KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

İŞBİRLİKÇİ MİKROSERVİSLER İLE GERÇEK ZAMANLI VİDEO GÖRÜNTÜLERİ ÜZERİNDE ÇOK DEĞİŞKENLİ FİLTRELEME: AKILLI TRAFİK SİSTEMLERİ UYGULAMASI

SEDA KUL

KOCAELİ 2021

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

DOKTORA TEZİ

İŞBİRLİKÇİ MİKROSERVİSLER İLE GERÇEK ZAMANLI VİDEO GÖRÜNTÜLERİ ÜZERİNDE ÇOK DEĞİŞKENLİ FİLTRELEME: AKILLI TRAFİK SİSTEMLERİ UYGULAMASI

SEDA KUL

Prof. Dr. Ahmet SAYAR		
Danışman,	**Kocaeli Üniversitesi**	**.....**
Prof. Dr. Yaşar BECERİKLİ		
Jüri Üyesi,	**Kocaeli Üniversitesi**	**.....**
Doç. Dr. Mehmet Sıddık AKTAŞ		
Jüri Üyesi,	**Yıldız Teknik Üniversitesi**	**.....**
Dr. Öğr. Üyesi Orhan AKBULUT		
Jüri Üyesi,	**Kocaeli Üniversitesi**	**.....**
Dr. Öğr. Üyesi Seçkin Arı		
Jüri Üyesi,	**Sakarya Üniversitesi**	**.....**

Tezin Savunulduğu Tarih: 25.11.2021

Etik Beyan Ve Araştırma Fonu Desteği

Kocaeli Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez/proje çalışmasında,

- Bu tezin/projenin bana ait, özgün bir çalışma olduğunu,
- Çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı,
- Bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi,
- Bu çalışmanın Kocaeli Üniversitesi'nin abone olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü'nün belirlemiş olduğu ölçütlere uygun olduğunu,
- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- Tezin/Projenin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez/proje çalışması olarak sunmadığımı,

beyan ederim.

☐ Bu tez/proje çalışmasının herhangi bir aşaması hiçbir kurum/kuruluş tarafından maddi/alt yapı desteği ile desteklenmemiştir.

☒ Bu tez/proje çalışması kapsamında üretilen veri ve bilgiler TÜBİTAK tarafından 116E202 no'lu proje kapsamında maddi/alt yapı desteği alınarak gerçekleştirilmiştir

Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

.....
(İmza)

Seda KUL

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI

Fen Bilimleri Enstitüsü tarafından onaylanan lisansüstü tezimin/projemin tamamını veya herhangi bir kısmını, basılı ve elektronik formatta arşivleme ve aşağıda belirtilen koşullarla kullanıma açma izninin Kocaeli Üniversitesi'ne verdiğimi beyan ederim. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin/projemin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanımı bana ait olacaktır. Tezin/projenin kendi özgün çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin/projenin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alarak kullandığımı ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim kurulu tarafından yayınlanan "***Lisanüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge***" kapsamında tezim aşağıda belirtilen koşullar haricinde YÖK Ulusal Tez Merkezi/ Kocaeli Üniversitesi Kütüphaneleri Açık Erişim Sisteminde erişime açılır.

☐Enstitü yönetim kurulu kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 2 yıl ertelenmiştir.
☐Enstitü yönetim kurulu gerekçeli kararı ile tezimin/projemin erişime açılması mezuniyet tarihinden itibaren 6 ay ertelenmiştir.
☒Tezim/projem ile ilgili gizlilik kararı verilmemiştir.

.....
(İmza)

Seda KUL

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışmasında, araçların sınıflandırılması ve sınıflandırılmış görüntülerin yayınla/kaydol sistemi ile istemcilere gönderilmesi amaçlanmaktadır. Sisteme kayıtlı kullanıcılara istedikleri özelliklere sahip araçlara ait bilgi/görüntülerin gerçek zamanlı olarak aktarılması gerçekleştirilmeye çalışılmıştır.

Tez çalışmamda desteğini esirgemeyen, çalışmalarıma yön veren, ve bana güvenen danışmanım saygı değer hocam Prof. Dr. Ahmet SAYAR'a sonsuz teşekkürlerimi sunarım.

Bu doktora tez çalışmasını destekleyen Türkiye Bilimsel ve Teknolojik Araştırma Kurumu'na (TÜBİTAK) (Proje No: 116E202) teşekkürlerimi sunarım.

Son olarak aileme özel minnettarlığımı ifade etmek istiyorum. Hayatım boyunca tüm zorlukları benimle göğüsleyen ve hayatımın her evresinde bana destek olan, bana güç veren en büyük destekçilerim sevgili annem Aydan KUL ve babam İsmail KUL'a sonsuz minnet duygularımı sunarım.

Kasım – 2021 Seda KUL

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

x	: Girdi
n	: n girdi sayısı
W	: Ağırlık
y	: Hedef Değerlerin Vektörü
$\hat{y}$	: Tahmin Değerlerinin Vektörü

Kısaltmalar

3DES	:Triple Data Encryption Standart (Veri Şifreleme Standardı)
AES	:Advanced Encryption Standard (Gelişmiş Şifreleme Standardı)
ANN	:Artificial neural network (Yapay Sinir Ağı)
ARM	:Acorn RISC Machine (RISC Tabanlı İşlemci Mimarisi)
CBIR	:Content Based Information Retrival (İçeriğe Dayalı Bilgi Erişimi)
ConNN	:Convolutional Neural Network (Konvolüsyonel Sinir Ağları)
CUDA	:Compute Unified Device Architecture (Hesaplanmış Birleşik Cihaz Mimarisi)
DES	:Data Encryption Standart (Veri Şifreleme Standardı)
DNN	:Deep Neural Network (Derin Sinir Ağları)
FPS	:Frames per second (Saniyelik Görüntü Sayısı)
HOG	:Histogram of oriented gradients (Yönlü Gradyanlar Histogramı)
HTML	:Hyper Text Markup Language (Hiper Metin İşaret Dili)
IDE	:Integrated Development Environment (Bütünleşik Geliştirme Ortamı)
IR	:Information Retrival (Bilgi Erişimi)
RC4	:Rivest Cipher 4
RSA	:Rivest–Shamir–Adleman
SBIR	:Semantic Based Information Retrival (Semantik Temelli Bilgi Erişimi)
SVM	:Support vector machines (Destek Vektör Makineleri)
TBIR	:Text Based Information Retrival (Metin Tabanlı Bilgi Erişimi)

İŞBİRLİKÇİ MİKROSERVİSLER İLE GERÇEK ZAMANLI VİDEO GÖRÜNTÜLERİ ÜZERİNDE ÇOK DEĞİŞKENLİ FİLTRELEME: AKILLI TRAFİK SİSTEMLERİ UYGULAMASI

ÖZET

Bu tez çalışmasında, video akışları üzerinden belirli bir aracı izlemeye yönelik işbirlik içinde çalışan yeni bir mikroservis mimarisi önerilmektedir. Yayınla/abone ol modeline dayalı olarak anahtar sorgusu ile video parçalarını alma problemi incelenmiştir. Sunulan çerçevede verinin işlenmesi ve iletilmesi için katmanlı mimari yapısı geliştirilmiştir, (i) Veri toplama katmanında: gerçek zamanlı olarak veri almak ve iletmek için yayınla/abone ol paradiması kullanılır. (ii) Sınıflandırıcı katmanında: mikro hizmetler, çevrim dışı olarak eğitilmiş yapay zeka modellerine göre görüntülerde filtreleme işlemlerini gerçekleştirir ve her özelliği birer olay/konu olarak Apache Kafka topik olarak yayınlar. (iii) İşbirlikçi-Sınıflandırıcı katmanında: Mikro hizmetler otomatik olarak kendi kendine bu topiklere abone olarak, topiklerin ikili kombinasyonunu oluşturur. (iv) Son katman olan Kompleks-İşbirlikçi-Sınıflandırıcı katmanında bulunan tüm mikro hizmetler tüm olasılıkları bir araya getirerek daha fazla topiğin yayınlanmasını sağlarlar.

Tüm bu senaryolar aynı zamanda güvenliğin önemini de vurgulamaktadır. Kriptografi, verinin korunmasında bilginin şifrelendiği bir yöntemdir. Kriptografik algoritmalar çeşitli şekillerde kullanılabilir. İdeal bir dünyada, düşük maliyetli, yüksek performanslı bir şifreleme yöntemine ihtiyaç duyulmaktadır. Yaygın olarak kullanılan Triple Data Encryption Algorithm (3DES), Advanced Encryption Standard (AES), Blowfish, CAST5, Rivest Cipher 4 (RC4) ve Rivest–Shamir–Adleman (RSA) kriptografik algoritmalarının, maliyet ve performansları mesaj ve anahtar uzunluğuna göre analiz edilerek yayınla/abone ol iletişim yöntemini güvence altına almak için uygulandı.

Son olarak, çözümümüz ara katman yazılımından bağımsız olarak uygulanır ve bu çerçeve popüler içerik tabanlı ağ uygulaması üzerinden güvenli bir şekilde gerçekleştirilir. Performans analizi, çözümümüzün ölçeklenebilir olduğunu göstermektedir.

Anahtar Kelimeler: Akıllı Trafik Sistemleri, Dağıtık Sistemler, Görüntü İşleme, Kriptoloji, Yapay Zeka.

MULTIVARIABLE FILTERING ON REAL-TIME VIDEO IMAGES WITH COLLABORATIVE MICROSERVICES: SMART TRAFFIC SYSTEMS APPLICATION

ABSTRACT

In this thesis, a new collaborative microservice architecture is proposed to monitor a specific vehicle over video streams. Based on the publish/subscribe model, the problem of retrieving video segments with key query is investigated. Layered architecture has been developed for processing and transmitting data in the presented framework, (i) In the data collection layer: publish/subscribe paradigm is used to receive and transmit data in real time. (ii) At the Classifier layer: microservices filter images based on artificial intelligence models trained offline and publish each feature topically in Apache Kafka as an event/topic. (iii) At the Merger-Classifier layer: Microservices automatically subscribe to these topics, creating binary combinations of topics. (iv) All microservices in the last layer, the Complex-Merger-Classifier layer, combine all possibilities and enable more topics to be published.

All these scenarios also highlight the importance of security. Cryptography is a method in which information is encrypted to protect data. Cryptographic algorithms can be used in a variety of ways. In an ideal world, a low-cost, high-performance encryption method is needed. The cost and performance of the widely used Triple Data Encryption Algorithm (3DES), Advanced Encryption Standard (AES), Blowfish, CAST5, Rivest Cipher 4 (RC4) and Rivest–Shamir–Adleman (RSA) cryptographic algorithms were analyzed by message and key length. implemented to secure the publish/subscribe communication method.

Finally, our solution is implemented independently of the middleware, and this framework is implemented securely over the popular content-based network application. Performance analysis shows that our solution is scalable.

Keywords: Intelligent Traffic Systems, Distributed Systems, Image Processing, Cryptology, Artificial Intelligence.

1. GİRİŞ

Akıllı şehir, yaşam kalitesini artırmak için daha iyi bir şehir inşa edilmesini prensip alan bir kavramdır. Akıllı bir şehir, şehir sorununun anlaşılmasını kolaylaştıracak ve onları çözmek için kesin çözümü bilen bir teknolojiye ihtiyaç duyar. Akıllı şehir mimarisinin arkasında çalışan birçok teknoloji vardır bunlar: büyük veri, yapay zeka, gerçek zamanlı analitik vb. Akıllı Ulaşım Sistemi (AUS), akıllı şehir için önemli özelliklerden biridir. Var olan trafik yönetimi ve uyarı sistemleri AUS ihtiyaçlarına tam olarak cevap verememektedir. Trafik yönetimi için daha iyi hizmet sağlamak maliyetlidir ve yapılandırılabilirliği yüksektir.

Akıllı Ulaşım Sistemlerinin, trafik akışını düzgün bir şekilde işleyebilmesi için çeşitli donanımsal ve yazılımsal teknolojiler yardımıyla trafikle ilgili bilgileri elde etmeleri gerekmektedir. Araç tespiti ve sınıflandırma sistemi özellikle AUS'te önemli bir araştırma alanına sahiptir (Kul ve diğ., 2016). Genel olarak araç tespit yöntemleri donanım tabanlı ve yazılım tabanlı olarak ikiye ayrılabilir. Donanım tabanlı araç tespit yöntemlerinin uygulanmasında radar, kızılötesi dedektörleri veya mikrodalga gibi donanımlar kullanılır (Fang ve diğ., 2007). Detektörlerin boyutlarının büyük, kurulumunun pahalı ve elde edilen bilginin sınırlı olması donanım tabanlı sistemlerin en büyük dezavantajlarıdır.

Yazılım tabanlı geliştirilen araç sınıflandırma yöntemlerinin donanım tabanlı sistemlere karşı, kolay kurulum yapabilme, çevre dostu olma ve düşük maliyetli olması gibi birçok avantajı bulunmaktadır. Düşük gerçek zamanlı işlem yapma gibi sorunlarına karşılık donanım tabanlı sistemlere göre çok daha iyi sonuçlar vermektedir (Wu ve diğ., 20011). Bu nedenle Kapalı devre televizyon (CCTV) ve IP kameralar gibi gözetleme kamera sistemleri yaygınlaşmış ve aktif olarak kullanılmaya başlanmıştır. Buna bağlı olarak Akıllı Ulaşım Sistemlerinde çalışan operatörlerin, güvenlik kamerası beslemelerini gerçek zamanlı olarak veya video kayıttan izlemeleri gerekmektedir. Bu durum da beraberinde insan hatasına bağlı problemleri ortaya çıkarmaktadır.

