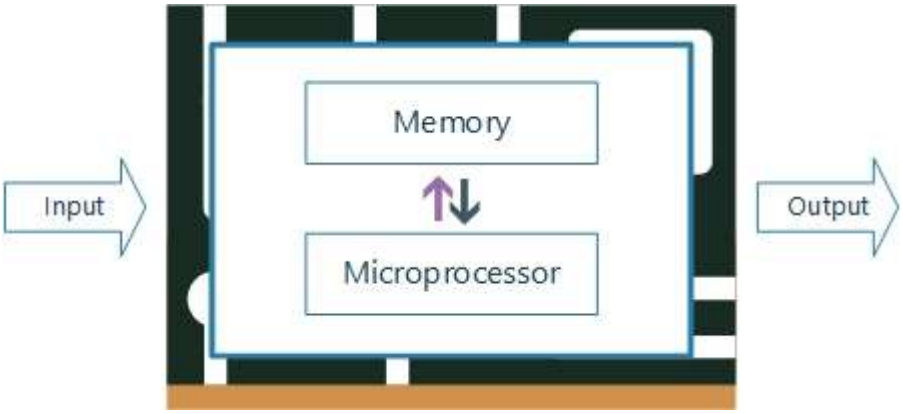
Yeni IoT işletim sistemini başlatılmasın ve Donanımın Test Edilmesi; hazırlanan SD kart Raspberry Pi'ye takılarak sistem çalışır duruma getirilmiştir. Tüm donanımın iyi çalıştığı kontrol edilmiş olup, kamera gerçek zamanlı video akışı için test edilmiştir. Sistem yükseltmesinin de yapıldığı Raspberry Pi'nin son masaüstü hali Şekil 2.17'de gösterilmiştir.

IoT Cihazınının Ağa Bağlanması, Şekil 2.11'de gösterildiği gibi, yerel yönlendiricideki yerel DHCP Havuzundan IP'ye sahip olan hem Ethernet kablosu hem de Wi-Fi bağlantısı ile ağa bağlanılabilir.



Şekil 2.17 Raspberry Masaüstü

Gömülü ESP32-CAM cihaz uygulaması; ESP32-CAM gömülü aygıt kabloları, Şekil 2.18'de gösterildiği gibi pimlere takılmıştır ve açılmaya hazır duruma getirilmiştir.



Şekil 2.18 ESP32-CAM Genel bir gömülü sistem mimarisi

OV2640 model Kameranın Takılması; IoT cihazların özelikleri Çizelge 2.3'de verilmiştir.

Çizelge 2.3'de belirtildiği gibi OV2640 model kamera, Şekil 2.19'de gösterildiği gibi ESP32-CAM Kartına takılmıştır.

Şekil 2.19 Attaching the Camera OV2640

Anakarta Güç Verilmesi ve Donanım Testi, ESPRESSIF, ESP32-CAM'in güç kaynağı için 5V spesifikasyon gereksinimi kullanılarak çalıştırılmasını önermektedir [76, 77].

Bu uygulama, USB güç kaynağınızla düşük gerilim sorunları yaşıyorsanız, onu 'normal' bir DC kaynağı kullanarak çalıştırabileceğiniz anlamına gelir. Şekil 2.20'de (a) ve (b) gösterildiği gibi bir dizüstü bilgisayar ve 1000 mA güç bankası kullanıyoruz. Bununla beraber normal bir DC kaynağı kullanarak cihaz çalıştırılabilir.

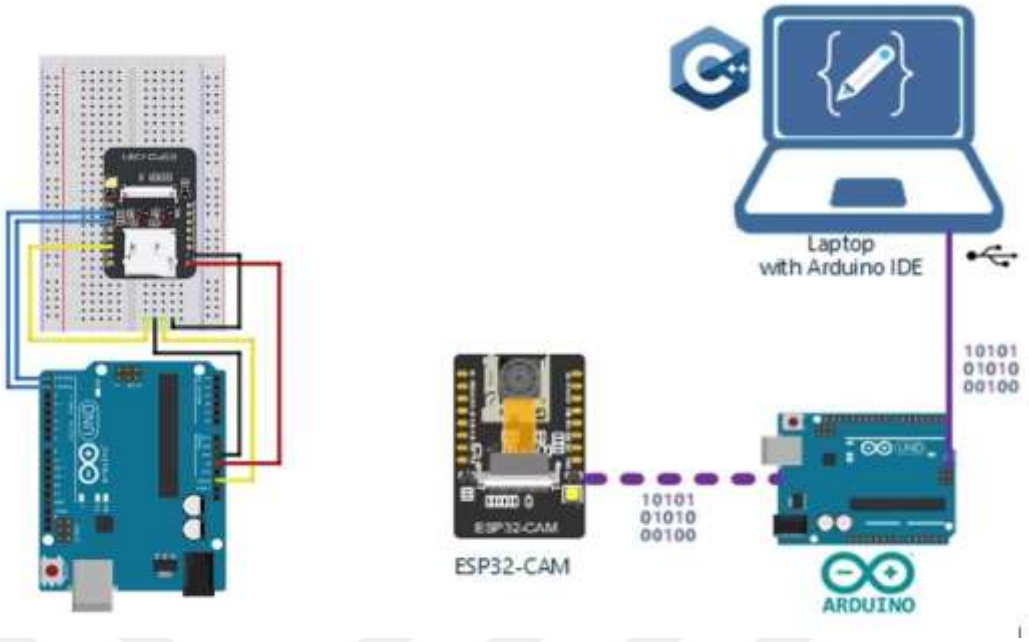
(a)

(b)

Şekil 2.20 (a) Raspberry Pi 3 Model B **(b)** Raspberry Pi 4 Model B

Arduino IDE'nin ESP32 ile kullanılması durumunda, kameranın test edilmesi işlemi kurulan örnek taslak kullanılarak yapılır.

Seri konsol kablolarının boarda bağlanması; ESP32-CAM, Şekil 2.21'de gösterildiği gibi Arduino Uno cihazı ve Arduino IDE programı tarafından seri arayüz kullanılarak PC'ye bağlanmıştır.



Şekil 2.21 Arduino UNO ile ESP32-CAM seri kablo bağlantısı ve bağlantı şeması

2.4.2. Yazılım Uygulama

2.4.2.1. IoT için Sanal ortam oluşturulması

Kurulum, Windows işletim sistemi üzerinde gerçekleştirilmiştir. VMware iş istasyonu 15, orijinal Raspberry Pi IoT cihazını taklit etmek için sanal bir IoT cihazı olarak seçilmiştir. Kodların gerçek IoT üzerinde uygulamaya hazır hale getirilene kadar testin tekrar tekrar çalıştırılabilmesi, orijinal görüntünün olabildiğince hızlı yedekleme ve geri yükleme yeteneği ile kodların uygulanması ve test edilmesi işlemleri, oluşturulan bu sanal ortamda gerçekleştirilmiştir. Sanal makineyi kurma aşamaları aşağıda verilmiştir:

- Sanal makine oluşturma adımları
- En son Raspberry Pi OS Installer ISO'yu indirdik ve İş İstasyonu ana bilgisayarının yerel sabit diskine kaydettik.
- "Yeni Sanal Makine Oluştur" u seçin
- "ISO görüntüsünü kullan" ı seçin ve Raspbian ISO dosyanıza göz atın
- Uyumluluk: Workstation 15 ve üzeri
- Konuk İşletim Sistemi Ailesi: Linux
- Konuk İşletim Sistemi Sürümü: Debian GNU / Linux 10 (64-Bit)
- CPU: 1 (veya istenen değer)
- MEM: 1GB (veya istenen değer)

- Sabit Disk 1: 10 GB
- Son olarak, Bitir ve Kapat'a tıklayın
- Sanal Makine, sihirbaz bittikten hemen sonra başlayacaktır
- Üst çubukta Başlat'a tıklayın
- İlk menüden Kur'u seçin
- Klavye düzeninizi seçin
- Bölüm diskleri için varsayılan seçenekleri koruyabilirsiniz (tüm diski kullan> tek bölümdeki tüm dosyalar> bitir> evet)
- Kurulum başlar
- Birkaç dakika sonra kurulum sihirbazı devam eder
- Ana önyükleme kaydına önyükleyici kurulumunu onaylayın
- / dev / sda'yı seçin
- Yeni işletim sisteminizi yeniden başlatmaya devam edin
- İsterseniz VMWare Araçlarını indirip kurmayı seçebilirsiniz (zorunlu değildir)
- İlk açılışta bir sihirbaz başlayacaktır
- Size en uygun ayarları seçin
- Varsayılan şifreyi değiştirin
- Sisteminizi güncelleyin ve son bir kez yeniden başlatın
- Sanal IoT cihazına bağlanma
- Workstation 15'te sanal makinenin konsolu ile local PC'den bağlanılır

Son adımda, Eylem Algılama teknikleri kod gereksinimlerini karşılamak için IoT sanal ortamında Eylem Algılama teknikleri çalıştırılmıştır. Tüm yazılım gereksinimleri uygulandıktan sonra, bir sonraki alt bölümde gerçek IoT cihazına donanım ve yazılımlar kurulacaktır.

2.4.2.2. Arduino IDE kurulumu

Arduino IDE'nin en son sürümü Arduino web sitesinden indirilmiştir. Komut dosyası doğru bir şekilde yürütülerek, çıktılar oluşturulur.

IDE'yi başlatmak için, Arduino kartı bilgisayara bir USB kablosuyla bağlanılmıştır.

Arduino IDE, (Unity menüsünde) mevcuttur. Eğer mevcut değilse, Arduino çalıştırılarak komut satırından başlatılabilir.

2.4.2.3. Python ve C ++ IDE'yi Kurun (BU KURULUM AŞAMALARININ HEPSİ EKTE GÖSTERİLEBİLİR)

Çalışmada NetBeans kullanılmıştır. NetBeans IDE 12'nin en son kararlı sürümünü kurmak için önce, Java JDK' aşağıda gösterildiği gibi kurulur.

Adımlar takip edilerek kurulum tamamlanır. NetBeans IDE'yi başlatmak için makine yeniden başlatılır. İşlemler bittikten sonra artık proje kodu ve Eylem algılama algoritmaları oluşturabilir [80][81].

2.4.2.4. Python Yükleme

Python 3.7, temel olarak IoT cihazları için uygun olacağı için seçilmiştir. Anlaşılması ve kodlanması kolay bir üst düzey programlama dilidir. Makine öğrenimi ve Derin Öğrenme algoritmalarını geliştirmek için en çok kullanılan dildir. Burada, ana bilgisayar IoT cihazının kendisi olduğundan apt ve pip3 aracını kullanarak Linux sistemine sahip olduğundan, Linux işletim sistemi ortamına Python'u kurma adımları aşağıda belirtilmiştir [18] [81].

Python 3.7'yi yüklemek için apt'yi kullanma; Raspberry Pi OS olan Debian için apt paket yöneticisi kullanılmıştır. Bazı Python paketleri Raspberry Pi OS arşivlerinde bulunabilir ve apt kullanılarak kurulabilir. Raspberry Pi işletim sistemindeki Python 2.x ile uyumlu Python paketleri her zaman bir python- önekine sahip olacaktır. Böylece, Python 2.x için picamera paketi python-picamera [81] olarak adlandırılır. Python 3 paketleri her zaman bir python3 ön ekine sahiptir. Bu nedenle, Python 3'de picamera kurmak için aşağıdaki komutlar kullanılmıştır. Bu aşamada Python çalışır duruma getirilmiş olur.

2.4.2.5. Derin Öğrenme Çerçevesi (Framework) Kurulumu

Bu adımda, 32 bit işletim sistemi ile Raspberry Pi üzerine TensorFlow çerçeve kitaplığı kurulmaktadır. Biri Python 3 ve bir C ++ API kitaplığı olmak üzere iki kurulum yapılır. Bu kurulum Raspberry Pi 3 ve Pi 4 için çalışır. TensorFlow, IoT cihazındaki yerel depolama SD-mikro kartında yaklaşık olarak 1 GB yer kaplar.

TensorFlow, derin öğrenme için özel olarak geliştirilmiş kapsamlı bir yazılım kitaplığıdır. Çok miktarda kaynak tüketir. Önceden oluşturulmuş derin öğrenme modellerini çalıştırmanın yanı sıra; Eylem Algılama kodlarında ihtiyaç duyulan modelleri de çalıştırabilir. Dondurulmuş TensorFlow modellerini TensorFlow Lite düz tampon modellerine dönüştürmek için

TensorFlow'un kitaplığı kullanılmıştır. Kodların çalıştırılacağı lite modellerin desteklenmesi için IoT cihazlarına kurulu TensorFlow Lite'a ihtiyaç duyulur.

Gömülü cihazlarda eylem algılama uygulamasında mikro denetleyici için TensorFlow Lite kullanılması uygun görülmüştür. Kullanılabilecek birçok hazır yapı modeli içerdiğinden çok daha hızlıdır ve daha az kaynak kullanır. ESP32 gibi küçük IoT Gömülü Cihazlar için tasarlanmıştır.

Önkoşullar

- Raspberry Pi Modeli: Raspberry 3/4 (3B ve 4B'de test edilmiştir)
- Raspberry Sürümü: Debian Buster (Debian 10) veya üstü
- Python Sürümü: 3.7 (Yeni bir Raspberry Görüntüsü ile Varsayılan python 3.7)

Bu aşamada TensorFlow ve TensorFlow Lite kurma işlemi tamamlanmıştır. Daha önce bahsedildiği gibi, IoT Gömülü cihazı ESP32 ile mikro denetleyici için TensorFlow Lite kodu kullanılacaktır.

2.4.2.6. Coral Yazılımının Yüklenmesi

Yazılım geliştirme kiti indirilir ve paket ekte komutlar ile ana dizine verilmiştir. Bu şekilde Coral kurulum işlemleri tamamlanmış olur.

USB Donanımsal Hızlandırıcısı, Şekil 2.22'de gösterildiği gibi, USB Coral eşlik eden kısa USB-C - USB-A kablosu kullanılarak takılmıştır. Raspberry Pi üzerinde çalışan yazılımın, Edge TPU donanımının mevcut olduğunu tanımasına izin veren bazı udev kuralları eklediği için kurulum komut dosyası test edilmiştir.

Şekil 2.22 USB Coral eşlik eden kısa USB-C - USB-A kablosu kullanılarak takılmıştır [82].

TensorFlow Lite'ta Çizelge 2.6'te hesaplanma türleri mevcuttur:

Çizelge 2.6 TensorFlow Lite'ta hesaplanma türleri [31].

Teknik	**Veri gereksinimleri**	**Boyut küçültme**	**Doğruluk**	**Desteklenen donanım**
Post-training float16 quantization	Veri yok	50%	Önemsiz doğruluk kaybı	CPU, GPU
Post-training dynamic range quantization	Veri yok	75%	Doğruluk kaybı	CPU, GPU (Android)
Post-training integer quantization	Unlabelled representative sample	75%	Daha küçük doğruluk kaybı	CPU, GPU (Android), EdgeTPU, Hexagon DSP
Quantization-aware training	Labelled training data	75%	En küçük doğruluk kaybı	CPU, GPU (Android), EdgeTPU, Hexagon DSP

Aşağıda, birkaç model üzerinde eğitim sonrası niceleme ve nicelemeye duyarlı eğitim için gecikme ve doğruluk sonuçları verilmiştir. Tüm gecikme sayıları, bir büyük çekirdekli CPU kullanan Pixel 2 cihazlarda ölçülür. Araç seti geliştikçe, buradaki rakamlar da gelişecektir. CNN Modelleri Aşamaları Çizelge 2.7'de verilmiştir.

Çizelge 2.7 CNN Modelleri Aşamaları [31]

Model	**Top-1 Accuracy (Original)**	**Top-1 Accuracy (Post Training Quantized)**	**Top-1 Accuracy (Quantization Aware Training)**	**Latency (Original) (ms)**	**Latency (Post Training Quantized) (ms)**	**Latency (Quantization Aware Training) (ms)**	**Size (Original) (MB)**	**Size (Optimized) (MB)**
Mobilenet-v1-1-224	0.709	0.657	0.70	124	112	64	16.9	4.3
Mobilenet-v2-1-224	0.719	0.637	0.709	89	98	54	14	3.6
Inception_v3	0.78	0.772	0.775	1130	845	543	95.7	23.9
Resnet_v2_101	0.770	0.768	N/A	3973	2868	N/A	178.3	44.9

2.4.2.7. Movidius Yazılımı Yüklenmesi

Neural Compute Stick'i (NCS) desteklemek için gereken yazılımı kurmak için aşağıdaki adımlar izlenir. Devam eden kurulum hakkında daha fazla günlük ve sorun giderme sağlayabilen GUI konsolu aracılığıyla kurulum yapmak için IoT Raspberry Pi.

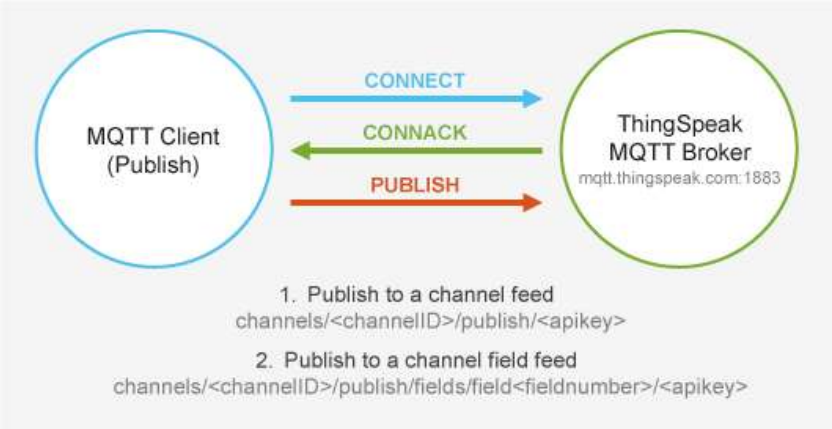
Aşağıdaki gibi çıktı alınması gerekir.

```
Hello NCS! Device opened normally.
Goodbye NCS! Device closed normally.
NCS device working.
```

Bu şekilde çubuğun tespit edildiği ve düzgün çalıştığı anlaşılır.

MQTT Broker Kanalı Uygulaması; Bu tez çalışmasında, IoT cihazı için MQTT aracısı olarak ThingSpeak'i kullanılmıştır. MQTT, HTTP / REST'ten farklıdır. Özellikle hafif olacak şekilde düşük RAM ve CPU performansına sahip gömülü aygıtlar için tasarlanmıştır. Ayrıca çoğu durumda MQTT daha az bant genişliği kullanır. Bu çalışmada istemcilerin MQTT aracısına bağlandığı ve diğer müşterilerden veri almak için veri yayınladığı veya konulara abone olduğu MQTT işlemleri kullanılmıştır.

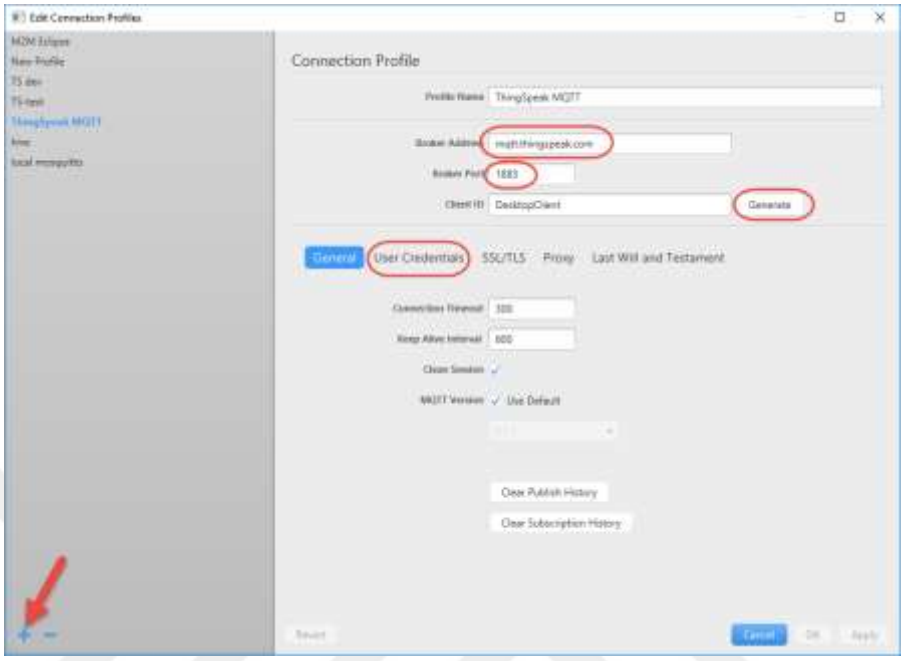
MQTT mesajları kolay ve düşük RAM, CPU ve bant genişliği gereksinimleri olan güvenliksiz TCP kullanarak gönderilmektedir. MQTT mesajları WebSockets kullanılarak da gönderilebilir. Bu, 1883 MQTT için standart bağlantı noktası ağ üzerinde engellendiğinde gerekli olabilir. WebSockets üzerinden MQTT kullanılırken, Şekil 2.23'te gösterildiği gibi iletişimi SSL ile şifrelemek mümkündür.



Şekil 2.23 MQTT mesaj yayınlama süreci [83]

Raspberry Pi'de, ThingSpeak'e veri göndermek için bir MQTT istemci kitaplığı gereklidir. Paho, bu örnekler için kullanılan açık kaynaklı bir MQTT istemci kitaplığıdır. Python dâhil birçok

dile uyarlanmıştır. Ardından, bir ThingSpeak hesabına ihtiyaç duyulur ve akabinde iki alanlı bir kanal oluşturması gerekmektedir. Veri toplamak için Şekil 2.24'de gösterildiği gibi yeni bir kanal oluşturulmuştur.



Şekil 2.24 MQTT Yeni bir Kanal Oluşturma [83]

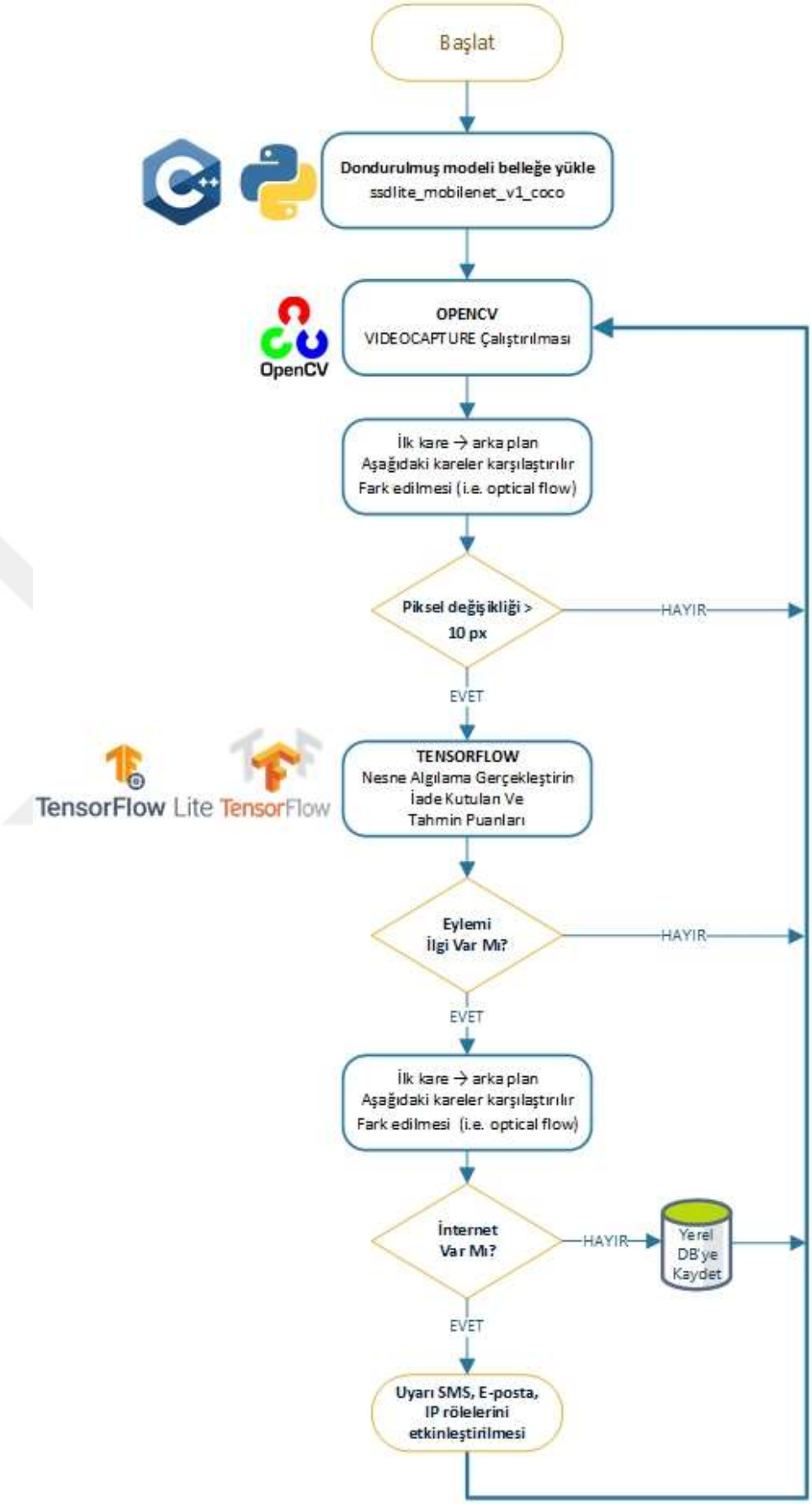
2.4.2.8. Çalışmada Kullanılan Kod

Bu bölümde, modeli IoT ve gömülü cihazlarda çalıştırmak için gereken kod bilgileri verilmiştir. Bu kod modeli, çevrimdışı ve doğrudan bir IoT cihazında çalışabilen bir derin öğrenme algoritması kullanarak eylem algılamanın nasıl oluşturulacağını ve dağıtılacağını göstermektedir. Dört ana öğe olarak Python 3.0, TensorFlow Lite ile C ++ ve TensorFlow Lite Micro kullanılmış ve derin öğrenmede kolay, düşük kodlu bir şekilde cihazlara dahil edilmiştir.

Çoğu durumda, IoT cihazını AI yetenekleriyle etkinleştirmek, verileri cihazdan bir sunucuya göndermeyi içerir [2]. Derin öğrenme hesaplamaları sunucuda gerçekleşir ve ardından sonuçlar uygun işlem için cihaza geri gönderilir. Ancak, veri güvenliği veya ağ bağlantısı söz konusu olduğunda bu ideal veya uygulanabilir bir yaklaşım değildir.

Çalışan kod TensorFlow Lite ile birleştirilmiş, Şekil 2.25'te gösterildiği gibi IoT ve Gömülü cihazlara derin öğrenme işlevselliği eklenmiştir.

Şekil 2.25'te, derin öğrenme tabanlı nesne algılamayı (SSD) çalıştıran önemli bir kod akış şemasını göstermektedir. OpenCV dahil olmak üzere farklı kitaplıklar içe aktarılmaktadır ve SSD algoritması başlatılmıştır. Kodda 20'den fazla nesne sınıfının tanımlayabildiği kolaylıkla görülebilir. Video çerçevesi girdi olarak verilmiştir ve SSD algoritması, döngü koşulu doğru olduğu sürece çerçeveleri işler. Video çerçeve boyutu herhangi bir boyuta ayarlanabilir ve bu deneylerde 300x300, 400x400 ve 800x800 piksel kullanılmıştır. Teknik detaylar sonraki bölümlerde açıklanmıştır.



Şekil 2.25 Kullanılan kod tam akış şeması

2.4.2.9. TensorFlow Modeli Oluşturma

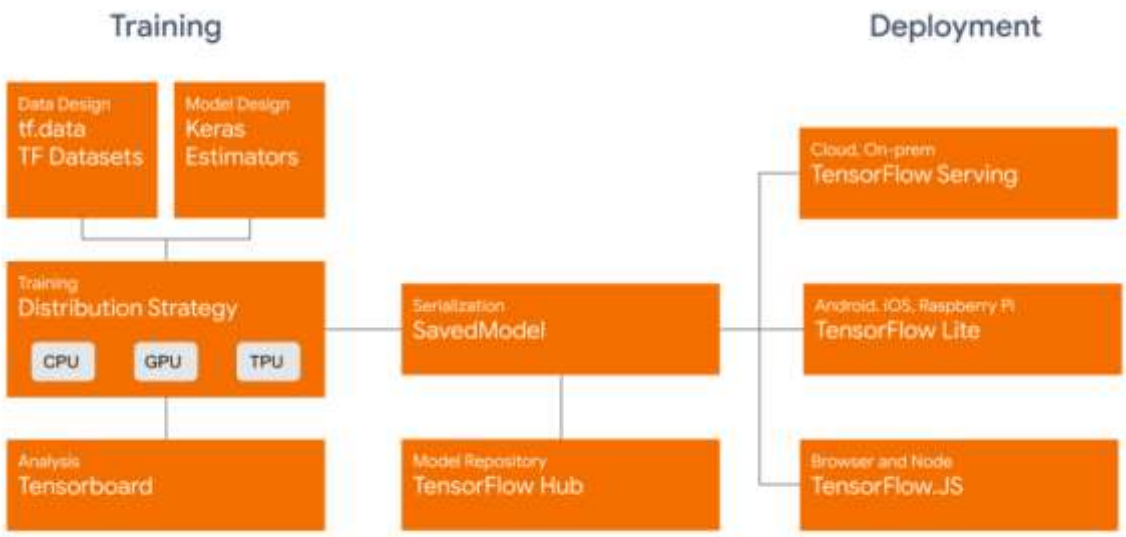
TensorFlow Modeli oluşturmak için öncelikle gerekli Modüller yüklenmiştir. Akabinde IoT Raspberry Pi'den, aşağıda gösterildiği gibi çözünürlük ve kare hızı değerlerini modele atlayarak "camera_type" işlevi aracılığıyla kamera Pi modeli başlatılır. Eğitimden sonra, modeli 'kaydet' işlevi ile bir dosyada saklanır.

Son olarak, TensorFlow API dönüştürücü kullanılarak Sinir Ağı Modeli oluşturularak ve eğitilmiş ve TensorFlow Lite ve TensorFlow Lite Micro'ya eylem algılama modelleri yüklemek için kullanılmıştır. SSD katmanlarını yığınlarken kodu bile TensorFlow model oluşturma kodundan oldukça fazla kendinden açıklamalı ve daha basittir. SSD çağrısındaki return_sequences parametresinin True olması gerekmektedir. Bu nedenle SSD hücresinin çıktısı, son yığılmış katman olmadığı sürece çıktı dizisindeki son sonuç yerine tam çıktı dizisi olacaktır.

Önceki iki yazdırma ifadesi, modeli IoT üzerinde dondurup çalıştırdığımızda ihtiyaç duyulan giriş düğümü adını (çift yönlü _1_ giriş) ve çıkış düğüm adını (aktivasyon_1 / Kimlik) yazdırmaktadır. Şekil 2.26'da derin öğrenmeye dayalı eylem algılamanın ayrıntılı kod akış şemasını temsil etmektedir.

2.4.3. Çalışma Modelinin Eğitimi

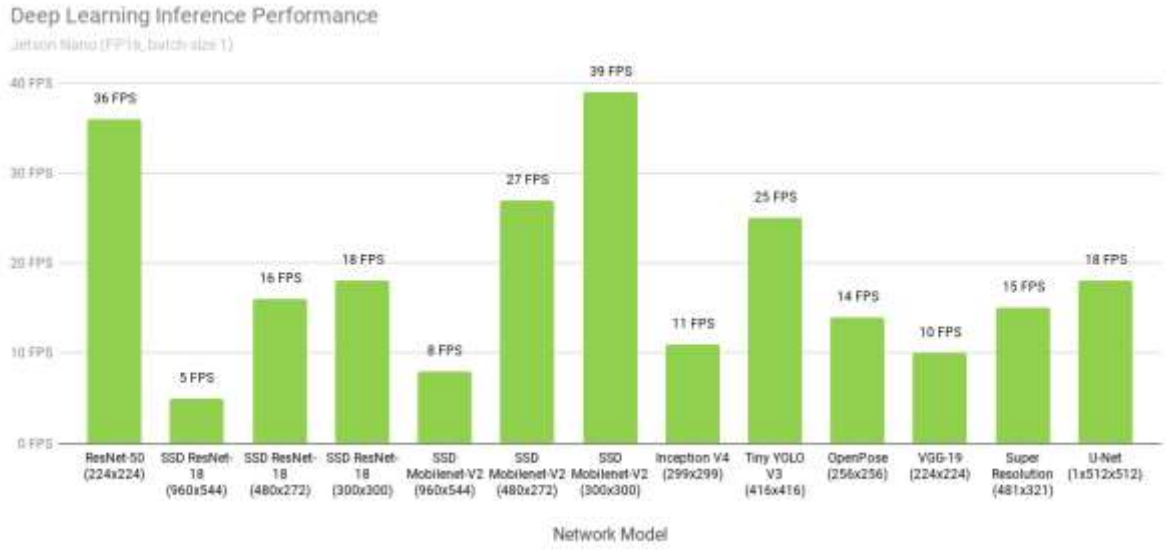
Bu çalışmanın algoritmasında, eylem algılama tekniklerini sunmak için önceden eğitilmiş bir model kullanılmıştır. TensorFlow tipi üç kitaplık kullanarak önceden eğitilmiş derin öğrenme modeli çalıştırılmıştır. Bu üç kitaplık TensorFlow, TensorFlow Lite ve TensorFlow Lite Micro kullanılarak çalışmaktadır. Şekil 2.26'de, TensorFlow Lite Modeli için eğitim aşamasından dağıtım aşamasına kadar olan tam yolu göstermektedir.



Şekil 2.26 TensorFlow Lite Modeli için eğitim aşamasından dağıtım aşamasına kadar olan tam yolu [67]

2.4.3.1. Neden önceden eğitilmiş bir model kullanılmıştır?

Bir Evrişimli Sinir Ağı'nı (CNN) eğitmek, verilere ve hedef göreve bağlı olarak çok kısa süre ya da çok uzun süre de gerektirebilir. Bu nedenle önceden eğitilmiş bir model kullanılmıştır. Önceden eğitilmiş ve aktarılmış öğrenme modellerini kullanarak, vizyonla ilgili görevleri çok daha hızlı çözen derin öğrenme uygulamaları oluşturmak mümkündür [84]. Eğitmek için gereken veri miktarının azaltılabilmesi için önceden eğitilmiş bir modele ihtiyaç vardır. Bu olmadan, algoritmamızda kullanılan modeli eğitmek için yaklaşık 100,00 görüntüye ihtiyaç vardır. Bu nedenle, sıfırdan bir derin öğrenme modeli eğitmek zaman ve çaba gerektiren bir süreç olabilir. Ayrıca TensorFlow, modeli SSD Mobilenet ile COCO veri setinde eğitmiştir ve bu model algoritmamızda kullanılabilecek en hızlı modellerden biridir. SSD Mobilenet-2'nin karşılaştırmalı değerlendirmede en yüksek FPS'ye sahip olduğunu Şekil 2.27'de gösterilmiştir [26].