Güncel, doğru ve ilgili verilere erişim ihtiyacı birçok durum için gereklidir. Bu nedenle, canlı video izleme akışlarının kullanıcılara ilgi alanlarına göre hızlı bir şekilde nasıl aktarılacağı ve dağıtılacağı en büyük zorluklardan biridir. Verimli tarama, arama ve erişime izin verecek şekilde çalışacak sistemlere ihtiyaç bulunmaktadır. Ana sorunlardan biri, geniş ve çeşitli bir koleksiyonda istenen görüntüyü bulmanın zorluğudur. Küçük bir koleksiyondan istenen görüntüyü insan gözü ile bulmak kolay olsa da, binlerce öğeyi içeren koleksiyonlarda daha etkili tekniklere ihtiyaç vardır (Hui ve diğ., 2010). Görüntü erişimi, 1970'lerden beri veri tabanı yönetimi ve bilgisayarla görü alanında aktif bir araştırma konusudur (Rui ve diğ.,. 1999). Bir görsel veritabanından görsel içeriklere bakma, tarama ve alma tekniğidir. Web kameralarının, dijital kameraların ve bu tür cihazlarla donatılmış cep telefonlarının icadı nedeniyle dijital kütüphanelerin hızlı ve patlayıcı büyümesi, insan açıklamalarıyla veritabanı yönetimini yetersiz hale getirmektedir. Görüntü indeksleme ve alma, mevcut senaryoda daha fazla dikkat çeken çok önemli bir araştırma konusudur. Dolayısıyla mevcut durumda, tam ve hızlı görüntü alımı gerekli hale gelmektedir.

Mevcut görüntü sorgulama çalışmaları üç kategoride sınıflandırılabilir: metin tabanlı görüntü erişimi (TBIR), içerik tabanlı görüntü erişimi (CBIR) ve anlamsal tabanlı görüntü erişimi (SBIR). Metin Tabanlı Görüntü Erişim (TBIR) yöntemi, görüntülere dosya adı, görüntü boyutu ve biçimi ile açıklamalar ekleyerek bunları bir veritabanında saklar (Chang ve diğ., 1992). TBIR'nin ilk dezavantajı, kapsamlı bir veritabanı için manuel olarak meta veri oluşturmanın pratik olmamasıdır. İkinci dezavantajı da, hem metin altyazıları hem de video verileri için doğru bir temsil bulmanın zorluğudur. Altyazıları kodlamak ve metni temsil etmek için çeşitli yöntemler (Tamura ve diğ., 1984; Mailaivasan ve diğ., 2014; Kata 1992; Singha ve diğ., 2012; Mitra ve diğ., 2005) bulunmaktadır fakat açıklamayı yazan kişiden kişiye bu bilgiler değişiklik gösterebilir. Ayrıca, metinsel açıklamalar dile bağlı olacaktır (Chauhan ve diğ., 2013) ve farklı diller için çalışmayacaktır.

İçerik Tabanlı görüntü erişim (CBIR) yöntemi, görüntüye göre sorgulama olarak da bilinir. CBIR yönteminde sistem, her bir görüntüden öznitelikler çıkarır ve bunları bir veri tabanında saklar. Kullanıcı bir görüntüyü sorguladığında, sorgu görüntüsünün özellik vektörü çıkarılır. Sorgu görüntüsünün öznitelik vektörü ile veritabanındaki görüntüler

arasındaki benzerlik ölçülür. Ardından sistem, sorgu görüntüsüne en çok benzeyen görüntüleri (Rui ve diğ., 1999) döndürür. Ancak bu, yüksek boyutlu vektörlerle çalışılması nedeniyle gerçek zamanlı (Hui ve diğ., 2010) çalışan sistemler için yavaş kalmaktadır.

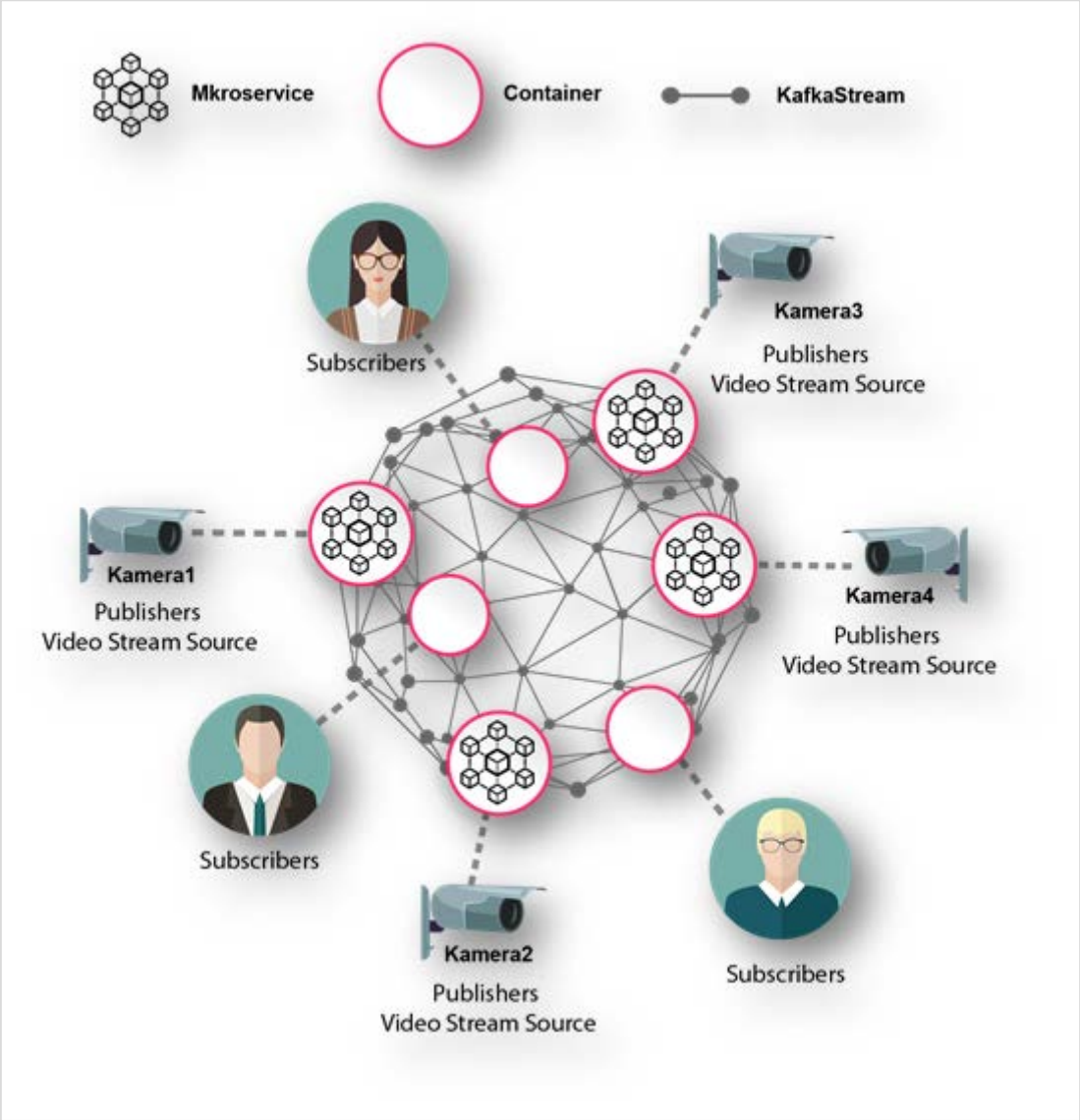
SBIR, CBIR ve TBIR'in birleşimidir. Semantik tabanlı görüntü erişiminde, görüntü, duygu, soyut nitelik, mantıksal çıkarım, şekil eşleştirme, yüz, parmak izi gibi özellikler yardımıyla görüntü erişimi yapılır. Daha iyi sonuçlar elde etmek için, birçok CBIR sistemi genellikle, görüntü ile anlamsal olarak daha benzer olan üst düzey kavramları tanımlamak için doku, renk ve şekil gibi alt düzey özelliklerin arayüzlerle birleşimini kullanır. Bu teknik, düşük seviyeli kavramları yüksek seviyeli kavramlara eşleyen ve anlamlı bilgilerin alınmasına yardımcı olan semantik boşluğu doldurmak olarak tanımlanır (Hollink ve diğ., 2005).

Bu nedenlerden dolayı, geleneksel bir akış veri işleme çerçevesi kullanarak yerel donanımlarda ya da uzak sunucularda akan veriyi işlemek ve istenilen metin sorgusuyla video parçalarını alan bir trafik izleme sisteminin gerçek zamanlı çalışması zordur. Bu tez çalışmasında, trafik gözetleme kameraları tarafından yayınlanan gerçek zamanlı video akışları üzerinden belirli bir aracı izlemeye yönelik işbirlik içinde çalışan yeni bir mikroservis mimarisi önerilmektedir. Yayınla/abone ol paradigmasına dayalı olarak anahtar/değer sorgusu ile video parçalarına erişim mimarisi gerçekleştirilmiştir.

İşbirlikçi mikroservisler ile gerçek zamanlı video görüntüleri üzerinde çok değişkenli filtreleme: Akıllı Trafik Sistemleri Uygulaması geliştirilmiştir. Şekil 1'de verilen görselde gösterildiği gibi sistem mimarisi eklenen her bir kameradan görüntü alabilecek, bu görüntüleri araçları tespit eden bir mikroservise gönderecektir. Nesnenin imge görüntüleri üzerinden çıkarılabilecek özellikler üzerinden "ve", "veya" şeklinde kompleks, bütünleşik, birleşik sorgular oluşturmak ve bu sorguları akan veri üzerinde gerçek zamanlı olarak sorgu tabanlı filtreleyerek sonucu, isteyen kayıtlı kullanıcılara yönlendirme gerçekleştirilmiştir.

Kullanıcı araç tipini ve/veya rengini ve/veya hızını seçtiğinde sistem otomatik olarak ilgili konuya abone olur. Böylece operatör videonun tamamını izlemek yerine sorgusuna göre sadece ilgili kısmı inceler.

Konular filtre görevi görür ve her filtre, araçların fiziksel özelliklerine (tür, renk ve hız) göre belirlenir.



Şekil 1.1. Dağıtık Sistem Mimarisi

Yayınla/abone ol mimarisi, canlı video gözetim akışlarını kullanıcıların ilgi alanlarına göre anında filtrelemek ve dağıtmak için kullanılmıştır. Bir yayınla/abone ol yöntemi, ilgili abonelere olayları iletmekten sorumlu olan bir mesajlaşma paradigmasıdır (Eugster ve diğ., 2003). Mikroservis mimarisinin ve Apache Kafka'nın avantajları açıktır, ancak yüksek profilli saldırılar, en iyi güvenlik uygulamaları ve ilgili maliyetleri hakkındaki belirsizliklerle birleştiğinde, birçok güvenlik açığı ortaya çıkabilmektedir. En önemlisi, yayınlama/abone olma sistemlerinde bulunan merkezi mesaj aracıları, tüm iletişim

mesajlarına erişebildikleri için güvenlik için belirlenmiş bir zayıf nokta sunar. Bu nedenle sunulan özellik-anahtar kelime tabanlı veri yayınla/abone ol şemasının gizliliği koruyan bir sistem olması önemlidir. Spesifik olarak, yayınlanan verilerin gizliliğini sisteme abone olmayanlara karşı korumak, yayıncıların veri erişimini kendi başlarına kontrol edebilmeleri adına yayınlanan verileri şifreleme gerçekleştirilmiştir ve bunun için mevcut simetrik ve asimetrik şifreleme yöntemleri kullanılmıştır.

Ekosistemin bir servisteki güvenlik açığından dolayı tüm sistemin güvenliğini tehlikeye atmasını önlemede en önemli pay kriptografiye aittir. Günümüz dünyasında kriptografi bilgi güvenliğinin hayati bir parçasıdır (Diffie ve diğ., 1976). Kriptografi, özellikle iletim ve depolama için dijital verileri istenmeyen erişimlere karşı anlaşılmaz hale getirme bilimidir. Kriptografinin temel bir işlevi olan şifreleme/şifre çözme, bilgiyi gizli tutmanın yaygın olarak kullanılan bir yoludur. Temel iletişim (düz metin), şifreleme sırasında şifreli metin olarak bilinen okunamaz bir forma dönüşür. Şifre çözme (düz metin) sırasında bir şifreli metin orijinal metne dönüştürülür. İki tür kriptografi vardır: simetrik ve asimetrik (Mikhail ve diğ.,. 2014). Simetrik anahtar şifrelemesi, verileri yalnızca gönderen ve alıcının bildiği gizli bir anahtar kullanarak şifreleyerek korur. Asimetrik anahtar şifreleme ise verileri korumak için genel ve özel anahtarları kullanır. Özel anahtar bir kişi tarafından tutulur ve başka kimseyle paylaşılmaz. Açık anahtar ise herkes tarafından bilinir.

Kriptografi tekniklerinin hesaplama süresi üç kategoriye ayrılır: şifreleme/şifre çözme, anahtar oluşturma ve anahtar değişimi (Shetty ve diğ., 2014). Bir anahtar oluşturmak için gereken süre, simetrik ve asimetrik şifreleme arasında farklılık gösteren anahtarın uzunluğuna göre belirlenir. Bir anahtarın değiş tokuş edilmesi için geçen süre, gönderici ve alıcı arasındaki iletişim kanalı tarafından belirlenir (Jincharadze ve diğ., 2017). Bu nedenle, tez çalışmasında, güvenli yayınla/abone ol iletişimi (URL-1, 2021) için 3DES, AES, Blowfish, CAST5, RC4 ve RSA gibi çeşitli simetrik ve asimetrik şifreleme/şifre çözme şemaları hız ve verimlilik gibi çeşitli faktörlere göre karşılaştırıldı. En iyi performansı gösteren algoritma ile kanal şifrelenerek güvenli bir yayınla/abone ol akışı elde edildi.

Bu tez çalışmasında, bir mikroservis mimarisinde güvenli yayınla/abone ol modelinin nasıl uygulanabileceğini gösterme hedeflenmiştir. Akıllı Trafik Sisteminde belirlenen

özellik filtrelerine göre olayları üretmek ve tüketmek için yayınla/abone ol modelini uygulayan bir araç olan Apache Kafka mesajlaşma sistemine ve bunun mikroservis mimarisiyle nasıl bütünleştiğine odaklanılmıştır.