Şekil 2.27 Derin Öğrenme çıkarım performansı [85]

2.4.3.2. Önceden Eğitilmiş Model Kurulumu

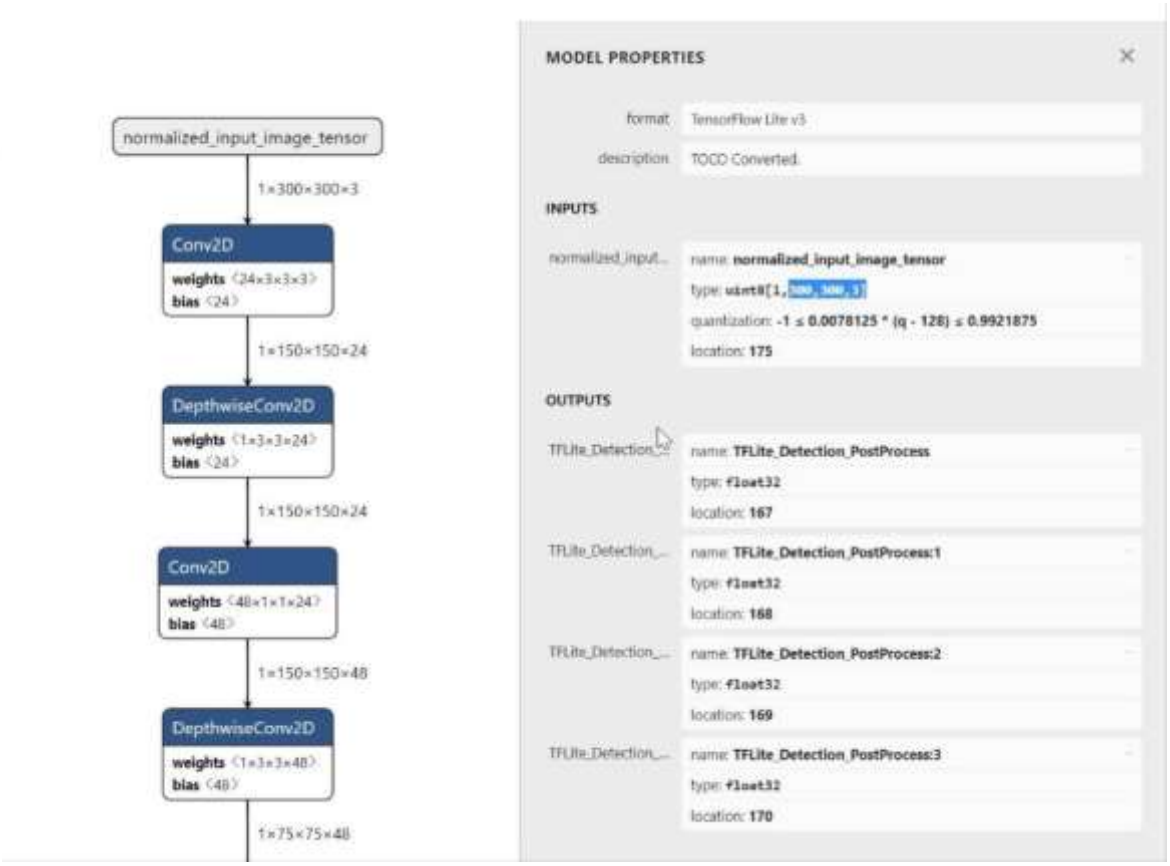
Bu bölümde, COCO veri kümesinde önceden eğitilmiş model kurulumu açıklanmaktadır. Bağlamda Ortak Nesneler (Common Objects in Context -COCO), Microsoft, Facebook ve çeşitli üniversitelerdeki araştırmacılar tarafından etiketlenmiş ve sınıflandırılmış kapsamlı bir görsel koleksiyonudur. Yaklaşık 90 nesne kategorisine sahip 200.000'den fazla resim içerir.

Nesne algılama veya nesne sınıflandırma modelleri, insanlar, arabalar, bisikletler, bardaklar, makaslar, hayvanlar gibi günlük nesneleri tanımak için bir başlangıç noktası sağlamak üzere COCO veri kümesinde eğitilebilir. Çalışmamızda nesne algılama modeli olarak COCO veri kümesinde eğitilmiş bir MobileNet V1 modeli kullanılmıştır [26].

2.4.4. Model performansı Testi

2.4.4.1. Kullanılan Modelin Sinir Ağının Görselleştirilmesi

Modeli çalıştırmadan önce, önceden eğitilmiş modelin Sinir Ağını görüntülemek için Netron kullanılmıştır. Netron, Sinir Ağları, derin öğrenme ve makine öğrenimi modelleri için görselleştirici olarak kullanılan önemli bir araçtır. Kullanılan modelin girdisi 300x300x3 bir dizidir (tensör). Şekil 2.28'te Netron aracından alınan anlık görüntü gösterilmektedir.

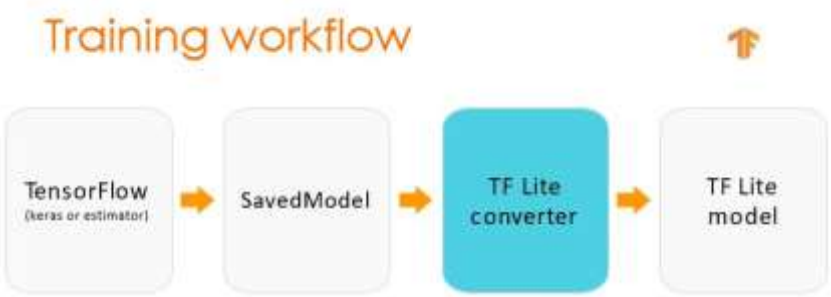


Şekil 2.28 Kullanılan Modelin Sinir Ağının Görselleştirilmesi [86]

Labelmap.txt dosyası, tüm olası kategorilerin sıralı bir listesini içerir. Bunlar, takip eden MobileNet modelini eğitmek için kullanılan COCO medyalar aynı hizadadır.

TensorFlow Lite Model Maker Kütüphanesi, özel bir veri kümesi kullanarak TensorFlow Lite modelini eğitme sürecini basitleştirir. Gerekli eğitim verisi miktarını azaltmak ve eğitim süresini kısaltmak için transfer öğrenmeyi kullanır.

TensorFlow Lite Model eğitimi iş akışı Şekil 2.29'de gösterilmiştir.



Şekil 2.29 TensorFlow Lite Model eğitimi iş akışı [67]

Mikrodenetleyiciler için TensorFlow Lite, yalnızca birkaç kilobayt belleğe sahip mikro denetleyicilerde ve diğer cihazlarda makine öğrenimi modellerini çalıştırmak için tasarlanmıştır. Çekirdek çalışma zamanı, Cortex M3 koluna 16 KB'ye sığar ve birçok temel modeli çalıştırabilir. İşletim sistemi desteği, standart C veya C ++ kitaplıkları veya dinamik bellek tahsisi gerektirmez. Bunu başarmak için, TensorFlow Lite dönüştürücü kullanılmıştır.

2.4.4.2. Desteklenen platformlar

Mikrodenetleyiciler için TensorFlow Lite, C ++ 11 ile yazılmıştır ve 32 bitlik bir platform gerektirir. Arm Cortex-M Serisi mimarisine dayalı birçok işlemci ile kapsamlı bir şekilde test edilmiş ve ESP32 dâhil olmak üzere diğer mimarilere taşınmıştır. Çerçeve bir Arduino kütüphanesi olarak mevcuttur. Ayrıca Mbed gibi geliştirme ortamları için projeler üretebilir. Açık kaynaklıdır ve herhangi bir C ++ 11 projesine dâhil edilebilir [73] [87].

Bir mikro denetleyicide bir TensorFlow modelini dağıtmak ve çalıştırmak için aşağıdaki adımlar gerçekleştirilmiştir:

2.4.4.3. IoT Model Optimizasyonu

IoT için optimize edilmiş algılama modelleri, çeşitli gecikme ve hassasiyet özellikleriyle birlikte gelir. Her biri aşağıda açıklanan giriş ve çıkış imzalarını takip eder.

Model Giriş İmzası, model girdi olarak bir görüntü alır, nesne algılama modelleri belirli bir boyuttaki girdi görüntülerini kabul eder. Bu durum, muhtemelen cihazınızın kamerası tarafından yakalanan ham görüntünün boyutundan farklı olacaktır ve ham görüntünün modelin giriş boyutuna uyacak şekilde kırpılması ve ölçeklendirilmesi için kod yazılmıştır. Modelin kullandığı boyut görüntüsü, her piksel başına üç kanal RGB (kırmızı, mavi ve yeşil) ile 300x300 pikseldir, bu görüntüler modele 270.000 baytlık değerlerde (300x300x3) düzleştirilmiş bir tampon olarak

beslenir. Kullanılan model hesaplandıkça, Çizelge 2.8'de niceleme alt bölümünde daha fazla ayrıntı vardır, her değer 0 ile 255 arasında bir değeri temsil eden tek bir bayt olmalıdır [73] [87].

Model Çıkış İmzası, 0-4 indislerine eşlenmiş dört dizi çıkarır. Model, 0-4 indislerine eşlenmiş dört dizi çıkarır. Çizelge 2.8 'de gösterildiği gibi 0, 1 ve 2 dizileri, her dizide her nesneye karşılık gelen bir öge ile N algılanan nesneyi tanımlar [73] [87].

Çizelge 2.8 0, 1 ve 2 dizileri, her dizide her nesneye karşılık gelen bir öge ile N algılanan nesneyi tanımlanmıştır [73] [87]

Index	Name	Description
0	Locations	Multidimensional array of [N][4] floating-point values between 0 and 1, the inner arrays representing bounding boxes in the form [top, left, bottom, right]
1	Classes	Array of N integers (output as floating-point values) each indicating the index of a class label from the labels file
2	Scores	Array of N floating-point values between 0 and 1 representing the probability that a class was detected
3	Number of detections	Integer value of N

Örneğin, model bir kişi ve arabayı algılayacak şekilde eğitilmiştir. Bir görüntü sağlandığında, bu görüntü Çizelge 2.9'te belirlenen sayıda algılama sonucunun çıkışını tahmin eder.

Çizelge 2.9 Belirlenen sayıda algılama sonucunun çıktısını tahmin edilmiştir [73] [87]

Class	Score	Location
Car	0.92	[18, 21, 57, 63]
Person	0.88	[100, 30, 180, 150]
Bird	0.87	[7, 82, 89, 163]

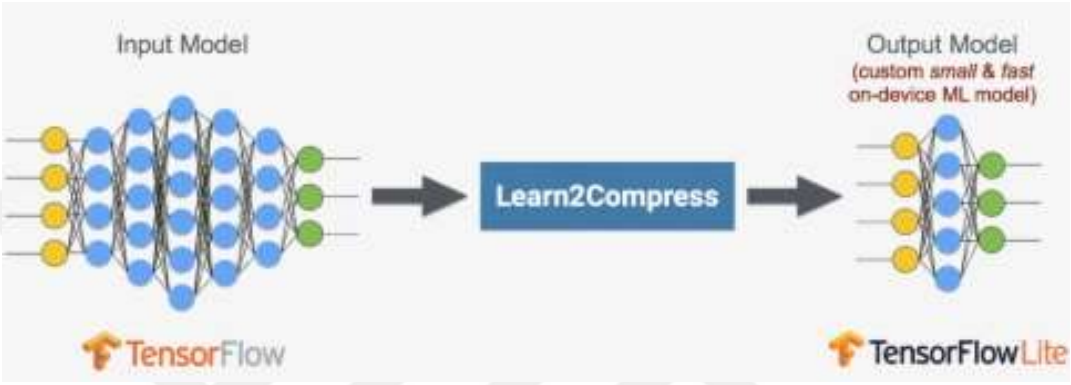
Nesne Konumunun Bulunması; model, algılanan her nesne için konumunu çevreleyen sınırlayıcı bir dikdörtgeni temsil eden dört sayı dizisi döndürmektedir. Sağlanan önceden eğitilmiş model için numaralar şu şekilde sıralanmıştır [73] [87].

[top, left, bottom, right]

Üst değer, dikdörtgenin üst kenarının görüntünün üst kenarına olan mesafesini piksel cinsinden temsil eder. Sol değer, sol kenarın giriş görüntüsünün solundan mesafesini temsil eder. Diğer değerler, benzer şekilde alt ve sağ kenarları temsil edilmektedir. Modelin piksel değeri çıktısı, kırpılan ve ölçeklenen görüntünün konumunu ifade eder. Doğru yorum yapabilmek için ham görüntüye sığacak şekilde ölçeklendirme yapılmıştır [73] [87].

2.4.4.4. Model Eğitim Sonrası Niceleme

Eğitim sonrası niceleme için kullanılan model, model boyutunu azaltabilen ve aynı zamanda model doğruluğunda çok az bozulma ile CPU ve donanım hızlandırıcı gecikmesini iyileştirebilen bir dönüştürme tekniğidir. Şekil 2.30'de gösterildiği gibi, TensorFlow Lite Converter kullanılarak TensorFlow Lite formatına dönüşüm yapıldığında, önceden eğitilmiş bir kayan TensorFlow modeli nicelendirilmiş olur.



Şekil 2.30 TensorFlow Lite Converter kullanılarak TensorFlow Lite formatına dönüşüm yapıldığını [67].

Nicelleştirilmiş Tensör Gösterimi, 8 bitlik nicemleme, aşağıdaki formül kullanılarak kayan nokta değerlerine yaklaşır [67].

$$\mathbf{real\_value} = (\mathbf{int8\_value} - \mathbf{zero\_point}) * \mathbf{scale}$$

Gösterimin iki ana bölümü vardır [36]:

• **Eksen başına** (diğer adıyla kanal başına) veya tensör başına ağırlıklar, sıfır noktası 0'a eşit olacak şekilde [-127, 127] aralığında int8 ikinin tamamlayıcı değerleriyle temsil edilir.

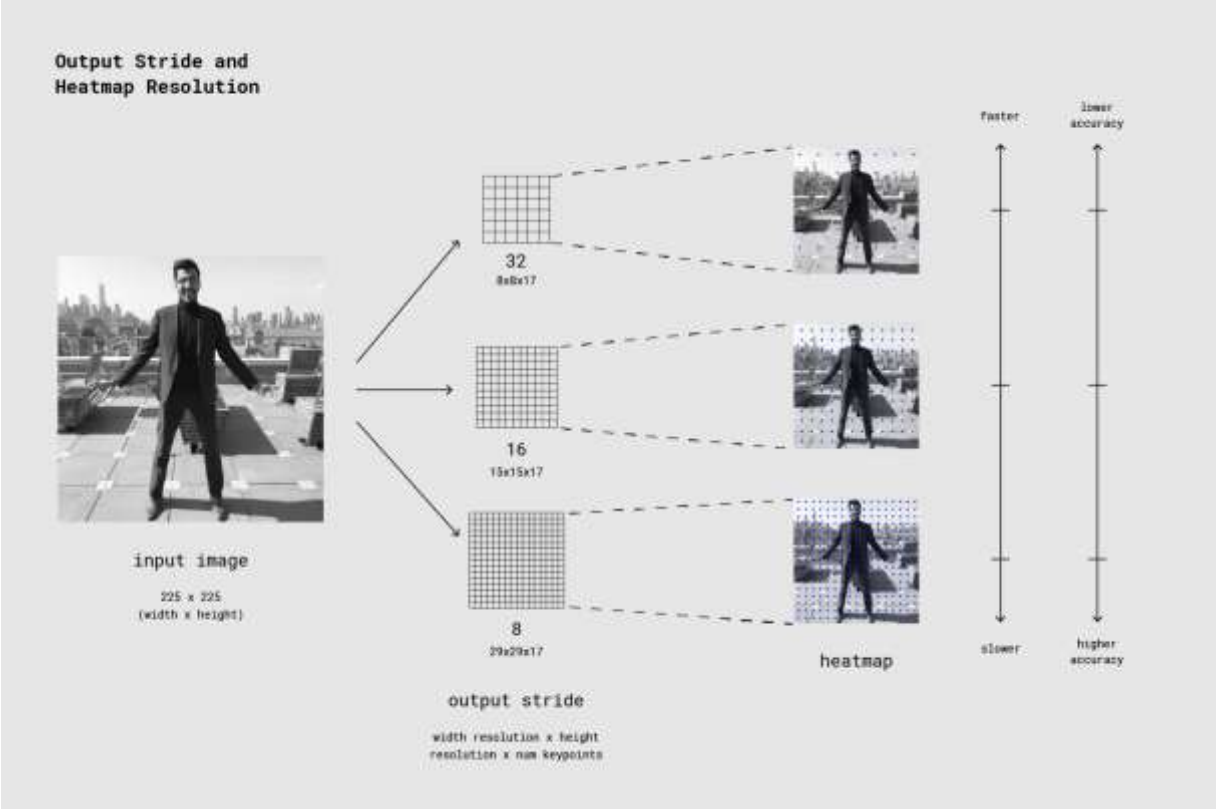
• **Tensör başına** etkinleştirmeler / girdiler, [-128, 127] aralığında int8 ikinin tamamlayıcı değerleriyle ve [-128, 127] aralığında bir sıfır noktasıyla temsil edilir.

2.4.4.5. Hesaplanmış Model Doğruluğu

Ağırlıklar eğitim sonrasında hesaplandığından dolayı, özellikle küçük ağlar için bir doğruluk kaybı olabilir. TensorFlow Lite model deposundaki belirli ağlar için önceden eğitilmiş ve tam olarak nicelendirilmiş modeller sağlanır. Hesaplamada herhangi bir bozulmanın kabul edilebilir sınırlar içinde olduğunu kabul etmek için nicelleştirilmiş modelin doğruluğunu kontrol etmek önemlidir. TensorFlow Lite model doğruluğunu değerlendirmek için araçlar vardır.

Alternatif olarak, doğruluk düşüşü çok yüksekse, nicelemeye duyarlı eğitim kullanılabilir. Ancak bunu yapmak, model eğitimi sırasında sahte niceleme düğümleri eklemek için değişiklik yapılmasını gerektirir. Bu çalışmada eğitim sonrası niceleme teknikleri, önceden eğitilmiş mevcut bir model kullanmaktadır [67] [88].

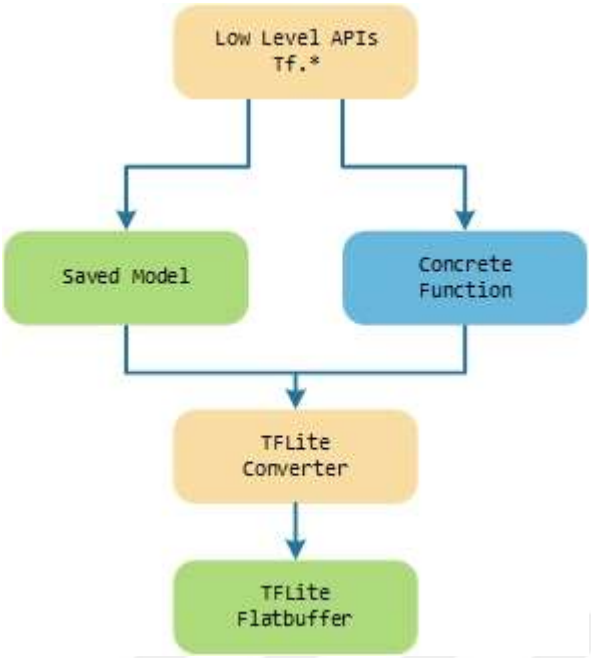
Şekil 2.31 deki görüntü çıktı adımının (stride), çıktının girdi görüntü boyutuna göre ne kadar küçültüldüğünün nasıl belirlediğini gösterir. Daha yüksek olan çıkış adımı daha hızlı çalışır ancak daha düşük doğruluk sağlar.



Şekil 2.31 Hesaplama Çıkışı Adımları ve Isı Haritası Çözünürlüğü [88]

2.4.4.6. Model Dönüştürücü

Bu bölümde, Şekil 2.32'da gösterildiği gibi TensorFlow Nesne Algılama API'sinden TensorFlow Lite'a dönüştürülen Single-Shot Detector modellerinin imzası açıklanmaktadır.



Şekil 2.32 API'sinden TensorFlow Lite'a dönüştürülen Single-Shot Detector [89].

Kullanılan nesne algılama modeli, birden çok nesne sınıfının varlığını ve konumunu algılamak için önceden eğitilmiştir. Modelimiz, temsil ettikleri kişinin sınıfını (örneğin, bir kişi veya araba) ve her bir nesnenin göründüğü yeri seçen verileri belirten bir etiketle birlikte çeşitli nesnelerin resimlerini içeren görüntülerle eğitilmiştir [90].

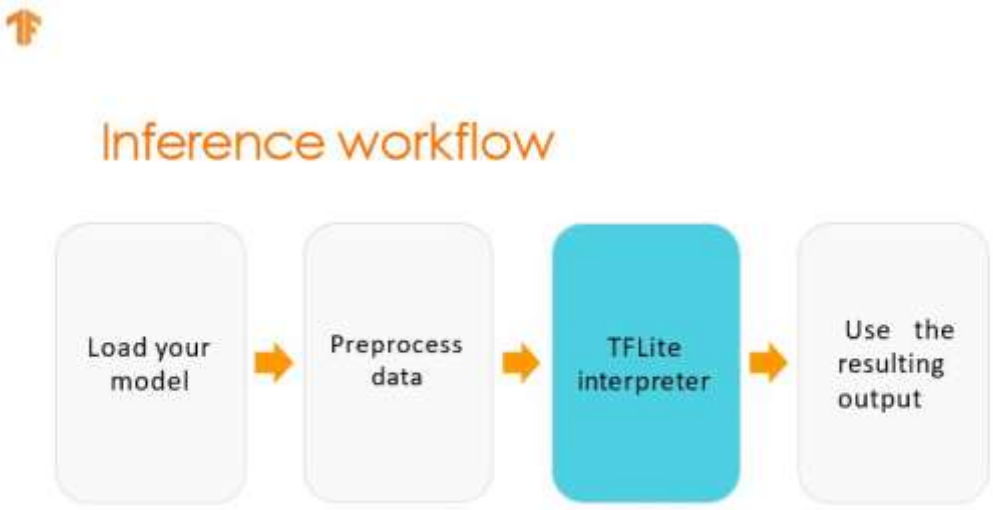
Modele sonradan bir görüntü verildiğinde, algıladığı nesnelerin bir listesini, her nesneyi içeren bir sınırlayıcı kutunun konumunu ve algılamanın doğru olduğuna dair güveni gösteren bir puan verir. Model eğitim aşaması bittikten sonra model çalıştırma aşamasına geçilmektedir [65] [91].

2.5. Faz III: Modelin Çalıştırılması, Sonuçların Elde Edilmesi, Kod Analizi

2.5.1. Model Tasarım ve Uygulanması

Bu bölümde model tasarım ve gerçeklemesi faz II'de belirtilen gereksinimler tamamlanarak yazılım ve donanım uygulaması yapılmıştır. Algoritma için kod dosyaları hazırlanmış ve kodun çalıştırılabilmesi için gerekli ayarlamalar yapılmıştır. IoT cihazlarında derin öğrenme kullanılarak eylem algılama modeli hazır hale getirilmiştir.

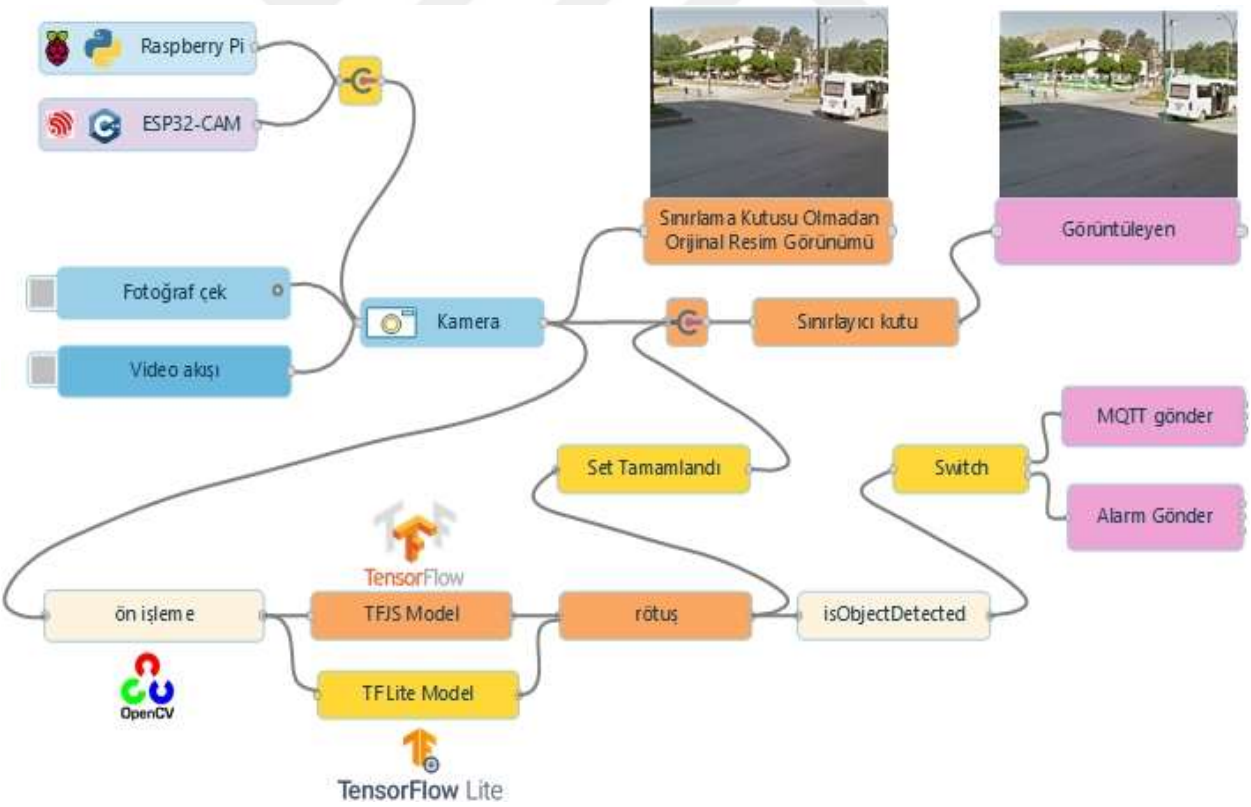
Bu bölümde anlatılan tasarımının amacı modelin IoT ve gömülü cihazlar üzerinde çalıştırılması ve makul sonuçların üretilmesidir. Bunun için, veriler tf.data kütüphanesiyle yüklenmelidir. Bu doğrultuda faz II kodlar verilmiş ve modelin alt bölümleri detaylı olarak eğitilmiştir. TensorFlow Lite modelini IoT ve Gömülü cihaza dağıtmak, Modeli IoT'ye yüklemek ve çalıştırmak için Python API kullanılmıştır. Gömülü cihazlarda ise modeli yüklemek ve çalıştırmak için C ++ API kullanılmıştır. C ++ ve Python API'leri model üzerinde çıkarım yapmadan önce TensorFlow eğitim kodunda kullanılan Session sınıfından farklı olarak TensorFlow-lite'a özgü Interpreter sınıfını kullanır. Yorumlayıcıyı için hem C ++ hem de Python kullanılmış ve sonuçları sonraki iki bölümde paylaşılmıştır [92]. Ayrıca IoT cihazlarına yerleştirilecek modeli hazırlamak için freeze_graph aracı kullanılmış ve çıkış düğüm adı "person" olarak tanımlanmıştır. Tahmin modelini çalıştırmak için aynı Model gömülü cihazda da kullanılmıştır [88]. Şekil 2.33'de modelin yükleme aşamasından çıktı sonuçlarına kadar olan tam yol gösterilmektedir.



Şekil 2.33 TensorFlow Lite Çıkarım İş Akışı [93].

Kullanılan algoritmada, önce yerel diskten önceden eğitilmiş model yüklenmiştir. Böylece kodlar verileri işleyebilmektedir ve TensorFlow Lite yorumlayıcı aşamasının bir sonraki adımına geçilmektedir. Bu aşamada, video çerçeveleri girdi olduğu için bir açılır pencere görünmektedir. Giriş akışı gerçek zamanlı video akışı sağladığı sürece kod while_loop'u tekrarlayacaktır. Her video karesi, hareketli bölgeleri video karesinden çıkarmak için ayrı ayrı işlenir. Her video karesindeki hareketli bölgeler, eylem tespiti için TensorFlow Modeline geçirilmektedir. Çerçevenin boyutu eşikten küçükse, dikkate alınmamaktadır. Aksi takdirde, çerçeveyi belleğe iter. Ardından, nesnenin etrafındaki sınırlayıcı kutunun veri kümesinde (COCO) eşleşip eşleşmediği kontrol edilmektedir.

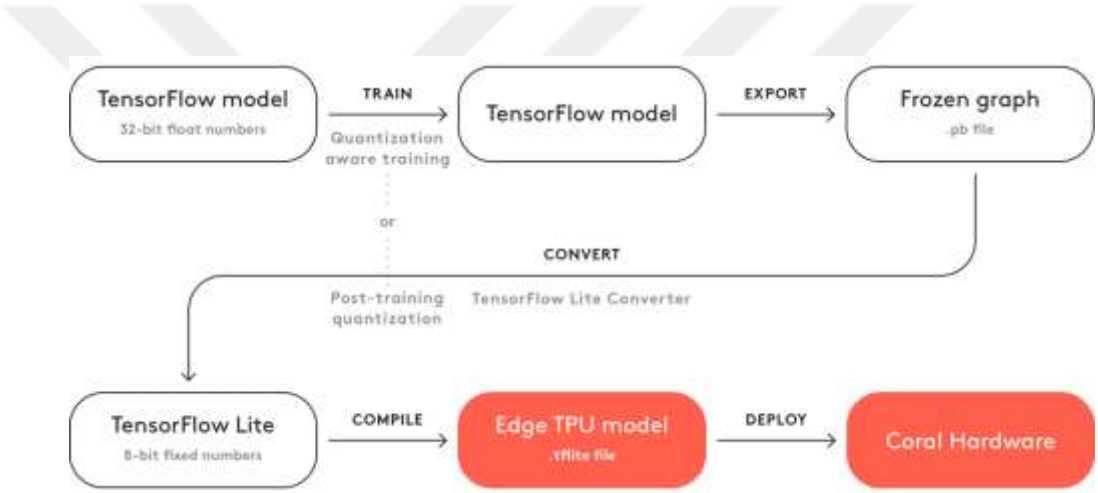
Daha sonra, o hareketli nesnenin etiketi bulunur. Varsa, tespit edilen nesne etiketlenir. Tespit edilen nesne "kişi" ise, mesaj MQTT aracısına yayınlanır ve ilgi alanında meydana gelen yeni eylemi son kullanıcıya bildirmek için SMS veya E-posta gönderilir. Son olarak, video akışından sonraki kareyi işlemek için bellek temizlenir. (Kodlama işlemi Şekil 2.34 Kod işleme işlevi akış şeması'daki akış şemasında ayrıntılı olarak gösterilmiştir) [94].



Şekil 2.34 Kod işleme işlevi akış şeması

Ardından, hızlandırıcı Coral (TPU) ve Movidius (VPU) içeren bir cihaz eklemek için IoT cihazında aynı model aynı parametrelerle yeniden çalıştırılmıştır. Donanım hızlandırıcılarla yapılan tüm çıkarımların TensorFlow Lite API'lerine (Python veya C / C ++) dayandığını akılda tutmak önemlidir. Hali hazırda TensorFlow Lite kullanılarak bir çıkarım yapıldığından, modeli IoT HW hızlandırıcıda çalıştırmak birkaç yeni kod satırı gerektirmektedir. IoT donanım hızlandırıcılar, TensorFlow Lite modelleriyle uyumludur.

Uyumlu bir modelle hem Python hem de C / C ++ kullanılarak IoT donanım hızlandırıcı üzerinde çıkarım yapılmaktadır. Her iki durumda da, IoT ve Gömülü cihazlar için, TensorFlow API'lerini ve diğer gelişmiş özellikleri saran kullanışlı işlevler sağlayan Coral API'lerini kullanma seçeneği de mevcuttur. Her seçenek ve gerekli yazılım aşağıda açıklanmış ve Şekil 2.35'te gösterilmiştir.



Şekil 2.35 çıkarım için üç seçenek ve ilgili yazılım bağımlılıkları [73].

Modelin TensorFlow Lite Python API kullanılarak IoT üzerinde çalıştırılması:

IoT cihazlarında (RPi 3B ve 4B) TensorFlow Lite modellerini çalıştırmak için standart Python API kullanılmıştır. Modeli çalıştırmak için Edge donanım hızlandırıcılarında mevcut bir TensorFlow Lite kodu çalıştırılmıştır. Kodun donanım hızlandırıcı tarafından desteklenen IoT cihazlarında çalıştırılması için TensorFlow Lite Python API ve IoT donanım hızlandırıcı Çalışma Zamanı (libedgetpu) kullanılmıştır.

Varsayılan olarak, TensorFlow Lite yorumlayıcısı modeli CPU üzerinden yürütülmektedir. Ancak model IoT donanım hızlandırıcısı için derlenmişse işlem başarısız olur. Çünkü bir IoT HW hızlandırıcı modeli, özel bir IoT HW hızlandırıcı operatörü içermektedir. IoT HW hızlandırıcısı için bir temsilci belirlenmesi gerekir. Ardından, Yorumlayıcı IoT HW hızlandırıcı özel

operatörüyle her karşılaştığında, bu işlemi IoT HW hızlandırıcısına gönderir. Bu nedenle, IoT HW hızlandırıcısında çıkarım için yalnızca TensorFlow Lite API gerekmektedir.

2.5.2. Modelin C / C ++ TensorFlow Lite API kullanılarak gömülü cihazda çalıştırılması

TensorFlow Lite modellerini çalıştırmak için standart C / C ++ API kullanılmıştır. Bu hızlandırıcı kurulumu için, statik veya dinamik olarak bağlanan Edge donanım hızlandırıcı Çalışma Zamanı (libedgetpu) ve derlenmiş TensorFlow Lite C ++ kütüphanesi kullanılmıştır. Standart modellerde tensör girdilerini ve işlem sonrası tensör çıktılarını önceden işlemek için çeşitli libcoral API'leri de kullanılabilir. Bu durum, bir modelin birden çok IoT donanım hızlandırıcıyla boru hattı oluşturması gibi ek özellikler sağlar. C ++ ile Edge HW hızlandırıcısında çalıştırma çıkarımı hakkında ayrıntılar önceki bölümlerde paylaşılmıştır. Coral cihazı kurulurken modelin yüklenmesi gereken tüm kütüphaneler alınmıştır(derleme yapılandırmaları gerektiren C ++ hariç). Desteklenen platformlardan biri kullanılmadığı durumlarda bu kütüphaneler istenen platformun amacına uygun olarak oluşturulabilir. Yukarıda gösterilen her kaynak açık kaynaklıdır.

Model parçasının eğitimi tamamladıktan sonra, bir kişinin varlığını veya yokluğunu algılamak için eylem algılamayı gerçekleştirmeye ve kamera verilerini bir görüntü sensörüyle yakalama işlemleri gerçekleştirilmiştir. Bunun için aşağıdaki adımlar uygulanmış ve bir mikro denetleyicide TensorFlow modeli çalıştırılmıştır.

2.5.3. IoT'de Eylem Algılama Çalıştırılması

Bu alt bölümde yapılan işlemler temel olarak model çalıştırır ve OpenCV kullanarak kameradan bir kare yakalar, çerçeveyi 300x300 piksele yeniden boyutlandırır (en boy oranını korumaz) ve elde edilen tensörü TensorFlow Lite'a iletir. İnvoke () işlevi, çerçevede algılanan nesnelerin bir listesi, her nesnenin bir güven puanı ve sınırlayıcı kutuları için koordinatlarla geri döner. Kod, bu koordinatları alır ve kullanıcıya göstermeden önce nesnenin etrafına yeşil bir dikdörtgen çizer. İlk olarak, yeniden boyutlandırılabilen bir pencere oluşturmak için cv2.WINDOW_NORMAL kullanılmıştır. İkinci olarak, her nesnenin merkezini hesaplayan ve algılanan nesneleri konsola listeleyen bir bölüm eklenmiştir. Verilen Kod çalıştırılınca IoT kamera veya gömülü cihaz kamerasının beslemesini sağlayan yeni bir pencere açılır.

Kod, TensorFlow Modelimizi yüklemek, giriş görüntüsünü modele beslemek, modeli çalıştırmak ve çıkarım sonucunu döndürmek için kullanılmıştır. Daha sonra model, çıkış düğümü adlarından oluşan bir sıra dizisi geçirilerek çalıştırılmıştır. Burada hızlı stil aktarım modeli için,

sadece bir giriş düğümü ve bir çıkış düğümü vardır. Son olarak, modelin çıktı değerleri çıktı düğümü adı atlanarak alınmıştır.

2.5.4. Kıyaslama Kodu

Bu bölümde, Raspberry Pi 3 Model B, Pi 4 Model B ve ESP32-CAM üzerinde TensorFlow, TensorFlow Lite ve TensorFlow Lite Micro kullanılarak yapılan orijinal kıyaslamalar açıklanmıştır. TensorFlow karşılaştırmaları, TensorFlow çerçevesinin performansını test etmek için tasarlanmış açık kaynaklı makine öğrenimi uygulamalarıdır. TensorFlow Lite karşılaştırma araçları, aşağıdaki önemli performans metrikslerine ilişkin istatistikleri ölçer ve hesaplar.