Bu tez çalışmasının hedefleri ve katkıları aşağıdaki gibi özetlenmiştir:

- Mesajlaşma sistemlerinin üzerine inşa edilen yayınla/abone ol modeline ait inceleme verilmiştir.
- Yayınla/abone ol modelinin kurulmasında temel olan mikroservis mimarisine ait inceleme verilmiştir.
- Mikroservis mimarisinde kullanılan ve önemli bir bileşen olan Docker'ın kullanımı gösterilmiştir.
- Apache Kafka'yı yayınlama ve abone olma mesajlaşma aracı olarak tanıtmak ve bunun mikroservis mimarisine nasıl uyguladığını, değer kattığı ve tamamladığı gösterilmiştir.
- Mikroservis tabanlı, birden fazla yayıncının veri erişimini kontrol etmesine, birden fazla abonenin seçici olarak veri almasını sağlayan ve bunlar gizliliği koruyan bir özellik tabanlı veri yayınla/abone ol şeması dahilinde önerilmiştir.
- Hem yayınlama hem de abone olma modelini ve mikroservis mimarisini kullanan bir kullanım senaryosu sağlanmıştır.
- Gerçek zamanlı video akışları üzerinden yapılan araç tespiti, dağıtılmış birçok hizmetin iş birliği ile elde edilir ve bu hizmetlerin görevlerine göre gruplandırılması tanımlanır. Bazı hizmetlerin kameralardan ham akışlar alması ve özellik tabanlı filtreleme gerçekleştirmesi, bazı filtrelerin de var olan filtrelerin kombinasyonları şeklinde çalışmasına yönelik protokolleri belirlenmiştir.
- Mikroservis mimarisinin herhangi bir yayınla/abone ol ilkesi kontrol prosedürü, erişim ilkesini ve abonelik ilkesini birbirine bağlamak için yeni haberleşme protokolleri oluşturulmuştur.
- Olayların yayınlanmasında kullanılmak üzere akan veriye ait veri yapısı tasarlandı ve kuralları oluşturuldu.
- Yayınla/abone ol mimarisinin iletişim güvenliği kriptografik algoritmalar ile güvenli hale getirildi.

Tez, uç noktaların mikro hizmetler gibi yazılım geliştirme modellerini kullanarak birbirlerinden haberdar olmadan verileri yapay zekâ tabanlı filtrelerden geçirerek gönderebilmeleri ve bunun sonucunda büyük hacimli verilerin üretilmesi ve tüketilmesi gerçeğiyle motive edilmektedir. Ayrıca, mikroservis mimarisini kullanarak sistemler tasarlayan veya tasarlamayı planlayan ve modern mesajlaşma sistemlerinin faydalarından yararlanmak isteyen sistem mimarları, görevi büyük miktarda veri toplamak olan veri analistleri analiz için çeşitli kaynaklardan ve mikroservis mimarisi ile ilgili olarak yazılım geliştirme alanını ilerletmek ile ilgilenen diğer tüm çalışmalara katkı olması amaçlanmıştır.

Nesnenin imge görüntüleri üzerinden çıkartılabilecek özellikler üzerinden “ve", "veya” şeklinde kompleks, bütünleşik, birleşik sorgular oluşturmak ve bu sorguları akan veri üzerinde gerçek zamanlı olarak sorgu tabanlı filtreleyerek sonucu, isteyen kayıtlı kullanıcılara yönelendirme çerçevesi sunulmaktadır.

Kullanıcı araç tipini ve/veya rengini ve/veya hızını seçtiğinde sistem otomatik olarak ilgili konuya abone olur. Böylece operatör videonun tamamını izlemek yerine sorgusuna göre sadece ilgili kısmı inceler. Tez çalışmasının araştırma kapsamında incelemiş olduğu problemler şu şekilde sıralanabilir:

- Problem 1: Yayınla/Abone ol haberleşme mimarisinin analizinin gerçeklenmesi
- Problem 2: Aracı tipine göre sınıflandırılma
- Problem 3: Aracı rengine göre sınıflandırılma
- Problem 4: Aracın hızını tespit etme
- Problem 5: Mikroservis mimarisini oluşturma
- Problem 6: Mikroservislerin haberleşmesini sağlama ve hataya dayanıklı hale getirme
- Problem 7: Haberleşme kanalını güvenli yapma

Tezin bölüm bölüm organizasyonu şu şekildedir:

- Bölüm 2, Tez kapsamı dahilinde ilgili çalışmaları içerir. Kapsam dahilinde akıllı trafik sistemleri, görüntü erişim teknikleri, yapay zeka sınıflandırıcıları, mikroservis mimarisi kullanımı, yayınla/abone ol haberleşme kanalının kullanımı ve güvenli veri iletimi ile ilgili konular alt başlıklar olarak verir.

- Bölüm 3, tez kapsamında kullanılan veri setleri verilmiştir ve veri etiketleme yöntemi açıklar.
- Bölüm 4, Yapay zeka sınıflandırıcılarını, mikroservis mimari tasarımını, mimarinin konteynerlaştırılmasını, iletişim kanalında kullanılacak Apache Kafka'nın açıklanmasını, servisler arasında el sıkışma mekanizmasının açıklanmasını, akan veriye ait veri yapısının tasarlanmasını, mesajların özelliklere göre kombinasyonlar halinde birleştirilmesini ve birleştirme kurallarını, verinin güvenli iletimini açıklar.
- Bölüm 5, araştırma sonuçlarını ve performans analizlerini verir.
- Bölüm 6, tez çalışmasının neticesi ve gelecek çalışmalar hakkında bilgi verir.

2. LİTERATÜR TARAMASI

Bu bölüm, bu tezin temelini oluşturan temel kavramların yeterli bir arka planını sağlamaya çalışmaktadır. Gerçek zamanlı video akışları üzerinden metin tabanlı sorgular yapabilen, dağıtık ve işbirlikçi bilgi işlem ortamı bulunmaktadır. Gerçek zamanlı video akışları üzerinden yapılan filtreleme işlemleri, birçok dağıtık servisin iş birliği ile elde edilir ve bu servisler görevlerine göre gruplandırılır. İşbirlikçi filtreler arasındaki tüm mesajlaşma, kuyruk yapılarını kullanarak kalıcı iletişimi sağlayan, yayınla-abone ol mesajlaşma paradigması ile yapılır. Bu yaklaşım, gerçek zamanlı video izlemeyi ölçeklenebilir ve cevap verebilir hale getirir. Aynı zamanda yayınla/abone ol paradigması üzerinde veriler şifrelenerek güvenli iletişim gerçekleştirilmiştir.

Bu tezin kapsadığı çalışma alanları çeşitli araştırma alanlarını içermektedir ve dört ana sınıfta toplanabilir:

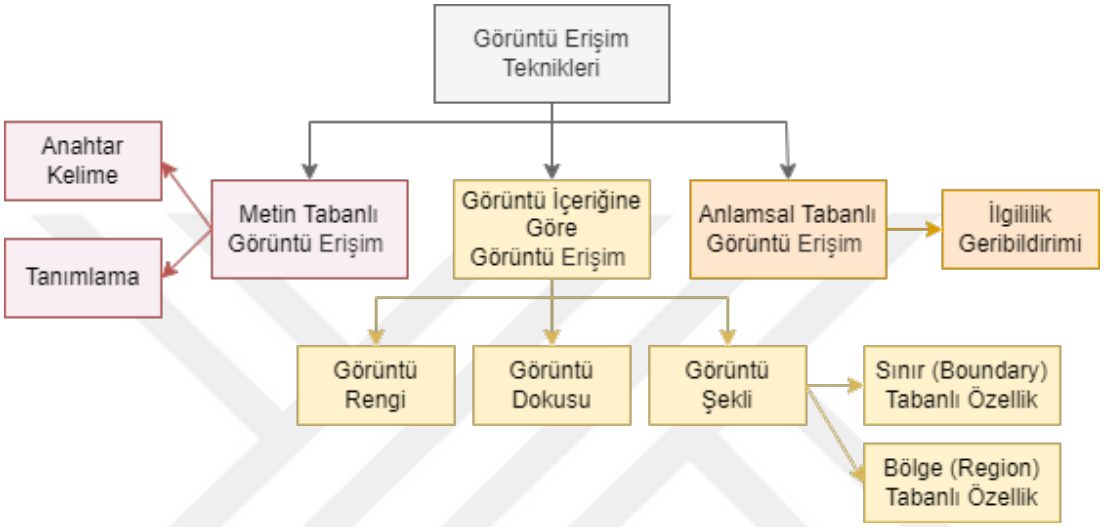
- Görüntü erişim teknikleri
- Görüntü erişim tekniklerinde kullanılan Yapay Zeka sınıflandırıcıları: Araç Tip, Renk ve Hız Tespit Etme çalışmaları
- Mikroservis Mimarisi ve Yayınla/Abone Ol haberleşme kanalının kullanımı
- Yayınla/abone ol haberleşmesinin güvenliği

Bu bölümün geri kalanında üç ana sınıfa toplanan başlık üzerine literatürde bulunan güncel çalışmalar incelenerek çalışmalar tüm yönleri ile ele alınmıştır. Çalışmaların literatürdeki açığı saptanmış ve açıklanmıştır.

2.1. Görüntü Sorgulama Teknikleri

Görüntü sorgulama sistemleri, dijital görüntülerin hızla büyümesi nedeniyle verimli ve etkili bir tekniğe ihtiyaç duymaktadır. Görüntü alımı, özellikle içerik tabanlı görüntü alımında (CBIR) kapsamlı bir araştırma alanı olarak kabul edilir. CBIR, son zamanlarda çok aktif bir araştırma alanı olan görüntü özelliklerine dayalı büyük görüntü veri tabanından benzer görüntüleri alır. Renk, doku, şekil vb. gibi görüntüden türetilebilen içeriğe özellik denir. Bu bölümde farklı tipteki görüntü alma (IR) sistemlerinin güncel literatür çalışmaları verilmiştir.

Görüntü alımı, 1970'lerden bu yana veri tabanı yönetimi ve bilgisayarla görme (Rui ve diğ., 1999) alanlarında aktif bir araştırma alanı olmuştur. Bu nedenle, görüntü sorgulama, bir görüntü veritabanında; görüntüleri arama görevi olarak tanımlanabilir. Şekil 2.1'de gösterildiği gibi, görüntü sorgulama teknikleri üç kategoride sınıflandırılabilir: metin tabanlı görüntü sorgulama (TBIR), içerik tabanlı görüntü alımı (CBIR) ve anlamsal tabanlı görüntü alımı (SBIR).

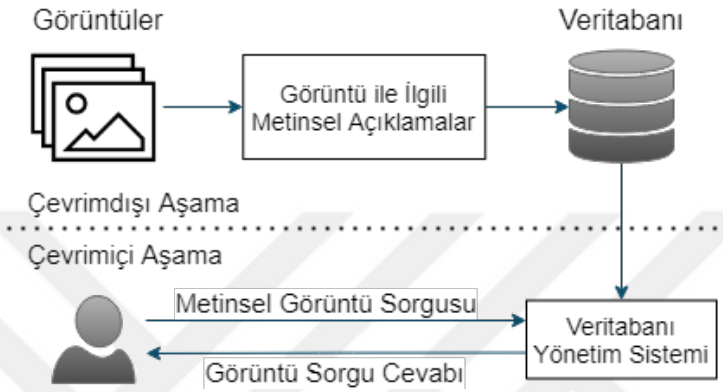


Şekil 2.1. Görüntü sorgulama teknikleri

2.1.1. Metinsel Tabanlı Görüntü Sorgulama Tekniği

Görüntü sorgulama tekniklerinden TBIR, görüntüleri metin ile açıklar ve daha sonra görüntü sorgulamayı gerçekleştirmek için metin tabanlı veritabanı yönetim sistemlerini kullanır (Rui ve diğ., 1999). Metin tabanlı yaklaşım, geleneksel basit bir anahtar kelime tabanlı aramadır. Görseller, görselin başlığı gibi açıklamalar veya anahtar kelimeler içeriğine göre manuel olarak indekslenir; dosya adı, web sayfasının başlığı ve alternatif etiket...vb. ve veritabanında saklanır. Bir kullanıcı sorgusunu işleme adımları, sorgu içinden durdurma sözcüklerinin kaldırılmasını ve kelimenin kökünün elde edilmesi gibi doğal dil işleme adımlarını içerebilir. Anahtar kelimeye dayalı görüntü alma yaklaşımlarından bazıları, kelime çantası modeline (Bag-of-Words) dayanır. Görüntü erişimi daha sonra bilgi alma teknikleriyle birlikte standart veritabanı yönetimi yeteneğine kaydırılır. Google Görsel Arama ve Yahoo Görsel Arama gibi bazı ticari görsel arama motorları, anahtar kelime tabanlı görsel alma sistemleridir.

Şekil 2.2’de TBIR adımları verilmiştir. Görüntünün meta verilerinin tanımlandığı ve veri tabanına görüntü dosya adı, biçimi veya görüntü boyutu gibi özellikleri kaydettiği aşama çevrimdışı aşama olarak adlandırılır. Daha sonra kullanıcı, bu açıklamalara dayalı olarak bazı kriterleri karşılayan tüm görüntüleri almak için metinsel veya sayısal sorgular formüle ederek sorgulama işlemini gerçekleştirir. Bu aşama çevrimiçi aşama olarak adlandırılır.



Şekil 2.2. Metin Tabanlı Erişim (TBIR)

TBIR ile ilgili iki geniş kapsamlı araştırma makalesi bulunmaktadır (Chang 1993, Tamura 1984). Bu yöntem 1990'ların başında büyük ölçekli görüntü koleksiyonlarının ortaya çıkması, manuel görüntü açıklamalarının da en büyük zorluğu olmuştur.

Sunulan başka bir çalışmada (Li ve diğ., 2011), web'den (Flickr.com) ilgili ve alakasız görüntüler toplanarak, bunları görüntü kategorizasyonu ve erişimi için etiketli eğitim verileri olarak kullanılmıştır. İlk olarak, ilgili ve alakasız Web görüntülerini kümelere ayırdılar ve ardından her kümeyi bir "torba" ve her bir torbadaki resimleri "örnekler" olarak ele aldılar. Örneklerin (görüntülerin) etiketlerini tahmin etmek için, pozitif torbaları seçmek için PMIL-CPB adlı aşamalı bir şema önerdiler. NUS-WIDE (Chua ve diğ., 2009) veri kümesini ve Google veri kümesini (Fergus ve diğ., 2005) kullanarak kapsamlı deneyler yaptılar ve sonuçlar yaklaşımlarının etkinliğini göstermektedir. Ancak bu yöntemler, pozitif bir torbanın en az bir pozitif örneğin varlığıyla belirlenebileceği kısıtlamasına dayanır. Bu varsayım, diğer olumlu örneklerin analizinin eksikliğine yol açar.