Bunun için izlenen adımlar;

• Başlatma süresi

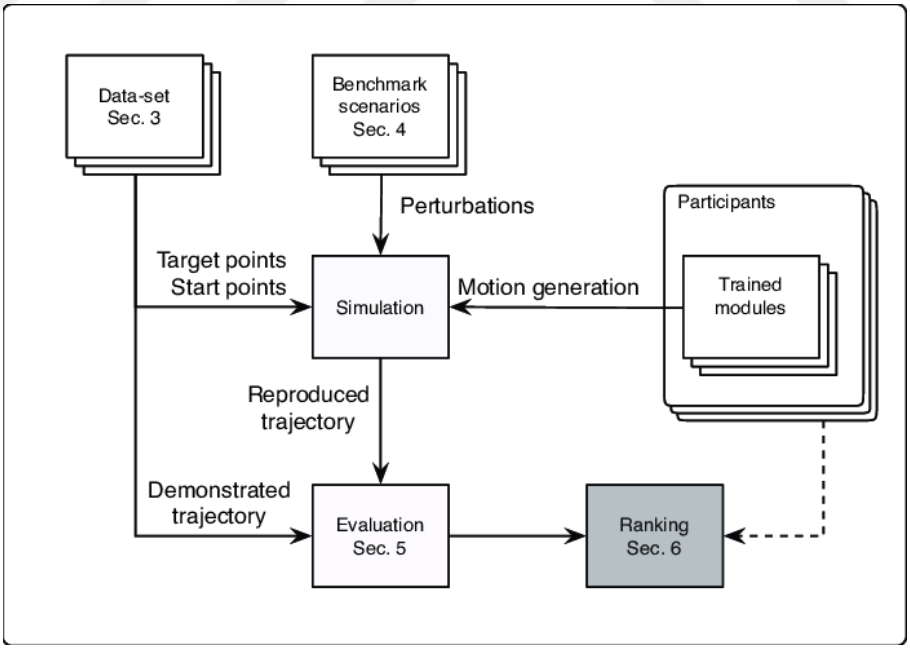
• Isınma durumunun çıkarım süresi

• Kararlı durumun çıkarım süresi

• Başlatma süresi sırasında bellek kullanımı

• Genel bellek kullanımı

Kullanılan cihaza bağlı olarak, bu seçeneklerden bazıları mevcut olmayabilir veya hiçbir etkisi olmayabilir. Kıyaslama aracı aynı zamanda yerel bir ikili benchmark_model olarak sağlanır. Bu araç, Linux, Mac, gömülü cihazlar ve IoT cihazlarında bir kabuk komut satırından çalıştırılabilir. Kıyaslama mimarisinin şematik gösterimi Şekil 2.36'de gösterilmiştir.



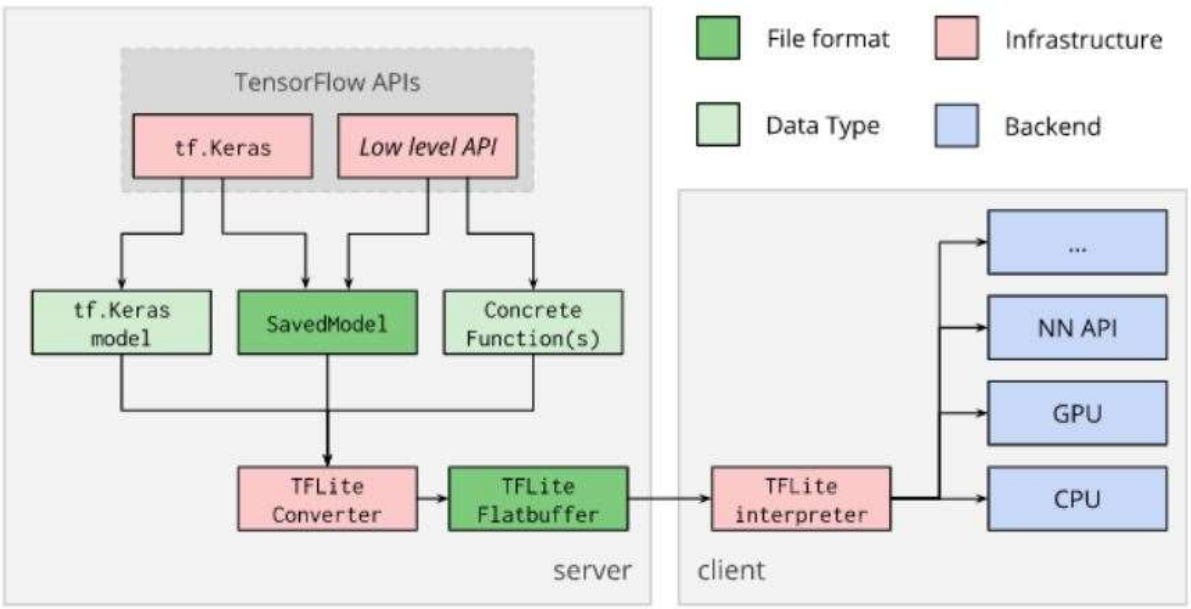
Şekil 2.36 Kıyaslama mimarisinin şematik gösterimi [95]

Tek bir çalıştırmada birden çok performans seçeneğini karşılaştırmak için kullanışlı ve basit bir C ++ ikili programı da sağlanmıştır. Bu ikili program, bir seferde yalnızca tek bir performans

seçeneğini kıyaslayabilen yukarıda bahsedilen araca dayalı olarak oluşturulmuştur. Aynı derleme / yükleme / çalıştırma sürecini paylaşırlar, ancak bu ikili dosyanın BUILD hedef adı benchmark_model_performance_options'tır ve bazı ek parametreler gerektirir. Bu ikili program için önemli bir parametreler aşağıda verilmiştir.

TensorFlow karşılaştırma aracı yerel komut satırı ikili dosyaları olarak kullanılmıştır. IoT cihazları arasındaki çalışma zamanı ortamındaki farklılıklar nedeniyle mevcut seçeneklerin ve çıktı formatlarının biraz farklı olduğu unutulmamalıdır.

Kod ikinci kez her iki durum arasındaki farkları göstermek için hızlandırıcı donanımı ile uygulanmıştır. Çıkarım, Bağlamda Ortak Nesneler (COCO) veri seti üzerinde eğitilen model olan MobileNet SSD derinlik SSD modeli ile gerçekleştirilmiştir. TensorFlow API dönüştürücü tarafından TensorFlow Lite'a dönüştürülmüştür. Bu model, COCO veri kümesiyle çalışan en hızlı model olduğu için seçilmiştir. TensorFlow API dönüştürücü Şekil 2.37'de gösterilmiştir [96]



Şekil 2.37 TensorFlow API dönüştürücü [96].

Model, oda içinde eylem algılama kodu çalıştırılarak test edilmiş ve daha sonra IoT'ye ve gömülü cihazlara 5 voltluk güç kaynağı sağlamak için bir telefon güç bankası kullanılarak dışarıda çalıştırılmıştır ve sonuç gerçek zamanlı gecikme ve algılama doğruluğu olmak üzere iki faktöre odaklanmıştır.

2.5.5. Çalışmanın Sonuçlarının Oluşturulması

Algoritmanın eğitim sürecinin bir önceki bölümde tamamlanmasının ardından, bu bölümde IoT ve gömülü cihazlarda çalışan eylem algılama modelinden sonuç üretilmesi açıklanmaktadır. Sonuçlar, üç faktörü temsil eden önceki iki bölümden toplanacaktır. İlki, çalışan modelin hızını temsil eden karşılaştırmadır. İkincisi FPS, üçüncüsü ise doğruluk için güven oranıdır. Bu üç faktör, IoT cihazlarında Derin öğrenme algoritması performansını değerlendirmek için test verileri olarak kullanılmıştır.

Bir video iki sınıftan oluşur: gerçek ortam için küçük ölçekli kişi, araba ve modeli çalıştırmak için kullandığımız algoritmayı değerlendirmek için kanıt verileri olarak 300x300 piksel laboratuvar ortamına sahip hareketsiz görüntü. Yapılan testin sonuçları şu şekildedir;

• **Test videosunun** her karesi, kıyaslama toplamak için analiz edilmiştir. Bunlar Doğruluk, Kesinlik ve Geri Çağırma hesaplamaları için gereklidir. Böylece performans puanları kontrol edilerek performans ölçümü yapılabilir.

• **Şekil 2.38-2.40'ye** kadar TensorFlow, TensorFlow Lite ve TensorFlow Lite Micro tarafından yapılan algılamaların ekran görüntülerinin yanı sıra algılamaların güven aralıkları gösterilmektedir.

• Verilen şekillerde de görüldüğü gibi, modellerin kişileri ve ulaşım araçlarını değişik açı ve mesafelerden maksimum % 99 güvenlikle başarılı bir şekilde algılayıp takip ettiği görülmüştür.

• Testler, web kamerasından canlı yayın sağlanarak ve test verileri olarak gerçek ulaşım araçları kullanılarak birden çok kez gerçekleştirilmiştir.

Şekil 2.38 TensorFlow tarafından yapılan ilk tahminler

Şekil 2.39 TensorFlow Lite tarafından yapılan ikinci tahminler

Şekil 2.40 TensorFlow Lite Micro tarafından yapılan üçüncü tahminler

2.5.5.1. Güven puanı

Sonuçların yorumlanmasında tespit edilen her nesnenin puanına ve konumuna bakılır. Puan, nesnenin gerçekten algılandığına dair güveni gösteren 0 ile 1 arasında bir sayıdır. Sayı 1'e ne kadar yakınsa model o kadar kesinleşir. Bu kesinleşme % 100 olasılık oranıyla temsil edilir.

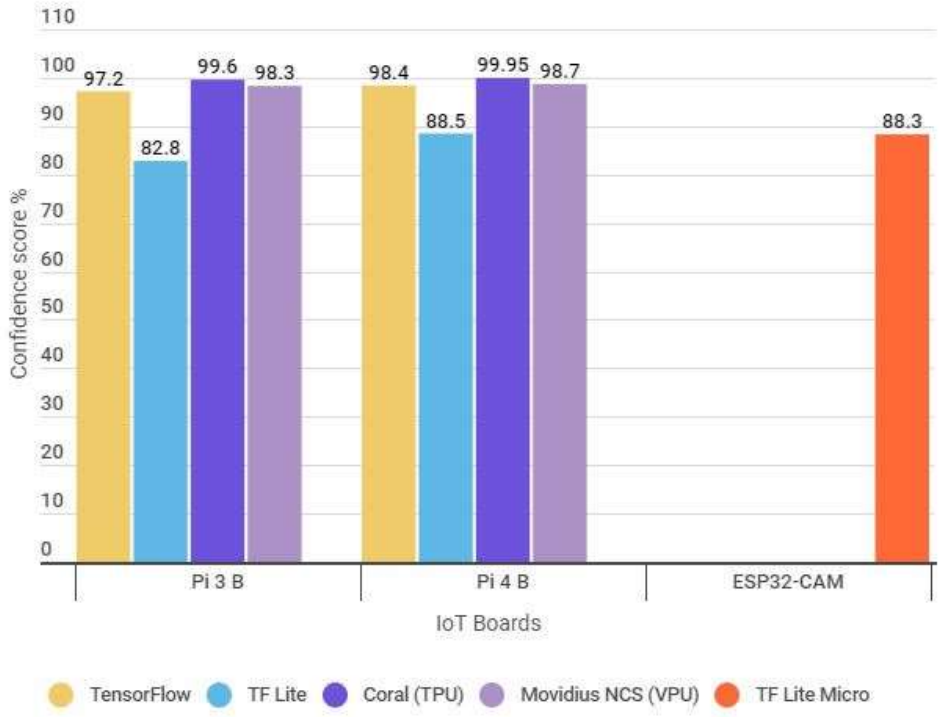
Modele bağlı olarak, altında tespit sonuçlarının atanacağı bir kesme eşiğine karar verilebilir. Mevcut model için gerçekçi bir kesme 0.5 puandır. Bu, tespitin geçerli olduğuna dair % 50 olasılık anlamına gelir. Bu durumda, dizideki son iki nesne göz ardı edilir. Çünkü bu güven puanları 0.5'in altındadır.

İnce ayarlı eylem algılama modeli,% 99'luk bir algılama doğruluğuna ulaşmıştır. İnce ayar öncesinde ve sonrasında IoT modelinin uygulanmasının sonuçları Şekil 2.38-2.40'ye kadar gösterilmektedir. Üç IoT kamerasının gündüz görüntüleri, modelin performansını gerçek dünya senaryolarında gösterir. IoT cihazlarında model algılama güven puanı çalıştırma sonuçları Çizelge 2.10'da gösterilmiştir.

Çizelge 2.10 IoT cihazlarında model algılama doğruluk puanı çalıştırma

	TF	TFL	Coral (TPU)	Movidius NCS (VPU)	TFL Micro
Pi 3 B	97.2	82.8	99.62	98.3	-
Pi 4 B	98.4	88.5	99.95	98.7	-
ESP32-CAM	-	-	-	-	88.3

Cihazlarında model algılama doğruluk puanı çalıştırmanın sonuçları Şekil 2.41'de gösterilmiştir.



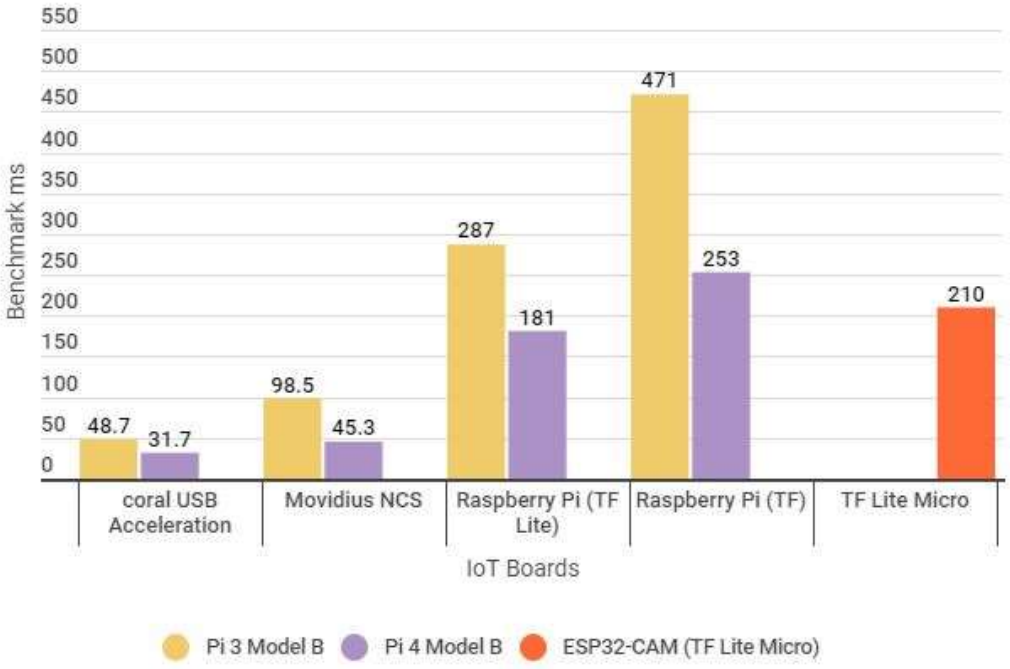
Şekil 2.41 IoT cihazlarında model algılama güven puanı çalıştırma

Ayrıca, sistemin gerçek zamanlı performansını değerlendirmek için, farklı gömülü donanım ve IoT cihazlarının kamera sayısı göz önüne alındığında her kurulumun elde ettiği maksimum FPS sayısı değerlendirilmiştir. Bu amaçla çeşitli olayların yer aldığı 120 saniye süreli bir video kaydedilmiştir. Bu video, her bir gömülü cihaza ve donanım hızlandırıcıların her biri ile aynı anda beş kereye kadar (yani, beş kameralı akışları simüle ederek) yayınlanmıştır. IoT cihazları, o cihaza uygulanan TensorFlow Modelini kullanarak her çerçevede eylem algılama gerçekleştirmiştir. TensorFlow'a yönelik NCS (Movidius) desteği sınırlı olsa da, çalışan algoritmanın kullanımını

kıyaslamak için SSD MobileNet modeli kullanılmıştır. Çizelge 2.11'da görüntülenen sonuçlar verilmiştir. Şekil 2.42'de ise, aynı sonuçları çubuk grafik biçiminde göstermektedir.

Çizelge 2.11 IoT cihazlarda çalışan model için performansı gerçek zamanlı ms olarak değerlendirilen sonuçları

	RPi 3 Model B	**R Pi 4 Model B**	**ESP32-CAM (TF Lite Micro)**
Coral USB	48.7	31.7	-
Movidius NCS	98.5	45.3	-
TF Lite	287	181	-
TF	471	253	-
TF Lite Micro	-	-	210



Şekil 2.42 IoT cihazlarda çalışan model için performansı gerçek zamanlı olarak değerlendirilen grafik

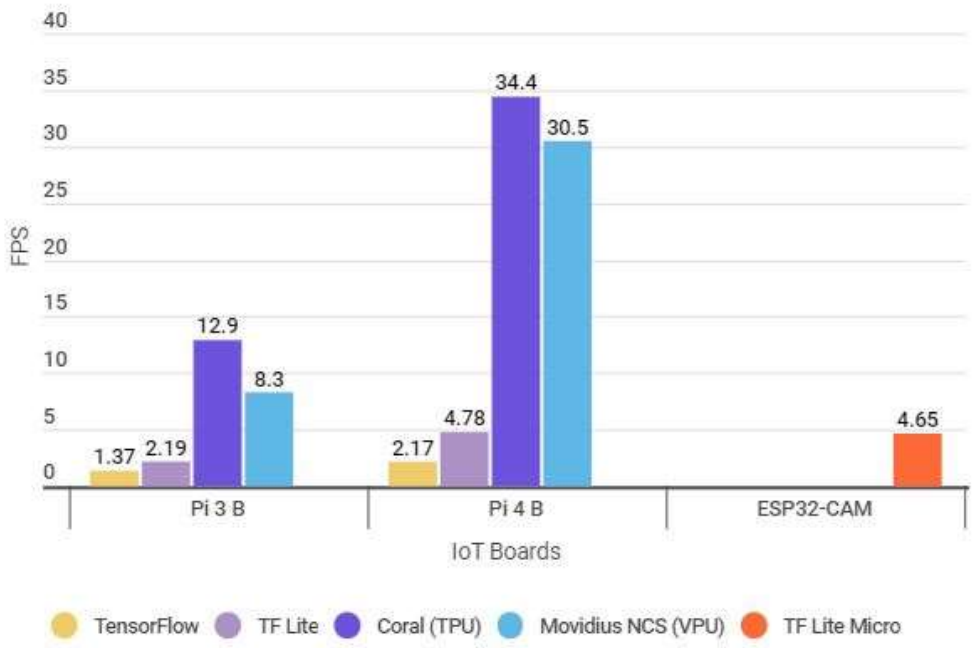
Her adımdaki FPS sayısı ve doğruluk sınırlayıcı kutu güven oranının yüzdesi Şekil 2.38-2.40'ye kadar görülebilir. Sınıflandırma doğruluğunun FPS'nin yüksek oranında çok yüksek bir değerde başladığı unutulmamalıdır. Düşük donanım, algoritma çalışırken modelde FPS'yi ve eylem algılamanın doğruluğunu kademeli olarak azaltır [89]. Ayrıca her tanımlama için bir güven

kontrolü (yani olasılık) gerçekleştirilmiştir. Güven yeterince yüksekse (yani eşiğin üstünde ise), tahmini uçbirimde ve eylem için metin ve renk sınırlayıcı kutusunda görüntülenir. Bu olay önceki bölümlerde kod akış şemasında Şekil 2.25'de açıklanmıştır. FPS sayısı ve doğruluk sınırlayıcı kutu güven oranının yüzdesi Çizelge 2.12'de verilmiştir.

Çizelge 2.12 FPS sayısı ve doğruluk sınırlayıcı kutu güven oranının yüzdesi

	TensorFlow	**TF Lite**	**Coral (TPU)**	**Movidius NCS (VPU)**	**TF Lite Micro**
Pi 3 B	1.37	2.19	12.9	8.3	-
Pi 4 B	2.17	4.78	34.4	30.5	-
ESP32-CAM	-	-	-	-	4.65

FPS sayısı ve doğruluk sınırlayıcı kutu güven oranının yüzdesi Şekil 2.43'de verilmiştir.

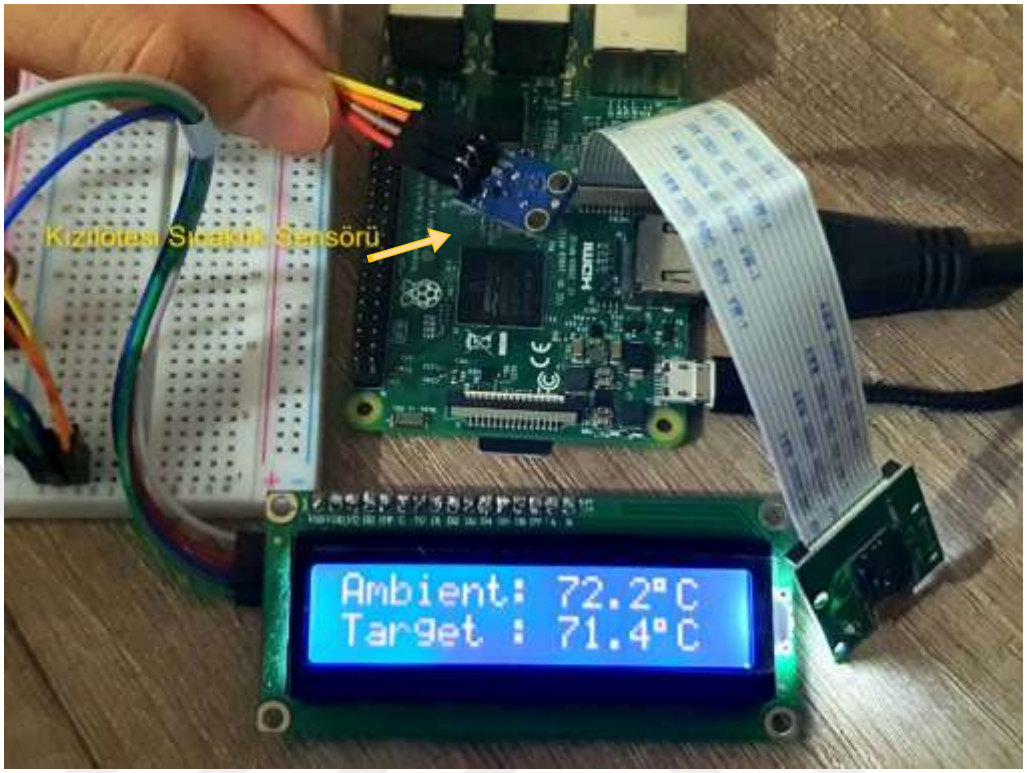


Şekil 2.43 FPS sayısı ve doğruluk sınırlayıcı kutu güven oranının yüzdesi

2.5.5.2. IoT Cihazlarının Sıcaklık etkisi

Cihazların sıcaklığı modelin çalışma süresine göre değişmektedir. Bu değişiklik ve cihaz üzerindeki etkilerini görmek oldukça önemlidir. Sıcaklık değişimini algılamak için Şekil 2.44'de

gösterildiği gibi sıcaklık sensörlü kullanılmıştır. Şekilde Raspberry Pi 3 IoT cihazının CPU tepe noktasından yakalanan sıcaklığı gösterilmektedir.



Şekil 2.44 Raspberry Pi 3 IoT cihazının CPU tepe noktasından çekilen sıcaklığı

2.5.5.3. Gün ışığı ve mesafe etkisi

Şekil 2.38-2.40'de, nesne algılama algoritmasının tipik çıktısı gösterilmektedir. Bu örnek şekilde,% 99,13 doğrulukla kişi ve% 99,86 doğrulukla ulaşım aracı olmak üzere iki nesne algılanmış ve karşılık gelen doğrulukla etiketlenmiştir. Bazı araçlar ve kişilerin tespit edilme doğruluğunun, düşük ışık ve kameradan uzak olmaları nedeniyle düşük çıktığı gözlemlenmiştir. Çizelge 2.103, değişen deneysel koşullar altında nesne algılama doğruluğu açısından performans özet sonuçlarını sunmaktadır. Bu sonuçlardan, standart TensorFlow derin öğrenme modelinin kullanılmasının yüksek doğruluk verdiği anlaşılmaktadır. Işık seviyesi ne çok yüksek ne de çok düşük olmalıdır. Geleneksel ışık, Çizelge 2.14'de verildiği gibi 201-1000 lüks arasında olmalı, nesne kameraya yakın ve çerçeve boyutu küçük, yani 300 × 300 olmalıdır. Işık seviyesini düşürülünce, karanlıkta nesneyi tespit etmek kolay olmadığından doğruluk da düşmektedir. Nesne kameranın yakınındaysa, doğruluk çok yüksek çıkmaktadır ve nesne kameradan uzaklaştıkça doğruluk da düşmektedir. Video karesinin boyutu arttırılınca video bulanıklaşmakta ve doğruluk düşmektedir. Bulanık videoda nesne tespit etmek zorlaşmaktadır.

Çizelge 2.13 Gündüz ve gece standart dış mekân ışık seviyeleri [97]

Durum	Aydınlatma	
	(ftcd)	(lux)
Sunlight	10000	107527
Full Daylight	1000	10752
Overcast Day	100	1075
Very Dark Day	10	107
Twilight	1	10.8
Deep Twilight	0.1	1.08
Full Moon	0.01	0.108
Quarter Moon	0.001	0.0108
Starlight	0.0001	0.0011
Overcast Night	0.00001	0.0001

Çizelge 2.14 Farklı çalışma alanları için önerilen ışık seviyeleri [97]

Aktivite	Aydınlatma	
	FC	LUX
Warehouses, Homes, Theaters, Archives	13.95	150
Easy Office Work, Classes	23.25	250
Normal Office Work, PC Work, Study Library, Groceries, Show Rooms, Laboratories	46.50	500
Supermarkets, Mechanical Workshops, Office Landscapes	69.75	750
Normal Drawing Work, Detailed Mechanical Workshops, Operation Theatres	93.00	1,000
Detailed Drawing Work, Very Detailed Mechanical Works	139.50 - 186.00	1,500 - 2,000

İşletim sistemi, Raspberry Pi, Coral ve NVIDIA Jetson Nano'da aşağıdaki sonuç CPU sıcaklıklarını bildirmiştir.

```
$paste      <(cat      /sys/class/thermal/thermal_zone*/type)      <(cat
/sys/class/thermal/thermal_zone*/temp)  |  column  -s  $'\t'  -t  |  sed
's/\(.\)..$/.\1°C/'
AO-therm                                                          38.0°C
CPU-therm                                                         31.0°C
GPU-therm                                                         30.5°C
PLL-therm                                                         28.5°C
PMIC-Die                                                         100.0°C
thermal-fan-est  31.0°C
```

TensorFlow çalıştıran hızlandırılmamış Raspberry Pi kartı, genişletilmiş test sırasında 75 ° C'lik bir sıcaklığa ulaşmıştır ve ek kademeli kısılmanın meydana geleceği 80 ° C noktasına yaklaşmıştır. Bu CPU'nun termal kısılmasından muzdarip olduğu anlamına gelir. Bu nedenle, Raspberry Pi'yi kullanarak uzun süreler boyunca çıkarım yapılması durumunda, CPU kısılmasını önlemek için pasif bir soğutucu eklenebilir yada küçük bir fan kullanılabilir. Derin öğrenme modelini çalıştırırken tüm IoT cihazı için yakalanan sıcaklık değerleri Çizelge 2.15'te verilmiştir.

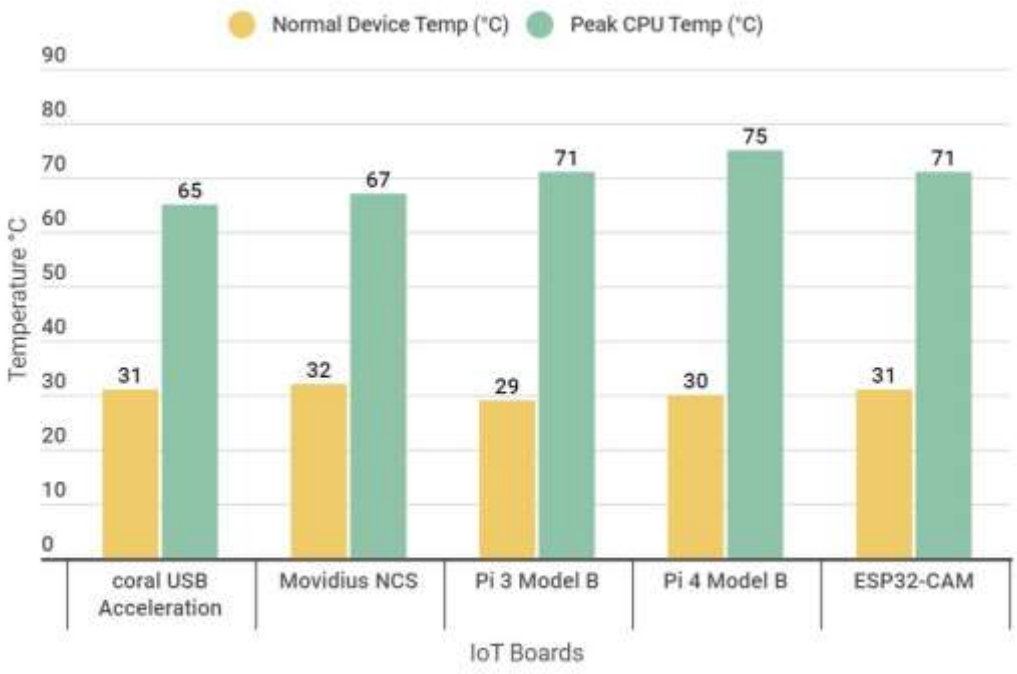
Çizelge 2.15 Derin öğrenme modelini çalıştırırken tüm IoT cihazı için yakalanan sıcaklık değerleri

	Ortalama Cihaz Sıcaklığı (°C)	**Tepe CPU Sıcaklığı (°C)**
Coral USB	31	65
Movidius NCS	32	67
Pi 3 Model B	29	71
Pi 4 Model B	30	75
ESP32-CAM	31	71

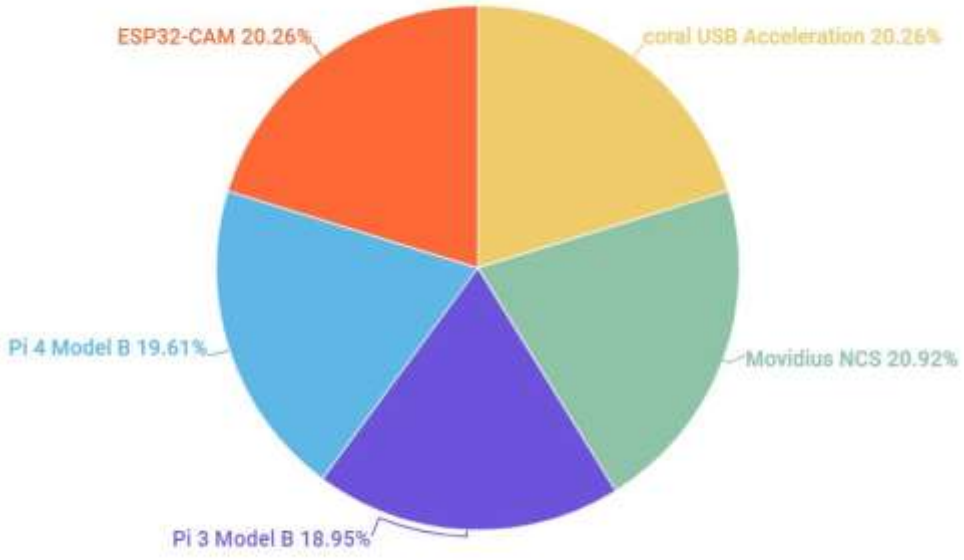
USB Hızlandırıcının dış yüzeyi 10 ° C daha sıcak çalışan Intel Neural Compute Stick 2 ile karşılaştırıldığında, çıkarım sırasında oldukça soğuk kaldığı gözlemlenmiştir. Bu gözlem sırasında Raspberry Pi CPU sıcaklığı + 9 °C daha yüksek olduğu görülmüştür.

Coral Board'un dış sıcaklığı da ölçülmektedir. Coral üzerindeki fan, CPU sıcaklığı ~ 65 ° C'ye ulaştığında dönmekte ve CPU sıcaklığını ~ 60 ° C'ye düşürmektedir. Bu, soğutucunun dış sıcaklığını ~ 50 ° C'den ~ 35'e düşürmüştür. Modelde bahsedilen fan soğutmada kullanılmıştır.

IoT cihazı için yakalanan sıcaklık değerleri Şekil 2.45'de gösterilmiştir.



(a)



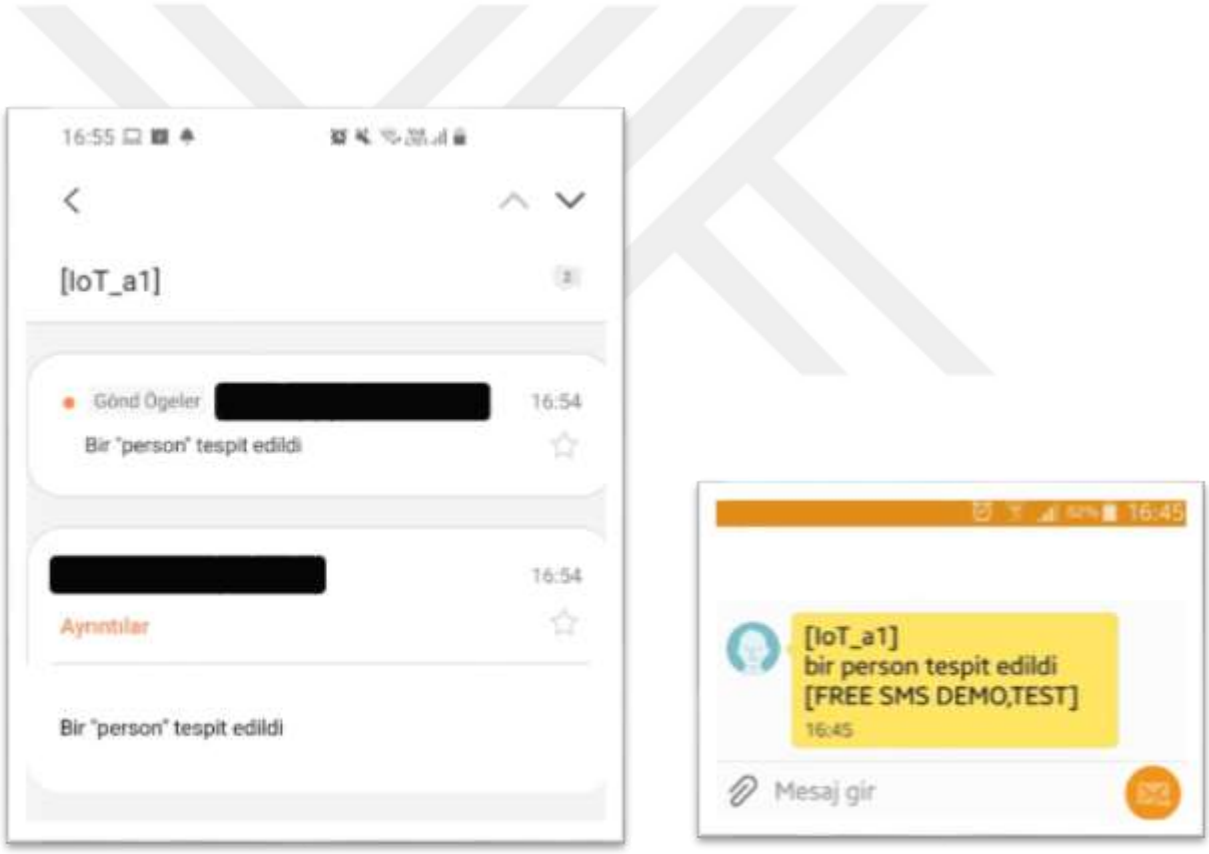
(b)

Şekil 2.45 (a) ve **(b)**, derin öğrenme modelini çalıştırırken tüm IoT cihazı için yakalanan sıcaklık değerleri

2.5.5.4. Bildirimlerin Yayınlanması

Algılama yapıldığında mesaj yayını, kamerada tespit edilen bir kişiye bağlantı kurulduğu andan itibaren yapılır. Sonuçlar değişen ışık seviyeleri, çerçeve boyutu ve nesnenin kameraya olan mesafesi ile toplanmıştır. Toplam uygulama süresi, ışık seviyesi karanlık olduğunda, ışık seviyesinin parlak olduğu duruma göre daha fazladır.