TBIR'ın dezavantajları aşağıdaki gibi sıralanabilir:

- Açıklamalar manuel olarak girilmelidir ve büyük bir görüntü veritabanı için manuel açıklama yapmak pratik değildir.
- Görüntü içeriğini yazıya dökmek görevi oldukça özneldir.
- İlgili arama sonuçlarında çok sayıda alakasız arama sonucu olabilir, bu da metin tabanlı aramanın hassasiyeti düşrür.
- Çoğu zaman, birkaç kelime görüntü içeriğini tam olarak tanımlayamaz ve birçok kelimenin birden fazla anlamı vardır. Bir yorumcu tarafından sağlanan metinsel açıklamalar diğer kullanıcıdan farklı olmalıdır.
- Kullanıcı metin sorguları ve resim açıklamaları için çeşitli farklılar olmalıdır.
- Metinsel açıklamalar dile bağımlıdır.

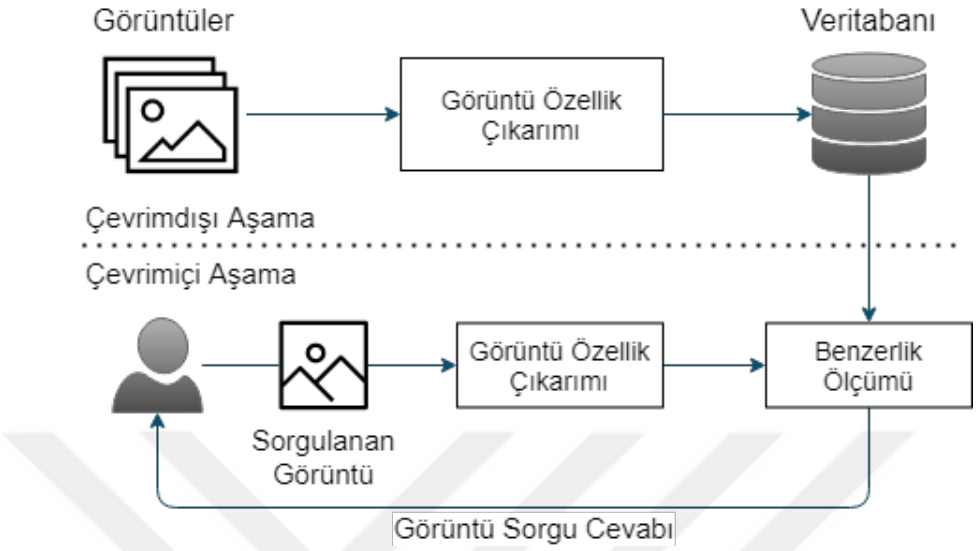
Bu durumdan kaçınmak için metin tabanlı sorgulamadan ziyade içerik tabanlı görüntü sorgulama iyileştirilmiştir. CBIR yöntemi, metin tabanlı anahtar kelimeler kullanmak yerine, görsellerin renk ve doku olarak görsel içerikleriyle tanımlanması gerektiği anlamına gelmektedir.

2.1.2. Görüntü İçeriğine Göre Görüntü Sorgulama Tekniği

CBIR, aktif ve hızlı ilerleyen bir araştırma alanı olarak kabul edilmektedir. Görüntü içeriğine göre sorgulama ve içerik tabanlı görsel bilgi alma (Mailaivasan, 2014) olarak da adlandırılır. CBIR terimi ilk olarak, Kato ve ark.'nın (Kato, 1992) renk ve şekle dayalı olarak bir veri tabanından görüntülerin otomatik olarak alınması çalışmasında ortaya çıktığı düşünülmektedir. CBIR terimi, normalde öznitelikler (renk, şekil, doku...vb.) olarak adlandırılan görüntü görsel içeriklerine dayalı geniş bir veritabanından istenen görüntülerin alınması sürecini tanımlamak için yaygın olarak kullanılmıştır (Singha 2012).

Şekil 2.3'te gösterildiği gibi, tipik bir CBIR, çevrimdışı özellik çıkarma ve çevrim içi görüntü alma olarak ikiye ayrılır. Çevrimdışı aşamada, sistem veri tabanındaki her görüntünün özelliklerini otomatik olarak çıkarır ve bunları bir öznitelik veri tabanında saklar (görüntülerin özellikleri çıkarılır ve öznitelik vektörleri ile temsil edilir). Çevrimiçi aşamada, kullanıcı sisteme bir görüntü sorgusu girer. Sorgu görüntüsünün özellikleri çıkarılır ve temsil edilir. Benzerlik, sorgu görüntüsünün öznitelik vektörü ile veri

tabanındaki görüntülerin öznitelik vektörleri arasında ölçülür. Son olarak, sistem sorgu görüntüsüne en çok benzeyen görüntüleri döndürür.



Şekil 2.3. İçerik Tabanlı Görüntü Erişim Teknikleri

İçerik tabanlı görüntü almada temel sorun, görüntüler arasındaki benzerliğin nasıl verimli bir şekilde ölçüleceğidir. Görsel nesneler veya sahneler çeşitli değişikliklere veya dönüşümlere uğrayabileceğinden, görüntüleri doğrudan piksel düzeyinde karşılaştırmak çok başarılı sonuçlar vermemektedir. Genellikle, görsel özellikler görüntülerden çıkarılır ve ardından görüntü temsili için sabit boyutlu bir vektöre dönüştürülür.

Görüntüleri sadece metinsel açıklamalarla sorgulamak yeterli değildir (Duanmu ve diğ., 2010). Bu sebeple görüntülerin içeriğine odaklanan bir sorgulama yöntemi önerdiler. Önerilen yöntem, iki aşamalı bir kümeleme tekniği ile görüntünün bağlamına göre değişebilen küçük bir görüntü tanımlayıcısı kullanır. Yazarlar, çalışmaları için COIL-100 görüntü kütüphanesini kullanılmıştır. Deneylerden elde edilen sonuçlar, önerilen yöntemin verimli olduğunu kanıtlamıştır.

CBIR için MPEG-7 tanımlayıcısına dayanan yeni bir yaklaşım sunulmuştur (Shao ve diğ., 2008). Her görüntüden sekiz baskın renk seçilmiştir, özellikler histogram kesişim algoritması ile ölçülmüştür ve daha sonra benzerlik hesaplama karmaşıklığı hesaplanmıştır.

Hata difüzyon bloğu kesme kodlamasından (EDBTC) çıkarılan özelliklere dayalı yeni bir görüntüleri sorgulama bir yaklaşımı önerilmiştir (Guo ve diğ., 2015). Görüntü özelliği

tanımlayıcısını oluşturmak için, EDBTC tarafından üretilen vektör niceleme (VQ) kullanan iki renk niceleyici ve BITMAP görüntüsü işlenir. Sorgu görüntüsü ile veritabanındaki görüntü arasındaki benzerliği değerlendirmek için Renk Histogramı Özelliği (CHF) ve Bit Deseni Histogramı Özelliği (BHF) olmak üzere iki özellik tanıtılmıştır. Deneylerden elde edilen sonuçlar, önerilen şemanın eski BTC tabanlı görüntü indeksleme ve diğer mevcut görüntü alma şemalarından daha iyi performans gösterdiğini göstermektedir.

Bölge şekli benzerliğine dayalı görüntü erişim yöntemi (Chang ve diğ., 2001) sunmulmuştur. Bu yöntem birkaç adımdan oluşur. İlk başta, görüntüler ilkel bölgelere ayrılır. Daha sonra, benzerlik değerlendirme sürecinde görüntülerin anlamsal birimleri olarak kullanılan anlamlı bileşik şekiller oluşturmak için birleştirilirler. Her anlamlı şekli karakterize etmek için üç şekil özelliği ve bir dizi normalleştirilmiş Fourier tanımlayıcısı kullanmışlardır. Son olarak, iki görüntüdeki en benzer şekil çiftini bularak iki görüntü arasındaki benzerliği ölçmüşlerdir. Daha karmaşık görüntülerin işlenmesinde bu yöntemle iki olası sorun vardır. İlk sorun, makinelerin anlamlı bölgeleri belirlemesinin oldukça zor olmasıdır. İkinci sorun, birçok özellik ve benzerlik modelinin önerilmiş olmasıdır, ancak bunların hiçbirinin insan görme modeliyle aynı olduğu kanıtlanmamıştır.

Görüntüler için basit bir renk histogramı tabanlı arama ve görüntü erişim algoritmasını uygulayan ve test eden bir proje gerçekleştirilmiştir (Chakravarti ve diğ., 2009). Çalışmalarının güçlü yönü, kodlama açısından uygulanmasının nispeten kolay olmasıdır. Ek olarak, sistemleri, boyut olarak dönüştürülmüş ve aynı zamanda döndürmeler ve çevirmeler yoluyla çevrilmiş görüntülerin erişilmesine olanak tanımaktadır. Ana zayıf nokta, renkli histogramların uygulanması, algoritma tarafından görülen ilgili görüntülerin bir insan tarafından görülen ilgili görüntülerle aynı olmasına izin vermez. Elde ettikleri sonuçlar karışık ve tutarlı değildir.

İncelemelerden sonra, görüntü özellikleri temsilinin renk, doku, ve şekil gibi düşük seviyeli görsel özellikler kullanılarak yapıldığı özetlenmiştir. CBIR temsilinin performansını artırmanın çözümlerinden biri de füzyonda düşük seviyeli öznitelikler kullanmaktır. Farklı yerel özelliklerin füzyonu kullanılarak anlamsal boşluk azaltılabilir ve yerel özelliklerin birleştirilmesi kullanılarak performans artırılabilir. Yerel ve küresel özelliklerin birleşimi de bu alanda gelecekteki araştırmalar için araştırma konularından

biridir. CBIR için son araştırmalar derin sinir ağlarının kullanımına kaydırıldı ve birçok veri kümesinde iyi sonuçlar göstermiştir.

2.1.3. Semantik (Anlamsal) Görüntü Sorgulama Tekniği

CBIR'nin erişim doğruluğunu iyileştirmek için, araştırmalar düşük ve yüksek özellikler arasındaki 'anlamsal boşluğu' azaltmaya odaklanmıştır. Üst düzey kavramları tanımlamak için ontolojiyi kullanmak da dahil olmak üzere, bu alandaki farklı yönleri kapsayacak birçok yeni çalışma önerilmiştir. SBIR, hibrit yaklaşımı temel alır ve sınıflandırma amacıyla şekil, renk ve doku tabanlı yaklaşımları kullanır. Semantik teknolojiler, düşük seviyeli görüntü özelliklerini yüksek seviyeli ontoloji kavramlarıyla eşleştirmeye çalışırken, görüntü sorgulamaya umut verici bir yaklaşım sunar.

Alt düzey özellikler (renk, şekil, doku, nesne algılama) ile üst düzey kullanıcı özellikleri (soyut, nesneler, olay) kategorileri arasındaki ilişki Wang tarafından tanımlanmıştır. CBIR, düşük seviyeli öznitelikleri kullanarak görüntüleri indeksler, ardından görüntü tanımı ile yüksek seviye anlambilim arasında bir yorumlama karşıtlığı gösterir, bu süreç anlamsal boşluk olarak adlandırılır (International Journal of Research in Computer and Communication). Araştırmacılar, birçok teknik önererek anlamsal boşluğu kapatmaya çalıştılar. Şekil 1.4'te gösterildiği üzere, semantik haritalama işlemi, düşük özelliklere dayalı olarak parçalanmış veya kümelenmiş bölge/nesneleri tanımlamak için en iyi konsepti bulmak için kullanılır. Bu haritalama, düşük seviyeli özellikleri nesne kavramı ile ilişkilendirmek için denetimli veya denetimsiz öğrenme araçları aracılığıyla yapmaktadır ve görüntü açıklama işlemi yoluyla metinsel kelime ile açıklama yapılmaktadır Hui ve diğ., 2010; Wang ve diğ., 2009).

Kullanıcının ilgilendiği kavramları dikkate alma fikri ile ortaya çıkmıştır ve anlamsal tabanlı model, akıllı görüntü alımında kullanımını önerilmiştir (Zarchi ve diğ., 2014). Kullanıcının ilgilendiği bölgeleri belirlemek için 'etkileşimli görüntü segmentasyonu' adı altında bir algoritma gerçekleştirilmiştir. Görüntülerin karınca kolonisi optimizasyon algoritması özniteliklerinin (düşük seviyeli) kullanılması ile fazlalık ve uygun olmayan özelliklerin ortadan kaldırılması mümkün olduğunu göstermişlerdir. Görsel içeriğin semantiği, yeni bir teknik önerisi ile temsil edilmektedir.

Semantik özelliklere de dayalı olarak, görüntü alımı için iki farklı strateji önerilmiştir (Kurtz ve diğ., 2014). Yeni terim farklılık ölçüsünde karşılıklı olarak imaj hem ontolojik terim hem de imaj temelli ilişkileri dikkate almışlardır. Çok ölçekli Riesz dalgacıkları, ontolojik terimleri otomatik olarak "yumuşak" olarak tahmin etmek için kullanılmıştır. Bu iki strateji, görüntü açıklamalarının desteğiyle benzer ve doğru görüntülerin alınmasında birleştirici bir faktör olarak çalışır.

Görüntülerin temsili ve anlamsal erişilmesinden kaynaklanan problemlerle başa çıkmak için ontoloji ve MPEG-7 tanımlayıcılarını kullanan bir çerçeve önerilmiştir (Roung ve diğ., 2012). Çerçeve, aşamalı olarak çoklu ontolojilerin oluşturulmasına izin vermektedir ve yalnızca görüntü arayanlar arasında değil, aynı zamanda farklı alanlar arasında da belirli bir alan için tek bir ontoloji oluşturmak yerine ontoloji bilgilerini paylaşır. Nave Bayes çıkarımı kullanılarak, ilgili fotoğrafları eşleştirmek için sorgu ve etki alanı ontolojileri arasındaki benzerlik tahmin edilir. RDF çevirisi, kullanıcı sorgularının indekslenmesi ve eşleştirme süreci bu mimarideki üç ana süreçtir.

Görüntü erişimi için anlamsal boşluğu doldurmak hala büyük bir zorluk olarak görülmektedir. Görüntü erişim araştırmaları konusunda çok fazla çaba ve çalışma olmasına rağmen, tatmin edici bir performans sağlamak için yeterli değildir. Bununla birlikte, düşük seviyeden üst seviye kavramlara haritalama ile ilgili zorlukların yanı sıra iyileştirilmesi gereken bazı alanlar vardır. Ayrıca geniş alan veritabanındaki semantik boşluğun üstesinden gelmek karmaşıktır, çünkü geniş alanlardaki görüntüler çeşitli kavramlar kullanılarak tanımlanabilmektedir. Önemli bir veriyi içeren soyut özniteliklerle geri almaya odaklanan görüntü alımına dayalı semantik kavram için daha iyi destek görmeye ihtiyaç vardır. Ayrıca çıkarılan semantik özellikler her türlü görüntü koleksiyonu için uygulanmalıdır. Ayrıca, insan algısına uygun ve insan müdahalesi olmaksızın benzer görüntülerin elde edilmesi için etkili yollara ihtiyaç vardır.

Görüntü açıklamalarında bazı sorunlar zorluk teşkil etmektedir: görüntülere otomatik olarak anlamlı etiketlerle açıklama eklenmelidir. Açıklama ekleme sürecini ve farklı dillere açıklama eklemeyi kolaylaştıran nesneleri tanımlamak için gelişmiş bir görüntü bölümleme algoritmalarının geliştirilmesi ve verilerin kaydedilmesinde büyük hacimli veri tabanları ile dağıtık sistemler üzerinde gerçeklenmesi gerekir.

2.2. Görüntü Sorgulama Tekniklerinde Akıllı Filtreler: Araç Tip, Renk ve Hız Tespit Etme Çalışmaları

Araç algılama ve sınıflandırma sistemi, Akıllı Ulaşım Sistemlerinin ve görüntü erişim tekniğinin önemli bir parçasıdır. Tez çalışmasında gerçekleştirilmek üzere seçilen kullanım senaryosunda araçlar fiziksel özelliklere göre filtrelenebilir. Bunlar araçların tip, renk ve hız özellikleridir. Bu başlık altında bu özellikleri kullanarak yapılan sınıflandırma çalışmalarına yer verilmiştir.

Mevcut araç sınıflandırma sistemleri donanımsal ve yazılımsal olarak iki alanda toplanabilir. Donanımsal olarak sensörler (Gajda ve diğ., 2001; Cheung ve diğ., 2005; Gupte ve diğ., 2002; Abdelbaki ve diğ., 2001) ve mikrodalga radarları (Urazghildiiev ve diğ., 2002) temel olarak kullanılmaktadır. Ancak bu cihazların karşılaştığı fiziksel engeller ve kötü hava durumu koşulları çalışmalarını olumsuz etkileyebilmektedir. Ayrıca kurulum ve bakım maliyetinin yüksek olması donanım tabanlı sistemlerin en büyük dezavantajlarındandır.

Son yıllarda yazılım alanında makine öğrenmesi ve derin öğrenme algoritmaları ile birçok araç tespit çalışması yapılmıştır (Spagnolo ve diğ., 2006; Hadi ve diğ. 2014). Derin öğrenme modelinde, özellikler öğrenmeye dayalıdır, makine öğreniminde ise özellikler manuel olarak çıkarılır ve ardından sınıflandırma algoritmaları bu özellikler üzerinde eğitilir.

En popüler nesne algılama yöntemlerinden biri olan çerçeve farkı alma, ardışık iki görüntü arasındaki farkın hesaplanmasına dayanır. Algoritmanın uygulaması basit ve kolaydır ve ayrıca değişen doğa koşullarına hızlı uyum sağlayan bir yapısı vardır. Ancak, Çerçeve farklılaştırma yönteminin çalışma doğruluğu, çerçeve hızına (FPS) ve nesnenin hızına bağlıdır. Bu durumda tespit edilecek nesnenin durması veya yavaşlaması durumunda bulunamaması kaçınılmazdır. İkinci popüler yöntem olan Optik akış yöntemi, görüntünün optik akış dağılım özelliklerini hesaplamayı amaçlar. Bu yöntem daha iyi çalışır, ancak büyük miktarda hesaplama gerektirir. Ayrıca gürültüye duyarlılığı nedeniyle gerçek zamanlı uygulamalar için ideal değildir. Bir diğer popüler yöntem de arka plan çıkarmadır (Rakibe vd diğ., 2013; Liu ve diğ., 2014; Athanesious ve diğ. 2012).

Arka plan çıkarma algoritması, arka plan modelleme yöntemine dayanmaktadır. BGS, geçmiş kareler üzerinden ortalama bir model oluşturmaya çalışır.

Literatürde en yaygın olarak kullanılan nesne tanıma yaklaşımlarından biri de, Haar-benzeri kademeli sınıflandırıcı ve destek vektör makine sınıflandırıcı ile yönlendirilmiş gradyanların histogramı (HOG) öznitelik tanımlayıcılarıdır. Bu iki ortak yaklaşım, iyi ve gerçek zamanlı performans sağlar (Gaszczaka ve diğ., 2011; Viola ve diğ., 2001).

Makine öğrenmesi yöntemleri kullanılarak gerçekleştirilen çalışmalar; (Kleyko ve diğ., 2015) yolda bulunan sensörleri kullanarak araçların sınıflandırılmasını gerçekleştirmişlerdir. D. Kleyko ve diğerleri, araç sınıflandırması için farklı makine öğrenme algoritmalarının karşılaştırmasını göstermiştir. Veri setleri 3074 örnekten oluşmaktadır. Sınıflandırma için lojistik regresyon, sinir ağları ve destek vektör makineleri kullanıldı. Sonuçlara göre lojistik regresyon, %93.4 sınıflandırma doğruluğu ile diğer makine öğrenmesi yöntemleri arasında en iyi performansı göstermiştir. Çalışmada karşılaşılan temel problem veri kümesidir, sınıflar arasında eşit dağılım bulunmamaktadır.

Araç tipi sınıflandırması için sınıflandırma şemalarının karşılaştırılmasını sunulmuştur (Chen ve diğ.,. 2011). Boyut ve şekil ölçümlerini kullanarak araçların nasıl sınıflandırılacağını göstermişlerdir. Sınıflandırmada dört araç sınıfı (araba, kamyonet, otobüs ve bisiklet/motosiklet) kullanılmıştır. İki farklı yaklaşımla sınıflandırma gerçekleştirilmiştir: iki özellik tabanlı sınıflandırıcı (SVM ve rastgele ormanlar) ve model tabanlı yaklaşım kullanmışlardır. Çalışmaları, SVM'nin %96,26 gerçek pozitif sınıflandırma doğruluğu ile RF'den daha iyi performans gösterdiğini gösteriyor. Boyut ve de şekil özelliklerinin önemli ölçüde benzerlik göstermesinden dolayı, otomobil ve kamyonet kategorileri arasında çok fazla sayıda yanlış sınıflandırma olmuştur.

Araç boyutuna göre araç tip sınıflandırması gerçekleştirilmiştir (Lai ve diğ., 2001). Taksi, minibüs ve çift katlı araç olmak üzere 3 sınıf üzerine çalışılmıştır. Çalışmalarına göre araç uzunluğundaki tahmin doğruluğu %92,5'in üzerindedir. Sınıflandırma doğruluğu yolun yüzey yüksekliğine bağlı olmasıdır.

Araçları küçüki orta ve büyük olmak üzere 3 sınıfta sınıflandıran çalışma bulunmaktadır (Kul ve diğ., 2017). Çalışmaları, arka plan çıkarma, ön plan tespiti, özellik çıkarma ve araç sınıflandırmadan oluşmaktadır. Gün ışığında SVM sınıflandırması ile %97,3 başarı elde etmişlerdir. Çalışmalarının farklı hava koşullarında nasıl çalıştığına dair sonuç yoktur.

Üç tür araç türü, küçük arabalar, büyük arabalar ve motosikletler sınıflandırılmıştır (Wang ve diğ., 2014). Bu sistem, çeşitli dış aydınlatma ve hava koşullarında gerçekleştirilmiştir. HOG Tanımlayıcı ve SVM sınıflandırıcı kullanılmıştır. Eğitim aşamasında, 1.200 örnek toplanır. Test aşamasında, 2.375 küçük araç, 400 büyük araç ve 2.715 motosiklet görüntüsü kullanılır. Ortalama olarak, kesinlik oranı %93,82, geri çağırma oranı ise %88'dir. Çalışmalarının farklı hava koşullarında nasıl çalıştığına dair sonuç yoktur. Veri setlerinde dengesiz dağılım bulunmaktadır.

Görüntü işleme ve donanımdaki gelişmelerin yanı sıra, derin öğrenme tekniklerinin nesne tanımada iyi sonuçlar vermesi ve bilinen diğer yöntemlerden çok daha iyi olması şaşırtıcı değildir. 2006 yılından bu yana derin öğrenme, makine öğreniminin popüler bir araştırma konusu haline geldi ve geleneksel yaklaşımlara kıyasla büyük bir ilgi gördü (Zhang ve diğ., 2016). Sınıflandırma senaryoları için Derin Sinir Ağlarının (DNN'ler) kullanılması, nesnelerin tanımlanması ve algılanmasıyla ilgili zorlukların üstesinden gelmek için mükemmel bir çözüm olarak gösterilmiştir. DNN'ler,Destek Vektör Makinesi (SVM) ve Boosting gibi yaygın sığ yapılardan daha iyi yapan yapı avantajına sahiptir (Lauzon 2012). Genel bir DNN, birden çok girdi katmanından, en az iki gizli katmandan ve bir dizi çıktı katmanından oluşur. Gizli katmanlar, nöronun bir model oluşturmadaki etkisine atıfta bulunarak, birkaç nöron ve ağırlıkları aracılığıyla matematiksel işlemleri gerçekleştiren bir DNN'nin ana yapısı olarak bilinir. Görsel veri analizi için, bir görüntüdeki birden çok özellik arasında konumsal ilişkinin varlığından dolayı en yaygın durumlarda derin Evrişimli Sinir Ağları (Convolutional Neural Networks) adı verilen özel bir DNN türü kullanılır. Nesne algılama amaçları için, görüntülerden nesne algılama modelleri oluşturmanın yaygın zorluklarının üstesinden gelmek için ConNN'lere dayalı bazı algoritmalar geliştirilmiştir. Bu bağlamda, Bölge Tabanlı Evrişimli Sinir Ağları (Region-based Convolutional Neural Networks, R-ConNN), sınırlı sayıda aday bölgenin seçici arama algoritması kullanılarak seçildiği ve daha sonra nesnelerin nihai öznitelik

vektörünü üretmek için bir ConNN'de kullanıldığı yaygın bir yaklaşımdır (Girshick ve diğ, 2014). Araç algılama uygulamaları için DNN'lerin uygulanması, en son çalışmalarda yüksek doğruluk sağlamak için oldukça güvenilir bir çözüm olarak bilinmektedir. Çoğu durumda, araştırmacılar video karelerinde araları algılamak için DNN dayalı modeller geliştirmişlerdir.

Araçların önden elde edilen görüntülerinden yarı denetimli bir evrişimsel sinir ağı kullanan bir araç tipi sınıflandırma yöntemi önermişlerdir (Dong diğ., 2015). Araçların sadece önden görünümleri ile 9850 yüksek çözünürlüklü görüntüden oluşan BIT-Vehicle veri kümesi oluşturulmuş ve yayınlanmıştır. Sistemler gündüz koşullarında %96,1, gece koşullarında %89,4 oranında çalışmaktadır. Veri seti içerisinde dengesiz dağılım bulunmaktadır.

Araba ve kamyonu sınıflandırmışlardır (Wang ve diğ., 2017). Sistemleri 192 CUDA çekirdekli NVIDIA Jetson TK1 kartı üzerinde derin evrişimli sinir ağları kullanarak %90 doğruluk elde etmiştir. Bir aracın tespitini yaklaşık 0,3 sn sürede gerçeklemişlerdir, bu da bu çalışmanın gerçek zamanlı sistemler için uygun olduğunu göstermektedir.

Araç tanıma için ResNet, GoogLeNet ve AlexNet ConNN'i kullandılar, MIO-TCD veri kümesini seçtiler. Sistemleri %97.92 doğrulukla çalışmaktadır (Lee ve diğ., 2017).

(Huo v diğ.,2016) araçların çoklu görünüm sınıflandırmasını, görünümleri (arka, ön ve yan) öznitelik olarak içeren bir öznitelik tahmin işlemini gerçeklemişlerdir. İlgili çoklu görev öğrenimi, araç kategorilerini (araba, kamyon, otobüs ve kamyonet) sınıflandıran ve öznitelikleri eşzamanlı olarak tahmin eden Bölge Tabanlı Evrişimli Sinir Ağı (RCONNN) çerçevesinde uygulanır. Sahada yakalanmış bir araç veri kümesi üzerinde yapılan deneyler, araç tipi sınıflandırması için yaklaşık %83 doğruluk ve öznitelik tahmini için %90'ın üzerinde doğruluk göstermektedir.

Önerilen araç tipi sınıflandırma yönteminde YOLO sınıflandırma algoritmasını kullanarak otobüsleri, minivanları, mikrobüsleri, SUV'leri, sedanları ve kamyonları sınıflandırılmıştır (Şentaş ve diğ., 2018; Şentaş ve diğ., 2018). %97,90 hassasiyet,% 99,60 geri çağırma ve% 89,29 IOU'ya ulaştılar.