Bir kişi algılandığında kullanıcıya iki tür bildirim mesajı gönderilir. Şekil 2.46 (a), ilgi alanında tanımlanan kişiyi bilgilendirmek için alınan bir e-postanın ekran görüntüsüdür. Node-maililer kitaplığı, kullanıcılara bildirim e-postaları göndermek için kullanılır. Şekil 2.46 (b), ilgi alanında bir kişinin tespit edildiğini kullanıcıya bildirmek için SMS bildirimini gösteren mobil ekran görüntüsüdür. Nexmo API, kullanıcılara SMS bildirimleri göndermek için kullanılır.



(a) e-posta bildirimine dayalı hizmet **(b)** SMS bildirimine dayalı hizmet

Şekil 2.46 (a) ile **(b)** Kullanıcı bildirim hizmeti mesajlarının anlık görüntüleri

3. BULGULAR

Bu tez çalışması, derin öğrenme modelini kullanarak eylemi tespit eden IoT cihazı ile bu tespit esnasında yaşanan muhtemel limitlerle arasındaki ilişkiyi göstermektedir. Son zamanlarda gelişen bilgisayar görme algoritmaları, derin öğrenme ve büyük ölçekli veri kümelerini kullanmaktadır. Bunlar, büyük ölçekte donanım hızlandırmasına katkı sağlayan, karmaşık ve çok katmanlı sinir ağlarının verimli öğretimine ve çıkarımını sağlamamak için sağlam paralel hesaplama mimarilerine bağlıdır.

Donanım hızlandırmaları bilgi işlemde geleneksel CPU'larda çalışan yazılım uygulamalarına göre daha fazla verim ve daha düşük gecikme avantajına sahiptir. Geçmişte von-Neumann tipi bilgisayarların (CPU'lar) üretilmesinin başlıca nedeni aslında karmaşık görev planlamasıyla birlikte seri hesaplama yapma görevi görmekteydi. Eski CPU'lar yüksek enerji tüketiminden yola çıkarak, aynı anda yapılan yoğun hesaplamaların da neden olduğu yüksek hafıza bant genişliğini, büyük hesaplamaların tekrar yapılması konusunda oldukça sorunlar yaşanmaktaydı. Çünkü hafıza bant genişliği yetersiz kalıyordu. Çizelge 3.1'de, Derin öğrenme çalışma modelinin performansı, hassaslık ve sağlamlık açısından diğer yaklaşımlara göre karşılaştırılması göstermektedir.

IoT modellerin CPU, GPU ve RAM kalitesine göre normalde 25 frame (FPS) üreten kameranın ürettiği frameler daha fazla işlenebilmektedir, daha düşük kaliteli bir IoT modeli 2 ile 4 arası frame alabilirken daha kaliteli bir IoT model 25 frame alabilmektedir, bu sonuç Çizelge 3.1'de gösterilmiştir.

FPS (saniyede çerçevesi) numaraları her kurulum için Çizelge 3.1'de ayrıntılı olarak görülmektedir. Burada çıkan sonuçlara göre CPU, GPU ve RAM'in IoT ve embedded cihazlarıyla birlikte kullanıldığında FPS üzerindeki değişimine direk olarak etkisi ile ilgilidir. Ram ve CPU'unun birlikte çalışmasının gerekliliği aslında TensorFlow Lite'in CPU'da çalışırken, GL bölümü ile hafıza tampon özeliklerinin birlikte çalışamadığının anlaşılması bakımından önemlidir.

Dolayısıyla TensorFlow hafıza alanının boyutunu küçültebilmek için hafıza ofset hesaplama yaklaşımını kullanmaktadır.

Çizelge 3.1 Algılama Modelini Çalıştırırken Tüm IoT Cihazları İçin Performans Üzerindeki Değişimi

Ort. FPS	TFL Micro	TF		TFL		NCS Movidius		Coral TPU
ESP32-CAM	4.65	/		/		/		/
Pi 3B	/	1.37	+0.8 →	2.19	+6.0 →	8.3	+4.6 →	12.9
		⬇ +0.8		⬇ +2.6		⬇ +21		⬇ +21
Pi 4B	/	2.17	+2.6 →	4.78	+25 →	30.5	+3.9 →	34.4

Sonuçlara bakıldığında, TensorFlow'un kullandığı lite modelin MB'dan KB'ta sıkıştırılmış sınırlı ve limitli hafızanın quantization özelliği kullanılarak kaldırılmak zorunda olduğunu gösterilir.

Biz bu çalışmanın geliştirme esnasında bazı ayarların daha yüksek etkiye ve hassaslığa sebep olabileceğini fark ederek bazı önlemler aldık. Bu önlemler için:

- Öncelikle kamera pozisyonunun aldığı ışık en az 150 lux ile gündüz koşullarında çok daha kaliteli bir tespit yapması açısından gerekli olduğu kanısına varıldı.
- İkinci adımda kişisel tespitler için kullanılan cihazın kişiye mümkün olan en yakın dik açıda 3 ila 35 metre arasında olması önerilmiştir.
- Son olarak embedded cihazındaki hesaplamalar ile video büyüklüğünü düşüren etkiler, İlgili kullanıcıların alanlarına göre tanımlanmıştır.

IoT ve embedded cihazının enerji tüketimi nedeniyle, Pi 4 cihazının pi 3 ve ESP32 CAM dan daha fazla enerji tüketimi yaptığı tespit edilmiştir. Bununla ilgili olarak, ESP32-CAM cihazının daha düşük enerji tüketimi yaptığı

Çizelge 3.2'de gösterilmiştir. Bu sonuçlar, IoT cihazının çalıştığı esnada gerek duyduğu enerji ile bataryanın ömrünün kayda değer bir şekilde etkili olduğu gösterilmektedir.

Çizelge 3.2 IoT cihazları güç tüketen değerleri [98][99]

	Cihaz Durumu	**Güç Tüketimi**
Raspberry Pi 4 B	Idle	540 mA (2.7 W)
	ab -n 100 -c 10 (uncached)	1010 mA (5.1 W)
	100% CPU load	1280 mA (6.4 W)
Raspberry Pi 3 B	Idle	260 mA (1.4 W)
	ab -n 100 -c 10 (uncached)	480 mA (2.4 W)
	100% CPU load	730 mA (3.7 W)
ESP32-CAM	Idle	6 mA (0.03 W)
	Minimum	20-180 mA (0.5 W)
	Maximum	310 mA (1.6 W)

Yapılan çalışmanın desteklenmesi bakımından, IoT ve gömülü cihazlarda çalışan derin öğrenme algoritmalarının, algılama performansını değerlendirmek için bir deney yapılmıştır. Veri seti hazırlandıktan sonra, bu algoritmalar üzerine kullanılan veri seti üzerine önceden eğitilmiş bir model kullanılmıştır. Önceden eğitilmiş olan modeller, sahada toplanan test videolarında da test edilmiştir.

Çalışmanın devamında, derin öğrenme tekniklerinin kullanılarak çalışan eylem algılama modelinin fizibilitesi ve verimliliğini göstermek için IoT cihazları ve gömülü cihazlar kullanılmıştır. Akabinde kod dağıtılarak, derin öğrenme kullanılmış ve eylem algılama modelinin IoT ve gömülü cihazlar üzerindeki sınırlaması incelemeye alınmıştır. Yapılan incelemeler sonucunda yerel bir entegre işlem görme ortamında eylem algılama için bir tasarım önerilmiştir.

Sonuç olarak, IoT HW özelliklerinin (Ram, CPU ve destekleniyorsa GPU) önceki bölümde de görüldüğü üzere, derin öğrenme çalıştırma algoritmasını önemli ölçüde etkilediği gösterilmektedir. Bu etkileşim, derin öğrenme modelinin performansı bakımından da bir değişikliğe yol açabilir. IoT ve gömülü cihazlardaki çıkarımlarımıza göre, bu modeli çalıştırmanın (araştırma dâhilinde ESP32-CAM kullandık), sıkı kaynak kısıtlamaları nedeniyle zor olduğunu unutmamak gerekir. Sınırlı donanım katı güç gereksinimleri ile birlikte çalışmak zorundadır. Bu çalışmada hedeflenen amaç, IoT cihazlarında eylem algılama modelini çalıştırmak için daha iyi bir performans elde etmek için optimize edilebilen derin öğrenme destekli kütüphanelerde bulunan sınırlamaları göstermek istemekteyiz. Bunun için üç çeşit TensorFlow kullandık. Veriler, TensorFlow Lite'ın (TFLite) bellek kullanımının daha net anlaşılmasına katkıda bulunur ve sınırlı kaynaklarla IoT gömülü cihazlarda çıkarım yapmak için daha da iyi hale getirilebilir düzeydedir.

Böylece CPU kullanımı azalacak ve CPU kullanımının azalması genel pil kullanımını da azaltacaktır [26].

Derin öğrenme çıkarımı yalnızca sunucular için bir görev olarak kabul edilirken, teknolojide yaşanan son gelişmeler, çıkarım görevinin gecikmeden gizliliğe kadar çeşitli nedenlerle istenen IoT'ye ve gömülü cihazlara taşınmasına izin vermektedir. Bu cihazlar, işlem gücünün ihtiyaç duyduğu pilleri, düşük fiziksel CPU bellekleri ve önbelleği ile sınırlıdır. Böylelikle etkili bir bellek yöneticisi, uçtaki derin sinir ağı çıkarımı için oldukça önemli bir bileşen haline gelir [27]. Bellek tamponları, derin sinir ağlarındaki orta tensörler arasında akıllıca paylaşmak için TensorFlow stratejisini araştırılarak, bunları kullanmak, bellek sınırlamasının kilidini açmaya ve IoT cihazlarında bellek ayak izini optimize etmeye neden olabildiği gözlemlenmiştir. Bunun sonucunda, IoT cihazı donanım iyileştirmesinin Derin Öğrenme modelinin performansını önemli ölçüde arttıracağını göstermektedir. Öte yandan, Derin Öğrenme metodu, basit ve mikro modellerin kullanılarak HW etkisini optimize etmeye çalışıldığında, tespit sürecinin hızının iyileştirdiği önemli oranda fark edilmiştir. Yine de bu durum modelin doğruluğuna zarar vermektedir. Derin öğrenme metodunun basit modelinin kullanılması bile doğruluğa zarar verdiğinden dolayı, farklı bir yöntemle bunu minimize etmeye çalıştık. Önceki araştırmalardan yola çıkılarak, düşük maliyetli ve düşük güçlü IoT cihazlarını dikkate alarak, diğer araştırmacıların gömülü cihazları görmezden gelmesi, bizi Derin Öğrenme modeline odaklamıştır. Bu sonuçlar doğrultusunda, IoT cihazının ve Derin Öğrenme modelinin birbirinden doğrudan etkilendiği ortaya çıkmaktadır. Yaptığımız çalışmanın bulgularından yola çıkarak, IoT donanımını destekleyen ve DL modelinin doğruluğunu optimize eden mikro denetleyiciler için derin öğrenme Lite modelleri üzerinde araştırma yapılması gerektiği sonucuna varılmıştır.

İlk olarak, bu çalışmanın okuyucuları, çeşitli derin öğrenme modelleri ve IoT cihazlarıyla anlaşılması ve üzerinde çalışılması gereken IoT, Makine Öğrenimi, Derin Öğrenme ve Sinir Ağlarının çalışmasıyla ilgili bilgileri bulabilir. Okuyucular ayrıca IoT ve gömülü cihazlarda gerçek zamanlı eylem algılama ve nesne tanımayı gerçekleştirmek için en uygun derin öğrenme algoritmaları hakkında bilgi bulabilir ve TensorFlow'un derin öğrenme modelleri TensorFlow Lite ve TensorFlow Lite micro'yu önemli ölçüde daha iyi anlaşılabilir hale getirir.

IoT cihazlarında istenen algılamaları gerçekleştirmek için en önemli nokta, derin öğrenmenin önceden eğitilmiş modelinin veri kümesi hazırlanmasındaki adımları hakkında bilgi edinilebilmesidir. Uygulayıcılar, derin öğrenme algoritmalarının sınıflandırma performansını değerlendirmek için çeşitli değerlendirme ölçütlerine ilişkin bilgileri toplayabilirler. Uygulayıcılar bu araştırma ile elde edilen sonuçları kullanarak benzerliklerine bağlı olarak bu yöntemleri

çalışmalarına dâhil edebilirler. Ayrıca bu araştırmanın sınırlamalarından ders çıkarabilir ve araştırmaları için daha iyi sonuçlar elde etmeye çalışabilirler.

Genel tasarım, bir modelin bir IoT üzerinde veya bir seferde gömülü cihazlarda çalıştırılmasıyla sınırlıydı. Daha fazla eklenecek olan cihazlar sayesinde, maksimum düzeyde alanı kapsayan büyük bir resim göstermek ve geniş kullanılabilir uygulama sağlamak için daha yararlı sonuçlar doğuracaktır.

IoT'yi diğer platform hizmetlerinden, mevcut işletim sistemi sürümlerine dâhil etmek daha iyi olacaktır. Raspberry Pi 3B'de Android, google IoT platformu [28] tarafından desteklenen bir IoT cihazı olarak çalıştırdık. Makinenin gerçek zamanlı olarak video akışı sağlamadığı gözlemlenmiştir. Bununla birlikte hareketsiz bir görüntü nesnesi sınıflandırmamız da vardır. Dolayısıyla bu çalışma, gerçek zamanlı IoT video akışına dayalı bir tasarım önerdiği için, deneysel kurulumumuza dahil edilmemiştir. Diğer IoT platformlarında daha fazla araştırma yapılabilir. Bunlar ARM ve x86 / x64 cihazlarda çalışan “Microsoft10 IoT Core” ve google “android şeyler” örnek verilebilir.

HW hızlandırıcının derin öğrenme mikro modeli çalıştırma sürecinde bir faktör olup olmadığını ve mikro denetleyici derin öğrenme modellerini destekleyen gömülü HW platformlarına odaklanıp odaklanmadığını belirlemek için daha fazla araştırma yapılması gerekmektedir. Daha fazla araştırmacı, farklı uygulama alanlarında uygulanacak düşük güçlü ve düşük maliyetli IoT cihazları üzerine farklı bir çalışma yapabilir. Yapılan ilk araştırma hedefi hakkında dördüncü bölümde sunulan sonuçlardan birkaç sonuç çıkarılabilir ve geliştirilebilir. Verilerin toplandığı örneklem küçük olsa bile, araştırmacı bulduğu sonuçları, sektörün ortalama beklentilerine göre düzenli bir uygulamaya genelleştirilebilecek anlamlı bulgular ve içgörüler sağlayabilir.

4. SONUÇ VE ÖNERİLER

4.1. Sonuç

Bu tez çalışmasında, IoT ve gömülü cihazlarda eylem algılama için derin öğrenme yaklaşım modelleri incelenmiştir. Araştırma dâhilinde kullanılan sistem, derin öğrenme uygulamaları, farklı modeller (yüksek hassasiyet ve düşük gecikme hızı), değişken hedef uygulamaları ve kullanıcıların ihtiyaçlarına yönelik yerleşik bir ana yapı sağlayabilmektedir. Bu tezin ana amacı derin öğrenme yaklaşımını kullanan eylemleri tespit eden, IoT cihazının limitlerini ölçmek ve bunları gözden geçirmektir. Üç şekilde kullanılan yöntemde, IoT'yi kıyaslamak ve analiz etmek için geleneksel bulut IoT modellerinin gecikme hızı enerji tüketimi ve hassaslığı önceki bölümlerde de anlatılmıştır.

Tez çalışmasında kullanılan model ile IoT gömülü cihazlar için derin öğrenme çerçeve kitaplıklarını kullanan düşük maliyetli, düşük güçlü bir eylem algılama sistemi uygulamaları ortaya konmuştur.

Modelin etkinliğini uygulamak ve test etmek için TensorFlow platformu derin öğrenme kitaplığı kullanılmıştır. Çeşitli IoT cihazlar için model algılama hızı, algılama doğruluğu, yanıt süresi, ağ, sağlamlık ve güç tüketimi gibi farklı özellikleri karşılaştırılmıştır. Model performansının artırılması için algılama doğruluk oranı, saniyede işlenen çerçeve sayısı (FPS) ve çerçevelerin işlenebilme kapasitesinin artırılması gerektiği ortaya konmuştur. IoT cihazının CPU, GPU, RAM gibi kaynaklarının artırılması, TPU ve VPU gibi donanım hızlandırıcılarının kullanılması, standart DL yerine mini DL tekniklerinin kullanılması ile model performanslarının arttığı sonucu elde edilmiştir.

Yapılan bu çalışmanın IoT cihazlarında DL tekniği kullanacak çalışmalar için IoT cihazı seçimi ve üzerinde kullanılacak DL modellerinin seçiminde büyük fayda sağlayacağı düşünülmektedir.

4.2. Öneriler

Bundan sonra yapılacak olan çalışmalar, araştırmacının yaptığı Derin öğrenme modelini kullanan IoT gömülü cihazının kullanılarak eylemin tespit edilmesidir. Uygulanan Derin öğrenme IoT cihazları ve gömülü cihazları arasında kayda değer bir fark vardır. HW'nin kullandığı çevre koşulları, belirtilen modelin etkilerinin farklı olacağı ve performansının da değişimi konusunda etkileyici olacaktır.

Gelecekteki çalışmalar, IoT gömülü cihazlarını destekleyen mikro düzeyde derin öğrenme kontrol modellerini ve HW hızlandırıcı destekleyen çoklu IoT cihazlarını değerlendirecektir. Daha sonra da önerilen bazı dizaynları gömülü cihazları ile aynı anda entegre edilebilir.

Şebekelere Uygun Alternatif bir örnek olarak DeepIoT, Tiny YOLO, Tiny SSD ve FEATURE PYRAMİD verilebilir.