BIT veri kümesi kullanırak araç sınıflandırma işlemi gerçekleştirilmiştir (Kul ve diğ., 2018). Otobüs, minibüs, kamyon, kamyonet, sedan ve SUV YOLOv4 ve tiny YOLOv4 kullanarak sınıflandırılmıştır. 98.39 doğruluk oranı ile tiny YOLOv4 en yüksek doğruluğu vermiştir.

Hafif model olan Tiny-YOLOv3 araç algılama yaklaşımı önerilmiştir (Tajar ve diğ., 2021). Önerilen sistemde, görüntülerdeki araçları tanımlamak, bulmak ve sınıflandırmak için uçtan uca bir yaklaşım kullanılır. Önceden eğitilmiş Tiny-YOLOv3 ağı ana referans modeli olarak benimsenir ve ardından BIT-araç veri kümesi üzerinde eğitim gerçekleştirir. Eğitim ağı üzerinde bazı gereksiz katmanlar hariç tutularak budanır ve basitleştirilir. Sonuçlar, önerilen yöntemin doğruluk ve hız açısından avantajlarını göstermiştir. Ayrıca ağ, 17 fps hızında MAP = %95,05 ile altı farklı araç türünü tespit edip sınıflandırabilir. Bu nedenle, orijinal Tiny-YOLOv3 ağından yaklaşık iki kat daha hızlıdır.

Ggeleneksel yaklaşımdan derin öğrenme yaklaşımına kadar yenilikçi yöntemlerle son yıllarda geliştirilen trafik gözetleme sisteminde araç alımının çok önemli bir rol oynadığını belirtilmiştir (Phung ve diğ., 2021). Ancak, Araç görüntü erişim işleminin hala çözülmesi gereken görüntü çözünürlüğü, görüntü bozulması, tıkanma gibi bazı zorlukları vardır. Önerdikleri sistemde Faster R-ConNN, SVM ve YSA kullanmışlardır. Çalışmalarının, BIT-Vehicle veri kümesinde araç erişiminde bazı durumlarda gelişmiş yöntemlerden daha iyi performans gösterdiğini belirtmişlerdir. MAP sonuçları %95.23'e ulaşmıştır.

Yapısal ve görünüşe dayalı özellikleri kullanan bir araç tipi sınıflandırma yöntemi sunulmuştur (Dong ve diğ., 2013). Araç parçalarının uzamsal göreli yerleşimlerini karakterize eden yapısal özellik, bir aracı arka plandan ayırt etmeye yardımcı olur ve görünüşe dayalı özellik, aydınlatma değişiminin ve arka planın müdahalelerine karşı yerel ve sağlam olduğunu göstermişlerdir. Araçların kompakt ve ayırt edici temsillerini elde etmek için bu iki tip özelliğin dağılımları hesaplanmıştır. Araç türlerini sınıflandırmak için Multiple Kernel Learning algoritmasını kullanmışlardır. Veri setindeki tüm araçlar beş kategoriye ayrılmıştır: otobüs, minibüs, minivan, sedan ve kamyon. Her kategori için eğitim örnekleri olarak 100 görsel ve test için 50 görsel seçilmiştir. Algoritmaları %91.3 başarıya ulaşmıştır.

Otomatik araç tipi sınıflandırması, verimli Akıllı Ulaşım Sistemlerinin (ITS) geliştirilmesinde zorunlu bir rol oynadığını belirtilmiştir (Hedeya ve diğ., 2020). Çalışmalarında, araç tipi sınıflandırma problemi için bir super-learner topluluğu önerilmiştir. Bireysel temel öğreniciler ResNet50, Xception ve DenseNet'in güçlü yönlerinden yararlanmak ve zayıflıklarını azaltmak için yoğun bağlantılı bir tek bölmeli süper öğrenici kullanılır. Önerdikleri yöntem, halka açık en zorlu trafik gözetim veri kümelerinden ikisi kullanılarak değerlendirilmiştir: MIOvision Trafik Kamerası Veri Kümesi (MIO-TCD) ve Pekin Teknoloji Enstitüsü'nün (BIT) araç sınıflandırma veri kümesi. BIT-Vehicle veri kümesi görüntüleri üzerinde eğitilen süper öğrenici varyantları, %97,62'ye varan toplam doğruluk elde edilmiştir.

YOLOv2 tabanlı yeni bir araç algılama modeli YOLOv2_Vehicle önerilmiştir (Sang ve diğ., 2018). Çalışmalarında eğitim veri kümesindeki araç sınırlama kutularını kümelemek için k-means++ kümeleme algoritması kullanılmış ve farklı boyutlarda altı bağlantı kutusu seçilmiştir. Araçların farklı ölçeklerinin araç tespit modelini etkileyebileceği düşünülerek, sınırlayıcı kutuların uzunluk ve genişlik kayıp hesaplama yöntemini iyileştirmek için normalizasyon uygulamışlardır. Pekin Teknoloji Enstitüsü (BIT)-Araç verisetini kullanmışlardır, test veri kümesindeki deneysel sonuçlarında ortalama hassasiyet (mAP) üzerinde %94,78'e ulaşmışlardır.

YOLOv3'e dayalı bir araç algılama yöntemi önerilmiştir (Shi ve diğ., 2021). Ön işleme için veri büyütme yöntemini benimseyerek, etiket yumuşatma stratejisi eklemek, modelin genelleme yeteneğini geliştirir ve ortalama algılama doğruluğunu daha da iyileştirir. Eğitim sırasında, çok ölçekli eğitim kullanılır ve uygunsuz öğrenme hızının ve bozulma adımlarının olumsuz etkisini azaltan, model eğitimini hızlandıran ve azaltan adım öğrenme hızı bozulma stratejisinin yerine kosinüs bozunma öğrenme hızı bozulma stratejisi kullanılır. ağ eğitim döngüsü. Deneysel sonuçlar, geliştirilmiş yöntemin orijinal modelin tespit doğruluğunu daha az eğitim döngüsünde önemli ölçüde iyileştirebileceğini, iyi bir sağlamlığa sahip olduğunu ve karayolu araçlarını etkili bir şekilde tespit edebildiğini göstermektedir.

Evrişimli bir sinir ağına dayalı karayolu araç tespiti için bir algoritma önerilmiştir (Chan ve diğ., 2021). Derin öğrenme omurgası için VGG-16 kullandılar. JSHD veri kümesinde %6.5'lik bir mAP değerine ulaştılar.

Derin öğrenme ve çok katmanlı özellik birleştirmeye dayalı bir araç algılama yöntemi önerilmiştir (Zhao Min ve diğ., 2018). Omurga ağı, 19 katmana sahip Darknet'tir ve ImageNet veri kümesinde VGGNet seviyesinin doğruluğunu sağlar, ancak VGGNet'ten iki kat daha hızlı çalışır. Elde ettikleri sonuçlar, özellik füzyon yapısının algılama doğruluğunu %73,7'den %77,9'a etkili bir şekilde artırabildiğini göstermektedir. Sığ katmanların gelişmiş semantik bilgisi nedeniyle, küçük nesnelerin tespiti için çok faydalı olduğunu belirtmişlerdir.

Daha Hızlı R-CONNN algoritmasına dayalı bir araç tespit yaklaşımı ile önerilen yöntemde (Turani ve diğ.,2019), derin öğrenme omurgaları modified ResNet-50'dir ve veri kümesi olarak Cars ve kendi verisetlerini içerir. Çalışma başarısı %98,5 hassasiyet faktörüne ulaştılar.

Fast araç tespiti için daha hızlı bir R-ConNN çerçevesi önerilmiştir (Nguyen ve diğ., 2019). ilk olarak, Faster R-ConNN'de temel evrişim katmanını oluşturmak için MobileNet mimarisini kullanmıştır. Son olarak, MobileNet mimarisindeki derinlemesine ayrılabilir evrişim yapısı, önerileri sınıflandırmak ve tespit edilen her araç için sınırlayıcı kutuyu ayarlamak, Faster R-ConNN çerçevesinin son aşamasında sınıflandırıcıyı oluşturmak için kullanılmıştır. KITTI araç veri kümesi ve LSVH veri kümesi üzerindeki deneysel sonuçlar, önerilen yaklaşımın hem tespit doğruluğu hem de çıkarım süresinde orijinal Faster R-ConNN'ye kıyasla daha iyi performans sağladığını göstermektedir. Daha spesifik olarak, önerilen yöntemin performansı, orijinal Faster R-ConNN çerçevesine kıyasla KITTI test setinde %4 ve LSVH test setinde %24,5 oranında iyileştirilmiştir.

Alex Net ve Faster RConNN derin öğrenme modellerini kullanarak araç tespiti için bir Karşılaştırmalı Çalışma önerildi (Espinosa ve diğ. 2017). Alex Net ve Faster R-ConNN, bir kentsel video dizisi PASCAL VOC 2007'nin analizi ile karşılaştırılmıştır. Doğrulukları %76 F1-Skorudur. Özellik birleştirme ile evrişimli sinir ağlarına dayalı kentsel trafik gözetleme görüntülerinde bir araç algılama önerilmiştir (Zhang ve diğ.,

2015). DP-SSD derin öğrenme omurgasını ve UA-DETRAC veri kümesini kullandılar. %75,43 mAP'ye ulaştılar.

YOLOv4 dedektörünü, özellikle, k-ortalama kümeleme kullanarak bağlantı kutusu tahminlerini optimize etmek gibi bazı mevcut yöntemleri kullanarak araç izleme uygulamaları için geliştirilmiştir (Mahto ve diğ. 2020). Ayrıca, YOLOv4'ü veri kümesi (UA-DETRAC) gereksinimlerine göre optimize etmişlerdir. İnce ayarlı modelimiz aynı zamanda mevcut modellerle hassasiyet, geri çağırma, F1 puanı, ortalama ortalama hassasiyet ve ortalama IoU gibi bir dizi performans metriği üzerinden karşılaştırılmıştır. Deneysel sonuçları, halihazırda gerçek zamanlı nesne algılama yeteneklerine sahip olan SOTA modelinin, yüksek oranda hedeflenen kullanım durumları için daha da geliştirilebileceğini göstermektedir.

Çalışmalar göstermektedir ki, araç tespit sisteminde geleneksel yöntemlerle çözülemeyen birçok problem derin tabanlı yöntemlerle çözülmüştür. Bununla birlikte, tıkanıklıklarla karşılaşma, çeşitli hava koşulları, karmaşık sahneler ve aydınlatma varyasyon sorunları gibi bazı sorunları ve zorluklar hala çözülmeyi beklemektedir. Neredeyse gerçek zamanlı algılama performansı için daha hızlı R-ConNN algoritması sağlanabilir, ancak tam gerçek zamanlı kullanım için YOLO çerçevelerinin daha iyi sonuçlar verdiği gözlemlenmiştir. Bu nedenle, doğruluğu ve algılama hızını artırmak için gelişmiş YOLO algoritmaları kullanılır.

Geçtiğimiz 60 yıl boyunca, araç algılama teorisi tam hızda gelişmiştir. Araç algılama sistemlerinin gelişmesi, ulaşım davranışlarının analizi üzerinde önemli bir etkiye sahiptir. Aynı zamanda plaka uzun süredir akıllı ulaşım sistemleri alanındaki temel araştırma nesnelerinden biri olmuştur. Ancak, araçlardaki plakalar her zaman tam olarak görünmez (kısmi kapanma veya bakış açısı değişikliği nedeniyle) ve belirli durumlarda (gürültü, bulanıklık veya bozulma nedeniyle) tanınması kolay değildir. Buna karşılık, araçlardaki boyalar, araç gövdelerinin çok daha büyük bir bölümünü kaplar ve kısmi tıkanıklık, bakış açısı değişikliği, gürültü ve bozulma gibi parazit faktörlerine nispeten duyarsızdır. Bu nedenle, araç rengi video izleme, suç tespiti ve kanun yaptırımı gibi çok çeşitli uygulamalarda değerli bir ipucu olarak kullanılmıştır (Li ve diğ., 2010; Xu ve diğ., 2009; Fang ve diğ., 2011; Guo ve diğ., 2005; Xu ve diğ., 2009). Bu avantaj, otomatik araç renk

tanımayı akıllı ulaşım sistemleri alanında önemli bir araştırma konusu haline getirmektedir (Wang ve diğ., 2006). Ancak kontrolsüz ortamlarda araç rengini belirlemek zorlu bir iştir. Zorluklar esas olarak iki yönden kaynaklanmaktadır: 1) Bazı renk türleri diğer renk türlerine çok yakın olması ve bu nedenle ayırt edilmesinin çok zor olabilmektedir. Örneğin, gerçek dünya görüntülerinde cam göbeği yeşilden o kadar ayırt edilemez. 2) Bir aracın rengi, pus, kar, yağmur ve aydınlatma değişikliği gibi çok sayıda parazit faktöründen etkilenmeye meyillidir.

Renk tanımanın özü, renkleri temsil edecek uygun özellikleri tasarlamaktır. RGB histogramı, RGB renk uzayının geniş kullanımı nedeniyle renk tanıma için doğal bir seçimdir. Bununla birlikte, görüntüler karmaşık doğal sahnelerde yakalandığında RGB histogramı yeterince sağlam değildir. Örneğin, düşük aydınlatma, güçlü ışık ve kamera renk sapması, görüntülerde ciddi renk kaymasına neden olabilir ve bu nedenle RGB histogramını kararsız hale getirebilir. Bu nedenle, bu sorunu çözmek için çeşitli yöntemler önerilmektedir. Normalleştirilmiş RGB, her kanalın değerini üç kanalın toplamına bölerek her pikselin ham RGB değerlerini normalleştirir. Bu özellik, güçlü ışığın neden olduğu üç kanalda tutarlı ofsetlerin üstesinden gelebilir. RGB veya normalleştirilmiş RGB uzayının üç kanalı bağımsız değildir, oysa HSV/HSI üç bağımsız kanala sahip başka bir geleneksel renk uzayıdır. Araçların renk tespiti ile ilgili bazı güncel çalışmalar aşağıdaki gibidir.

Mevcut renk tanımlayıcılarını araştırmışlardır ve lineer bir modelde renk kayması ve aşırı pozlamaya karşı değişmezliklerini analiz eden çalışma gerçeklenmiştir (Van De Sande ve diğ., 2010). Analize dayalı olarak önerilen rakip renk histogramları, aşırı pozlama ve renk kayması için değişmezdir. Daha sonra (Van de Sande ve diğ., 2011) GPU kullanarak renk özelliklerinin hesaplanmasını hızlandırmak için bir strateji önermişlerdir. Görüntülerin rengini tanımlamak için Correlogram'ı geliştirildi (Huang ve diğ., 1997). Korelogram, renk türlerinin çiftlerinin bir histogramıdır ve orijinal RGB renk histogramından daha sağlam olduğunu savunmuşlardır.

Arka plan görüntüsü ve arabanın iki kare görüntüsü kullanılarak renk düzeltmesi önermişlerilmiştir (Hsieh ve diğ., 2015). Sadece renk düzeltme yöntemi değil, Hsieh ayrıca, araba görüntülerinin pencere kısmını kaldıran ve arabanın tampon ve kapıları gibi

alt kısımlarını kullanarak araç rengini sınıflandıran cam kaldırma yöntemi de önerilmiştir. Arabanın oryantasyonu alınarak yapılan cam sökme işlemi, detay segmentli araba görüntüsünü elips şekline uydurur ve elipsin yarısını keser. Hsiehve diğ. deneyleri G-Classifier, DC-Classifier ve DG-Classifier olmak üzere üç farklı sınıflandırıcı kullanarak yapmıştır. Gri ve gri olmayan renkleri sınıflandırmaktan sorumlu G-Sınıflandırıcı. Yöntem, gri renk için üç kanalın ortalaması olan RGB'nin her kanalın renk değerine çok yakın olduğu varsayımıyla çok basit bir eşik yöntemidir. DC-Classifier ve DG-Classifier, RGB ve CIE Lab renk uzayından çıkarılan özelliklerle SVM kullanılarak eğitildi. DC-Classifier kullanılarak sınıflandırılan kırmızı, yeşil, mavi ve sarı renk sınıfı ve DG-Classifier kullanılarak sınıflandırılan renk sınıfının geri kalanı. Deneylerden, Hsieh ve ark. siyah, gümüş, beyaz, sarı, kırmızı, yeşil ve mavi olmak üzere 7 renk sınıfı ile sistemin ortalama doğruluğunun %93,59 olduğunu bildiriniz.

Araç rengi tanıma problemi için diğer bir yaklaşım, 2D histogram özelliklerini kullanarak araç rengini sınıflandırmaktır. 2D histogram özelliklerini ve SVM sınıflandırıcısını kullanarak araç renk tanıma sistemi geliştirilmiştir (Baek ve diğ., 2007). 2B histogramı oluşturmak için HSV renk uzayındaki ton ve doygunluk kullanılır. Deneylerden, sistemin ortalama doğruluğu %94,92'dir. Deneyde kullanılan veri kümesi, siyah, beyaz, kırmızı, sarı ve mavi renk sınıfı olmak üzere beş renk sınıfına sahip 500 dış mekan araç görüntüsüne sahiptir.

Akıllı ulaşım sistemleri uygulamaları için son teknoloji, gerçek zamanlı bir nesne algılama sistemi olan You Only Look Once (YOLO) 9000'i kullanan gerçek zamanlı bir araç renk tanıma sistemi geliştirilmiştir (Wu ve diğ., 2020). Gözetleme sistemlerinden toplanan VDCR veri kümesi sunmuşlardır. Bu veri kümesi, on araç rengini (beyaz, siyah, kırmızı, mavi, gri, altın, kahverengi, yeşil, sarı ve turuncu) içeren 5216 görüntüden oluşmaktadır. Yöntemleri %95,47 doğruluk değeri göstermektedir ve bir görüntü için test süresi 74,46 ms'dir.

Benzerlik yöntemini kullanarak renk tanıma için başka bir olası yaklaşım önerilmiştir (Son ve diğ., 2007). HSV renk uzayının ton ve doygunluk kanalında çalışan ızgara çekirdeği kullanan sistem. Son ve ark. her bir renk sınıfı için yalnızca kesinlik ve geri çağırma metriklerini sunmuşlardır. Deneylerden kesinlik ve geri çağırma yüzdesi çok

yüksektir ve %100'e yakındır. Yüksek hassasiyet ve yüksek hatırlama, modelin iyi bir doğruluğa sahip olduğunu gösterir.

Derin VGG-16 modeli temelinde, derin evrişim sinir ağına dayalı bir araç renk tanıma ağı modeliönerilmiştir (Zhang ve diğ., 2015). Bu yöntem, yaygın araç rengini hızlı ve doğru bir şekilde tanıyabilir ve iyi avantajlara ve uygulama beklentilerine sahiptir. Araç renkleri altı kategoriye ayrılmıştır: beyaz, siyah, kırmızı, yeşil, mavi, sarı. Eğitim veri kümesinde 2520, test veri kümesinde 1000 adet verileri bulunmaktadır. Ortalama %92.68 doğruluğa ulaşmışlardır.

Farklı renk uzaylarının birkaç renk histogramını birleştiren, araç renk tanıma için bir sözcük torbası (BoW) tabanlı özellik önerilmiştir (Chen ve diğ. 2014). Yerel yamaların özellikleri bir kod çizelgesi tarafından kodlanır ve bir araç görüntüsünün rengini temsil etmek için tek bir vektörde toplanır. Özellikler, basit koşullar altında (ör. açık hava ve tek tip aydınlatma) mükemmel performans elde edebilir, ancak zorlu senaryolarda (ör. gece veya pus) iyi performans göstermemektedir. Sistemlerinin doğruluğu %92'nin üzerinde.

Rachmadi ve diğ., (2015) evrişimli sinir ağı kullanan araç renk tanıma yöntemi önermişlerdir. Veri setleri, siyah, mavi, camgöbeği, gri, yeşil, kırmızı, beyaz ve sarı olmak üzere 8 araç rengi sınıfına sahip 15601 araç görüntüsü içermektedir. %0.9447 ortalama doğruluk oranına ulaşmışlardır.

Chuanping ve diğ., (2015) derin öğrenmeyi kullanarak araç rengi tanıma için bir yaklaşım önermişlerdir. Bu sistemde ConNN, her bir giriş araç görüntüsü için bir vektör çıktısı veren özellik çıkarıcı olarak benimsenmiştir ve sınıflandırıcı olarak SVM, aracın rengini tahmin eden bir sınıflandırıcı olarak kullanılmıştır. Önce görüntü uygun bir boyuta yeniden ölçeklendirilir. Eğitim bölümünde, geri yayılım algoritması kullanılarak ağın ağırlıkları ve önyargıları güncellenir. Bu eğitilmiş parametreler ağda saklanır. Son olarak, sınıflandırma için kullanılan eğitimli sonuç. SVM, eğitim sonucunu ve girdi görüntüsünü kullanarak çıktıyı sınıflandırır. Sistemin girişi pussuz görüntü, çıkış ise aracın renginin (siyah, beyaz, mavi...) olmasıdır. %94.6 başarıya ulaşmışlardır.

Dong ve diğ. (2014), araç rengi tanıma için plaka rengine dayalı yeni bir yöntem önerdi. Plakanın rengi, aydınlatma koşullarındaki değişikliklere duyarlı olmayan ön bilgi ve

plaka tanıma sonucu ile tanınır. Plakanın yanındaki araç ROI'sini (ilgi bölgesi) seçerek ve plaka görüntüsünün renk alanını ve araç ROI görüntüsünü RGB'den HSV'ye dönüştürüyorlar. Araç rengi, spektrumdaki ROI renginin ve plaka renginin göreli konumu ile tanımlanır. Veri setlerini toplamıştır 306 görüntüye sahiptir. K ırmızı araçlarda doğruluk oranı 0.73, sarı 0.93, mavi 0.97 ve yeşil 0.53'tür. Veri seti üzerindeki ortalama doğruluk oranı 0.75'tir.Deneysel sonuçları, plaka renginin araç rengini tanımaya yardımcı olduğunu göstermektedir.

Kim ve diğ. (2018) CCTV sistemleri için temsili renk bölgesi çıkarımı ve Evrişimsel Sinir Ağı (ConNN) aracılığıyla yeni bir araç renk sınıflandırma tekniği önermiştir. Harris köşe noktası algılama yöntemi, temsili bir renk bölgesinin olasılık haritasını oluşturmak için kullanılır. Olasılık haritasından, ConNN için bir giriş görüntüsü oluşturmak üzere noktalar rastgele seçilir. Önerilen yöntemin performansını değerlendirmek için otoyoldaki kameradan toplam 5.941 görüntü toplamışlardor. Performans değerlendirmesi için 5 kat çapraz doğrulama gerçekleştirmişlerdir. Araç renk tanıma yöntemlerinin performansı yaklaşık %96,1 olarak gösterilmiştir.

Pillai ve diğ. (2020) araç tiplerini, renklerini ve plakalarını sınıflandırmaya ve tespit etmeye odaklanmışladır. Bu makale, aracın renklerini, türlerini sınıflandırmak ve farklı hava koşullarında kamera görüntülerinden her aracın plakasını tanımak için bir Makine Öğrenimi modeli tasarlamayı önermektedir. Araç türleri arasında SUV, Sedan, Kamyon, Otobüs, Mikrobüs, Minibüs ve Motosiklet olmak üzere yedi sınıf bulunmaktadır. Araç renkleri sekiz sınıf içerir: yeşil, mavi, siyah, beyaz, gri, sarı, beyaz ve kırmızı. ConNN, gerçek zamanlı nesne dedektörü YOLO ve karakter tanıma modelinin uygulanması, aracın türünü, rengini sınıflandıracak ve plaka numaralarını ve harflerini farklı çevresel koşullar altında doğru bir şekilde tanımakta olduğunu belirtmişlerdir.

Dehghan ve diğ. (2017), Sighthound'un tam otomatik araç markası, modeli ve renk tanıma sistemini önermektedirler. Sistemlerinin omurgası, yalnızca hesaplama açısından ucuz olmakla kalmayıp aynı zamanda çeşitli rekabetçi kriterlerde son teknoloji sonuçlar sağlayan derin bir evrişimsel sinir ağıdır. Ek olarak, derin ağları yarı otomatik bir süreçle etiketlenen birkaç milyon görüntüden oluşan büyük bir veri kümesi üzerinde eğitilmiştir. Kendi dahili test veri setlerinin yanı sıra birkaç genel veri kümesinde de sistemlerini test

etmişlerdir. Sonuçları, tüm kıyaslamalarda diğer yöntemlerden önemli farklarla daha iyi performans gösterdiğimizi gösteriyor.

Günümüzde video gözetim sistemi için belirli görüntülerin rengini belirlemek çok önemlidir. Polis, olay mahalline her geldiğinde olay yerindeki güvenlik kameralarından faydalı bilgiler alabilecek. Bu önemli bilgiler arasında "renk" önemli bir rol oynar ve bir nesnenin boyutundan, konumundan, zamanından veya şeklindeki değişikliklerden etkilenmez. Gerçek zamanlı renk tanımlamanın doğruluğunu ve verimliliğini etkileyen gerçekler arasında kamera merceğinin malzemesi ve parametresi, donanım kısmı için kızılötesi parametreleri ve yazılım algoritmalarının verimliliği, yazılım kısmı için doğruluk ve pratik derece vb. bulunur. Statik renk analizini kullanan birçok yöntem önerilmiş olmasına rağmen, gerçek zamanlı video sisteminde renk tanımayı etkin bir şekilde çözemezler. Çalışmalar derin öğrenme yöntemlerinin başarılı sonuçlar verdiğini göstermektedir.

Araç hız tespiti trafik güvenliği için de önemlidir ve literatürde hareketli nesnelerin hızlarını belirlemek veya tahmin etmek için birçok çalışma bulunmaktadır. Bu çalışmalar genellikle ya ekstra donanım kullanır ya da sorunu görüntü işleme teknikleriyle çözmektedirler. Araç hız tespiti, hız sınırlama yasasına uymak için çok önemlidir ve aynı zamanda trafik koşullarını da gösterir. Geleneksel olarak, araç hızı tespiti veya gözetimi, radar teknolojisi, özellikle radar dedektörü ve radar tabancası kullanılarak elde edilirdi. Radar sisteminin çalışması, Doppler kayması fenomeni olarak bilinir. Bu sistemle ilgili temel kavram, oluşturulan sesin hareket halindeki bir araçtan yansıtılması ve geri dönen sesin frekansının biraz değişmesiyle meydana gelen Doppler kaymasıdır. Mekansal denklemler ve ekipman ile bu yöntem, hareket eden bir aracın hızını elde eder. Bununla birlikte, bu yöntemin hala, radar tabancasının yönü gelen aracın doğrudan yolunda olmadığında meydana gelen kosinüs hatası gibi birkaç dezavantajı vardır. Buna ek olarak, ekipman maliyeti ve ayrıca gölgeleme (farklı yüksekliklere sahip iki farklı araçtan radar dalgası yansıması) ve radyo paraziti (iletimin yapıldığı radyo dalgalarının benzer frekansının varlığından kaynaklanan hata) önemli nedenlerden biridir. yayın) hız tespiti için hatalara neden olan diğer iki etkili faktördür ve son olarak, radar sensörünün herhangi bir zamanda sadece bir arabayı takip edebilmesi, bu yöntemin bir başka sınırlamasıdır. Birçok araştırmacı, araçların hızını tespit etmek ve ölçmek için pek çok teknik

uygulamaya çalışmaktadır. Tüm teknikler donanım ekipmanına ve bilgisayar yazılımına dayanmaktadır.

Elektronik donanım ve bilgisayar yazılımı kullanan çalışmalar şu şekilde sıralanabilir: Ki ve diğ. (2006) Kore'de araçların hızını ölçmek için çift döngülü dedektör donanımı ve Visual C++ yazılımı kullandı. Wilder ve diğ. (2008), Pelegri ve diğ. (2002), ve Koide ve diğ. (2006) araç hızını tespit etmek için bilgisayar yazılımı ile birleştirilmiş manyetik sensörler önerdiler. Cheng ve diğ. (2005) Araçların hızını gerçek zamanlı olarak ölçmek için Lazer tabanlı müdahaleci olmayan algılama sistemi kullandı. Z. Osman ve diğ. (2002) Araç hızını tespit etmek için mikrodalga sinyali uygladı. Jianxin Fang ve diğ. (2007), araçların hızını tespit etmek ve araçları sınıflandırmak için sürekli dalga radarını kullandı.