Yakın zamanda, TensorFlow kitaplığının mobil ve gömülü cihazları için hafif bir çözüm olan TensorFlow Lite piyasaya sürülmüştür. Hazırlanan bu platform, düşük bir gecikme ve küçük ikili ölçekleri de makinanın etkileşimi halinde öğrenmesini ve ayrıca bu donanımın hızlanmasını mümkün kılmıştır. Sonuç olarak bu sistemler için örneğin NVIDA JETSON TX2 çok az bir maliyet ile performansın arttırılması ile donanım sorununa çözüm odaklı sunulabilir ve bu kabul edilebilir bir değerdir.

5. KAYNAKLAR

[1] Sarkar D, Bali R, Ghosh T, 2018. Hands-On Transfer Learning with Python Implement Advanced Deep Learning and Neural Network Models Using TensorFlow and Keras.

[2] Schulz H, 2011. Pattern Recognition and Machine Learning. Relig und Konflikt. doi: 10.13109/9783666604409.185.

[3] Slater IVD, Zocca GSPRV, 2015. Python Deep Learning, 2nd ed. Packt Publishing.

[4] Wu X, Sahoo D, Hoi SCH, 2020. Recent advances in deep learning for object detection. Neurocomputing, 396:39–64.

[5] Voulodimos A, Doulamis N, Doulamis A, Protopapadakis E, 2018. Deep Learning for Computer Vision: A Brief Review. Comput Intell Neurosci. doi: 10.1155/2018/7068349.

[6] Feng X, Jiang Y, Yang X, Du M, Li X, 2019. Computer vision algorithms and hardware implementations: A survey. Integration, 69:309–320.

[7] Liu Q, Cheng L, Ozcelebi T, Murphy J, Lukkien J, 2019. Deep reinforcement learning for IoT network dynamic clustering in edge computing. Proc - 19th IEEE/ACM Int Symp Clust Cloud Grid Comput CCGrid 2019. doi: 10.1109/CCGRID.2019.00077.

[8] Aydin I, Othman NA, 2017. A new IoT combined face detection of people by using computer vision for security application. IDAP 2017 - Int Artif Intell Data Process Symp, :15–20.

[9] Slama D, Puhlmann F, Morrish J, Bhatnagar RM, 2015. Enterprise IoT : Strategies & Best Pratices for Connected Products & Services. O'Reilly Media.

[10] Xiao L, Wan X, Lu X, Zhang Y, Wu D, 2018. IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enh ance Security IEEE Signal Process Mag, 35(5):41–49.

[11] Luhach A, Singh D, Hsiung P, Hawari K, 2018. Advanced Informatics for Computing Research, 2018, Shimla, Part I.

[12] Mohammadi M, Al-Fuqaha A, Sorour S, Guizani M, 2018. Deep learning for IoT big data and streaming analytics: A survey. IEEE Commun Surv Tutorials, 20(4):2923–2960.

[13] Lin J, Chen W-M, Lin Y, Cohn J, Gan C, Han S, 2020. MCUNet: Tiny Deep Learning on IoT Devices.

[14] Deng L, Hinton G, Kingsbury B, 2013. New types of deep neural network learning for speech recognition and related applications: An overview. ICASSP, IEEE Int Conf Acoust Speech Signal Process - Proc, :8599–8603.

[15] Qi J, Yang P, Waraich A, Deng Z, Zhao Y, Yang Y, 2018. Examining sensor-based physical

activity recognition and monitoring for healthcare using Internet of Things: A systematic review. J Biomed Inform, 87(March):138–153.

[16] Mavrogiorgou A, Kiourtis A, Kyriazis D, 2019. IoT devices recognition through object detection and classification techniques. Proc 3rd World Conf Smart Trends Syst Secur Sustain WorldS4 2019, :12–20.

[17] Luo Y, Li S, Sun K, Renteria R, Choi K, 2018. Implementation of deep learning neural network for real-time object recognition in OpenCL framework. In: Proc. - Int. SoC Des. Conf. 2017, ISOCC 2017. Institute of Electrical and Electronics Engineers Inc., s:298–299.

[18] Alpaydin G, 2018. An Adaptive Deep Neural Network for Detection, Recognition of Objects with Long Range Auto Surveillance. In: Proc. - 12th IEEE Int. Conf. Semant. Comput. ICSC 2018. Institute of Electrical and Electronics Engineers Inc., s:316–317.

[19] Maturana D, Scherer S, VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition.

[20] Lewis G, 2016. Object Detection for Autonomous Vehicles. stanford .

[21] He K, Gkioxari G, Dollár P, Girshick R, 2020. Mask R-CNN. IEEE Trans Pattern Anal Mach Intell. doi: 10.1109/TPAMI.2018.2844175.

[22] Ren S, He K, Girshick R, Sun J, 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans Pattern Anal Mach Intell, 39(6):1137–1149.

[23] Kim CE, Maktab M, Oghaz D, Fajtl J, Argyriou V, 2016. A Comparison of Embedded Deep Learning Methods for Person Detection.

[24] Dai H, Khalil EB, Zhang Y, Dilkina B, Song L, 2017. Learning Combinatorial Optimization Algorithms over Graphs.

[25] Kong Y, Fu Y, 2018. Human Action Recognition and Prediction: A Survey. arXiv 13.

[26] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC, 2016. SSD: Single Shot MultiBox Detector. doi: 10.1007/978-3-319-46448-0.

[27] Redmon J, Farhadi A, 2018. YOLOv3: An Incremental Improvement. arXiv .

[28] Li H, Ota K, Dong M, 2018. Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing. IEEE Netw, 32(1):96–101.

[29] Zhang H, Zhang Z, Zhang L, Yang Y, Kang Q, Sun D, 2019. Object Tracking for a Smart City Using IoT and Edge Computing. doi: 10.3390/s19091987.

[30] Tuli S, Basumatary N, Buyya R, 2019. EdgeLens: Deep Learning based Object Detection in Integrated IoT, Fog and Cloud Computing Environments. 2019 4th Int Conf Inf Syst Comput Networks, ISCON 2019, :496–502.

[31] Mehmood F, Ullah I, Ahmad S, Kim DH, 2019. Object detection mechanism based on deep

learning algorithm using embedded IoT devices for smart home appliances control in CoT. doi: 10.1007/s12652-019-01272-8.

[32] Srinivasan K, Azhaguramyaa VR, 2019. Internet of Things (IoT) based Object Recognition Technologies. :216–220.

[33] Knezović J, Pervan B, Relja Z, Knezović J, 2019. Project Houseleek - A Case Study of Applied Object Recognition Models in Internet of Things. :1051–1055.

[34] Ke R, Zhuang Y, Pu Z, Wang Y, 2020. A Smart, Efficient, and Reliable Parking Surveillance System With Edge Artificial Intelligence on IoT Devices. IEEE Trans Intell Transp Syst, :1–13.

[35] Yao S, Zhao Y, Zhang A, Hu S, Shao H, Zhang C, Su L, Abdelzaher T, 2018. Deep Learning for the Internet of Things. Computer (Long Beach Calif), 51(5):32–41.

[36] David R, Duke J, Jain A, Reddi VJ, Jeffries N, Li J, Kreeger N, Nappier I, Natraj M, Regev S, Rhodes R, Wang T, Warden P, 2020. Tensorflow Lıte Mıcro: Embedded Machıne Learnıng On Tınyml Systems. arXiv .

[37] Lin J, Chen W-M, Lin Y, Cohn J, Gan C, Han S, 2020. MCUNet: Tiny Deep Learning on IoT Devices.

[38] Xu L Da, He W, Li S, 2014. Internet of things in industries: A survey. IEEE Trans Ind Informatics, 10(4):2233–2243.

[39] Amita Kapoor, 2019. Hands-On Artificial Intelligence for IoT.

[40] State of the IoT 2018: Number of IoT devices now at 7B – Market accelerating. https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/ (Erişim Tarihi:17/11/2020).

[41] Singh R, Gehlot A, Gupta LR, Singh B, Swain M, 2019. Internet of Things with Raspberry Pi and Arduino. Internet Things with Raspberry Pi Arduino. doi: 10.1201/9780429284564.

[42] Do We Need New Security Tools for the IoT? https://securityintelligence.com/do-we-need-new-security-tools-for-the-iot/ (Erişim Tarihi:03/01/2021).

[43] History of IoT: A Timeline of Development - IoT Tech Trends. https://www.iottechtrends.com/history-of-iot/ (Erişim Tarihi:20/11/2020).

[44] Yapay Zeka. Bir Otomasyondan Daha Fazlası | by Soner Canko | Medium. https://sonercanko.medium.com/yapay-zeka-bir-otomasyondan-daha-fazlası-8b6c4dffb4f8 (Erişim Tarihi:03/01/2021).

[45] Stewart BM, Oct R, 2020. Tiny Machine Learning : The Next AI Revolution. :1–13.

[46] Learning D, Understanding Sub-Sampling Layers Within Deep Learning. :1–4.

[47] Overview of artificial neural network. (A) In a biological neuron, the... | Download

Scientific Diagram. https://www.researchgate.net/figure/Overview-of-artificial-neural-network-A-In-a-biological-neuron-the-nucleus-transforms_fig1_325879445 (Erişim Tarihi:03/01/2021).

[48] Qingkai's Blog: Machine learning 3 - Artificial Neural Networks - part 1- Basics. http://qingkaikong.blogspot.com/2016/11/machine-learning-3-artificial-neural.html (Erişim Tarihi:03/01/2021).

[49] Burkov A, 1997. The Hundred Page Machine Learning. Computer (Long Beach Calif), 2005(April):414.

[50] Luhach A, Singh D, Hsiung P, Hawari K, 2018. Advanced Informatics for Computing Research, 2018, Shimla, Part I.

[51] Ian Goodfellow YB, Courville A, 2016. Deep Learning. Prmu, :1–10.

[52] Zhang Y, Tokmakov P, Hebert M, Schmid C, 2019. A Study on Action Detection in the Wild.

[53] Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Li FF, 2014. Large-scale video classification with convolutional neural networks. In: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. IEEE Computer Society, s:1725–1732.

[54] Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L, Large-scale Video Classification with Convolutional Neural Networks.

[55] Iwai H, Inamasu M, Totsuka T, Shimazaki T, Morita T, Takeyama S, 1983. Two-stream convolutional networks for action recognition in videos. Biochem Pharmacol, 32(5):849–855.

[56] Carreira J, Zisserman A, Com Z, Deepmind †, Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset.

[57] Donahue J, Hendricks LA, Rohrbach M, Venugopalan S, Guadarrama S, Saenko K, Darrell T, 2016. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. arXiv .

[58] Tran D, Ray J, Shou Z, Chang SF, Paluri M, 2017. ConvNet architecture search for spatiotemporal feature learning. arXiv .

[59] Tran D, Bourdev L, Fergus R, Torresani L, Paluri M, 2015. Learning Spatiotemporal Features with 3D Convolutional Networks. arXiv .

[60] Zhu Y, Lan Z, Newsam S, Hauptmann A, 2018. Hidden Two-Stream Convolutional Networks for Action Recognition. arXiv .

[61] Feichtenhofer C, Pinz A, Zisserman A, 2016. Convolutional Two-Stream Network Fusion for Video Action Recognition. arXiv .

[62] Qiu Z, Yao T, Mei T, 2017. Learning Spatio-Temporal Representation with Pseudo-3D Residual Networks *. arXiv .

[63] Asadi-Aghbolaghi M, Clapés A, Bellantonio M, Escalante HJ, Ponce-López V, Baró X, Guyon I, Kasaei S, Escalera S, Clapes A, Bellantonio M, Escalante HJ, Ponce-Lopez V, Baro X, Guyon I, Kasaei S, Escalera S, 2017. A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences. Proc - 12th IEEE Int Conf Autom Face Gesture Recognition, FG 2017 - 1st Int Work Adapt Shot Learn Gesture Underst Prod ASL4GUP 2017, Biometrics Wild, Bwild 2017, Heteroge, :476–483.

[64] Wu CY, Zaheer M, Hu H, Manmatha R, Smola AJ, Krahenbuhl P, 2018. Compressed Video Action Recognition. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit, :6026–6035.

[65] Ballard W, 2018. Hands-On Deep Learning for Images with TensorFlow : Build Intelligent Computer Vision Applications Using TensorFlow and Keras. Packt Publishing.

[66] VMware - Delivering a Digital Foundation For Businesses. https://www.vmware.com/ (Erişim Tarihi:04/01/2021).

[67] Warden P, Situnayake D, 2020. TinyML.

[68] Kim CE, Dar Oghaz MM, Fajtl J, Argyriou V, Remagnino P, 2018. A comparison of embedded deep learning methods for person detection. arXiv, :1–10.

[69] Singh H, 2019. Practical Machine Learning and Image Processing For Facial Recognition, Object Detection, and Pattern Recognition Using Python-Himanshu Singh. Apress.

[70] Gsponer D, 2018. IoT: Building a Raspberry Pi security system with facial recognition.

[71] The F, Of M, Official THE, Pi R, Projects book.

[72] ESP32-CAM and Other Cool Projects on RNT | Espressif Systems. https://www.espressif.com/en/news/ESP32_CAM (Erişim Tarihi:02/12/2020).

[73] Technology | Coral. https://coral.ai/technology/ (Erişim Tarihi:19/11/2020).

[74] Amodei D, Hernandez D, 2018. AI and Compute. Blog Open AI, :1–11.

[75] Frequently asked questions | Coral. https://coral.ai/docs/edgetpu/faq/ (Erişim Tarihi:28/11/2020).

[76] Monk S, 2017. Hacking Electronics Learning Electronics with Arduino® and Raspberry Pi. Mc Graw Hill Education.

[77] Devices LI, 2019. MASTER ' S THESIS Lightweight Edge-Based Networking Architecture for.

[78] Alina K, 2015. Object Detection Using Deep Learning - Learning where to search using visual attention.

[79] Liu JLC, Chen X, Zhou J, Tan T, Zheng N, Zha H, Hutchison D, 2018. Pattern Recognition and Computer Vision. Springer.

[80] How to Install NetBeans IDE 12 in Debian, Ubuntu and Linux Mint. https://www.tecmint.com/install-netbeans-ide-in-ubuntu-debian-linux-mint/ (Erişim Tarihi:03/12/2020).

[81] Harvey B, 2019. Algorithms and Data Structures. Comput Sci Logo Style. doi: 10.7551/mitpress/1974.003.0005.

[82] Hands-on with the Google Coral USB Accelerator - Bouvet Norge. https://www.bouvet.no/bouvet-deler/hands-on-with-the-google-coral-usb-accelerator (Erişim Tarihi:03/01/2021).

[83] MQTT - universal protocol for cloud and IoT applications | HW-group.com. https://www.hw-group.com/support/mqtt-universal-protocol-for-cloud-and-iot-applications (Erişim Tarihi:30/12/2020).

[84] Ganapathy N, Swaminathan R, Deserno TM, 2018. Deep Learning on 1-D Biosignals: a Taxonomy-based Survey. Yearb Med Inform, 27(1):98–109.

[85] Nvidia Jetson Nano tes. https://ainvidia.wordpress.com/nvidia-jetson-nano-tes/ (Erişim Tarihi:07/01/2021).

[86] Netron Uygulama | GitHub. https://netron.app/ (Erişim Tarihi:04/01/2021).

[87] Kaehler A, Bradski G, 2018. Learning OpenCV 3 Computer Vision in C++ with the OpenCV Library. O'Reilly Media.

[88] Jacob B, Kligys S, Chen B, Zhu M, Tang M, Howard A, Adam H, Kalenichenko D, 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. Google Inc.

[89] TensorFlow Lite for Microcontrollers. https://www.tensorflow.org/lite/microcontrollers (Erişim Tarihi:19/11/2020).

[90] Lite TT, Lite T, Api P, 2020. TensorFlow Lite conveter. (md):1–8.

[91] McMahan B, Rao D, 2019. Natural Language Processing with PyTorch - Build Intelligent Language Applications Using Deep Learning.

[92] Davies ER, 2017. Computer Vision: Principles, Algorithms, Applications, Learning: Fifth Edition.

[93] TensorFlow Lite converter. https://www.tensorflow.org/lite/convert/ (Erişim Tarihi:03/12/2020).

[94] Tamboli A, 2019. Build your own IoT platform : develop a fully flexible and scalable Internet of Things platform in 24 hours.

[95] Schematic illustration of the benchmark architecture. | Download Scientific Diagram. https://www.researchgate.net/figure/Schematic-illustration-of-the-benchmark-architecture_fig2_276087730 (Erişim Tarihi:17/12/2020).

[96] Antonio Guili; Amita Kapoor; Sujit Pal, 2019. Deep Learning with TensorFlow 2 and Keras: Regression, ConvNets, GANs, RNNs, NLP, and More with TensorFlow 2 and the Keras API.

[97] Illuminance - Recommended Light Level. https://www.engineeringtoolbox.com/light-level-rooms-d_708.html (Erişim Tarihi:12/12/2020).

[98] Power Consumption Benchmarks | Raspberry Pi Dramble. https://www.pidramble.com/wiki/benchmarks/power-consumption (Erişim Tarihi:19/11/2020).

[99] IoT Cihazlar Özellikleri. https://loboris.eu/ESP32/ESP32-CAM Product Specification.pdf (Erişim Tarihi:19/11/2020).

ÖZGEÇMİŞ

[redacted] yılında [redacted]'ta doğdu. İlkokulu [redacted] İlkokulu'nda, ortaokulu [redacted] Orta Okulu'nda ve liseyi [redacted] Lisesi'nde tamamladı. [redacted] yılında kazandığı [redacted] Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nden [redacted] yılında mezun oldu. [redacted] yıllarında [redacted] de Bilgi İşlem Sistemi ve Veri Merkezi Takım Lideri olarak çalıştı. 2012-2013 yıllarında [redacted]'nda Sistem Yöneticisi ve Veri Merkezi Üst Düzey Mühendis, [redacted] yıllarında [redacted]'de Bilgi İşlem Yöneticisi olarak çalıştı. [redacted] Ocak ayında [redacted] Teknik Bilimler MYO'nda [redacted] olarak göreve başladı. [redacted] de [redacted] Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı'nda yüksek lisansa başladı. Arapça, İngilizce, Türkçe, İspanyolca dillerini bilmektedir.

Halen [redacted] Üniversitesi Teknik Bilimler MYO'nda Öğretim Görevlisi olarak görev yapmaktadır.

Ahmed Yaseen Bishree AL-ANI