Bilgisayar yazılımı ve görüntü işlemeile gerçekleştirilen birçok çalışma bulunmaktadır. Huei-Yung ve Kun-Jhih (2004) araç hızını bulmak için bulanık görüntüler kullanmış ve Pumrin ve Dailey (2002) otomatik hız ölçümleri için kamera hareket algılamayı kullanmıştır. Shisong et al. (2006), araç hızı ölçümleri için özellik noktası takibinden yararlanmıştır.

Aliane ve diğ. (2011), hız sınırı ihlal edildiğinde sürücüyü araç hoparlörlerinden sürücüye sesli mesajlar şeklinde uyaran bir sistem önerdi. Sistem, sürücülerin alınan uyaranlara belirli bir süre tepki vermesine izin verir. Bu sürenin sonunda sürücünün hız sınırını ihlal etmeye devam etmesi halinde aracın GPS konumu ve yakalanan görüntü veritabanına kaydedilir. Sistem, dahili bir kamera sistemi ile birlikte verileri kaydeden bir araç içi bilgisayardan oluşur. Ancak, ekipmanın maliyeti çok yüksek olduğu için bu arabanın kullanışlı olmayabileceği gözlemlenmiştir.

Rad ve diğ. (2010), araçların hızını tahmin etmek için başka bir yaklaşım önerdi. Bu çalışmada, bir otoyol üzerine monte edilmiş sabit bir kamera ile çekilen trafik filmleri toplanmıştır. Kamera, doğrudan referanslar kullanılarak desteklenen geometrik denklemlere dayalı olarak kalibre edilmiştir. Kesin ölçümler için kamera kalibrasyonu mümkün olabilirken, doğru hız tahmininin elde edilmesi hala oldukça zor olabildiğini belirtmişlerdir. Tespit edilen araç hızının ortalama hatası ± 7 km/sa elde etmişlerdir ve deney farklı çözünürlüklerde ve farklı video dizilerinde çalıştırılmıştır.

Luvizon ve diğ. (2017), şehir içi yollarda araç hızı ölçümü için, video tabanlı bir sistem önermişlerdir. Araç plakalarını verimli bir şekilde bulmak için bir hareket dedektörü ve bir metin dedektörü kullanmışlardır. Ayırt edici özellikler daha sonra plaka bölgelerinde seçilir, birden fazla çerçeve boyunca izlenir ve perspektif bozulması için düzeltilir. Araç hızı, izlenen özelliklerin yörüngeleri bilinen gerçek dünya ölçüleriyle karşılaştırılarak ölçülür. Yazarlar önerdikleri sistemi, tek bir düşük maliyetli kamera tarafından farklı hava koşullarında kaydedilen yaklaşık beş saatlik videoları içeren bir veri kümesi üzerinde test etmişlerdir. Ölçülen hızların ortalama hatası -0,5 km/sa olup, vakaların %96.0'ından fazlasında çeşitli ülkelerde düzenleyici makamlar tarafından belirlenen [-3,+2] km/s sınırının içinde kalmaktadır. Plaka dedektörleri 0,93 hassasiyet ve 0,87 geri çağırma performansı elde etmişlerdir.

Cheng ve diğ. (2020), araç hızını hesaplamak için hedef takibini sağlamak için hareketli hedefleri çıkarmayı ve merkez mesafe eşleştirme yöntemini uygulamayı önerdi. İlk olarak, video görüntü dizisine ön adım uygulandı. Daha sonra, hedef ve kontur tespitini gerçekleştirmek için müteakip arka plan çıkarımı için KNN algoritmasına dayalı arka plan modelleme yöntemi kullanılmıştır. Bitişik video görüntü karelerinin hedef kontur özelliğinin fazla değişmemesi özelliğinden hareketle, araç takibi için merkez mesafe eşleştirme yöntemi benimsenmiştir. Deneysel sonuçlar, hız algılamanın bağıl hatasının yaklaşık %5'te kontrol edilebileceğini ve video izleme hız algılamasının gerçek zamanlı gereksinimlerini karşılayabileceğini göstermektedir.

Bell ve diğ. (2020), sabit bir DSLR kamera aracılığıyla bir araç hız tahmini önerdi. Araç tespiti için derin öğrenme algoritması YOLOv2 kullanılırken, araçların takibi içinimple Online Realtime Tracking (SORT) algoritması kullanıldı. Perspektif projeksiyonu ve ölçek faktörü, bir homografi aracılığıyla karşılık gelen görüntü ve gerçek dünya koordinatlarının uzaktan eşlenmesiyle ele alınmıştır. Ardından, kamera görüntülerinin İngiliz Ulusal Grid Koordinat Sistemine dönüştürülmesi, düzlemsel yol yüzeyinde gerçek dünya mesafelerinin türetilmesine ve ardından eşzamanlı araç hızı tahminlerinin gerçeklenmesi sağlanmıştır. İzleme, araçların sık sık hız değiştirdiği yoğun şehirleşmiş bir ortamda gerçekleştirilmiştir, bu nedenle çerçeveler arasında tahminler ardışık olarak belirlenmiştir. Hız tahminleri, bir GNSS ve IMU donanımlı araç platformundan alınan kesin yörüngeleri içeren bir referans veri kümesine göre doğrulanmış ve tahminler,

sırasıyla 0,625 m/s ve %20,922'lik ortalama bir ortalama karekök hatası ve ortalama mutlak yüzde hatası elde edilmiştir.

Wahyu ve diğ. (2007), video verilerinden görüntü işlemeyi ve birçok farklı kamera açısıyla Öklid mesafe yöntemini kullanarak bir araç hızı tahmini önerdiler. İlk adımda, video verileri çerçevelere çıkarılmıştır ve gölge etkisini en aza indirmek için çıkarılan çerçevelere ön işleme uygulamışlardır. Ardından, ön plan görüntüsünü çıkarmak için Gauss Karışım Modeli (GMM) kullanmışlardır. Bir sonraki adımda, elde edilen ön plan medyan filtresi, gölge giderme ve morfoloji işlemi kullanılarak filtrelemişlerdir. Tespit edilen araç nesnesi, çerçeveler arasındaki mesafeye göre hızı tahmin etmek ve her çerçevedeki konumu belirlemek için takip edilmiştir. Elde edilen sonuçlardan hareket eden aracın hızını en düşük doğruluk oranı %87,01 ve %99,38 ile en yüksek doğruluk oranına ulaşmışlardır.

Wu ve diğ. (2009), video kamera kullanarak gerçek zamanlı otomatik araç hızı izlemeyi gerçekleştirmek için dijital görüntü işlemeye dayalı yeni bir yöntem sunar. Geometrik optiğe dayalı olarak, ilk önce görüntü alanındaki koordinatları gerçek dünya alanına doğru bir şekilde eşlemek için basitleştirilmiş bir yöntem sunar. Deney, yalnızca tek bir dijital video kamera ve bir yerleşik bilgisayar gerektirdiğini ve aynı anda birden fazla şeritte araç hızlarını izleyebildiğini göstermektedir. Tespit edilen araç hızının ortalama hatası %4'ün altındadır.

Afifah ve diğ. (2019), trafik gözetim sistemleri için gerekli olan araç hızının belirlenmesini önermektedir. Çalışmalarında, çeşitli çevresel koşullar altında videodaki hedef araçların yerini belirlemeye yönelik kapsamlı bir yaklaşımı hedef almışlardır. Videodan çıkarılan geometri özellikleri sürekli olarak bir profile yansıtılır ve sürekli olarak izlenir. Araç şekillerini, renklerini ve türlerini tanımadaki karmaşıklığı telafi eden araç tanımlama için özelliklerin zamansal bilgilerine ve hareket davranışlarına takip etmişlerdir. Araç hızı tespiti Python dili ve OpenCV kütüphanesi ile gerçeklemişlerdir. Gerçekledikleri temel adımlar: arka plan çıkarma, öznitelik çıkarma, araç izleme vb. Hız, çerçeve sayısı ve çerçeve hızı üzerinden aracın kat ettiği mesafe kullanılarak belirlenir.

Koyuncu ve diğ. (2018), önerdikleri teknikte, kamera görüş alanından (FOV) geçen bir araç tespit edilir. Araç hızı, FOV tarafından kat edilen gerçek mesafe ile FOV'a giren ve

çıkan araç arasındaki sürenin oranı olarak hesaplanmıştır. Zaman süresi, FOV'un başlangıcındaki ve sonundaki zaman damgaları arasındaki süre olarak belirlenmiştir. İkinci teknikte, araç hızı, ilk çalıştırma zaman damgasına göre FOV içindeki farklı zaman damgalarında hesaplanır. Kamera alanı içerisinde istenmeyen araç olmaması için FOV ortasında dikdörtgen bir alan oluşturulmakta ve her iki teknikte de bu alanda sadece ilgili aracın yakalanması için görüntü işleme teknikleri uygulanmaktadır. Doğru araç hızları 50 km/saatin altında hesaplanır. Bu hızlar, araba hız ölçerleri ile elde edilen gerçek hızlarla karşılaştırıldı ve video sistemi ile tespit edilen hızların, 50km/saatin altında hızölçer hızlarına ±1.2m/sn yakın olduğunu tespit etmişlerdir.

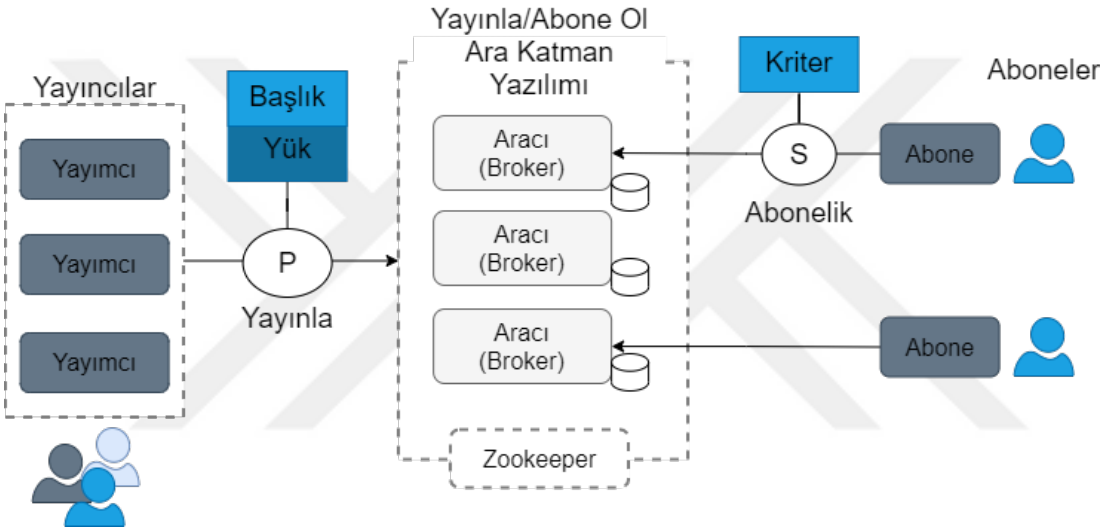
Farklı yöntem ve algoritmalar kullanılırken araçlar tespit edilemeyebilir veya yanlış tam tespit edilebilir. Aracın diğer şeritten hareket etmesi nedeniyle aracın yanlış algılanmasını ortadan kaldırmak için veya ağaç sallama gibi diğer küçük hareketler, ROI çıkarma kullanılarak önlenebilir. Arka plan, ROI maskesi ile çarpılır. Böylece araçlar doğru bir şekilde tespit edilir. Ayrıca gürültüyü azaltmak için eşikleme ve morfolojik işlemler de kullanılmaktadır. Eşik değerlemede eşik değeri seçimi çeşitli yöntemlere dayanmaktadır.

2.3. Yayınla/Abone Ol Mesajlaşma Kullanım Alanları

Yayınla/abone ol paradigması, asenkron iletişim, gevşek bağlı (loosely coupled) ve dağıtılmış uygulamalar için gereksinim arttıkça, giderek daha popüler hale geldi. Bir yayınla/abone ol sistemi, yayıncılardan (üreticilerden) abonelere (tüketicilere) konular aracılığıyla bilgi gönderilmesine izin verir. Operating Systems Principles Sempozyumunda sunulan bir makalede açıklanan Isis Araç Takımı'nın (Birman ve Joseph, 1987) "haber" alt yapı sistemi, kamuya açık olarak açıklanan ilk yayınla/abone ol sistemlerden biriydi. Yayınla-abone ol, yayıncı olarak bilinen mesaj göndericilerinin, doğrudan aboneler olarak bilinen bireysel alıcılarla gönderilecek mesajları tasarlamadığı, bunun yerine, eğer varsa hangi abonelerin var olabileceğini bilmeden, yayınlanan mesajları konulara göre gruplandırdığı bir iletişim modelidir. Aboneler, bir veya daha fazla konuya ilgi duyduklarını beyan eder ve hangi yayınların dahil olduğunu bilmeden yalnızca kendileriyle ilgili iletişimleri alırlar. Aboneler genellikle yayınlanan toplam iletilerin yalnızca bir alt kümesini alır. Konu tabanlı ve içerik tabanlı filtreleme, en popüler iki filtreleme türüdür (Eugster ve diğ., 2003; Banavar ve diğ., 1999; Campailla

ve diğ., 2001; Eugster ve diğ., 2000; Fabret ve diğ., 2001; Huang ve diğ., 2001; Pereira ve diğ., 2000). Bu paradigma, seçici bilgi dağıtımından ağ ve iş akışı yönetimi gibi dağıtılmış uygulama yönetimine kadar uzanan uygulamalarda geniş bir kullanım alanı bulmuştur.

Açıklandığı gibi yayıncılar ve aboneler diğerinin varlığından tamamen habersizdir. Broker adlı üçüncü bir bileşen, aralarındaki bağlantıyı yönetir. Broker sorumluluğu, gelen tüm mesajları filtrelemek ve bunları abonelere uygun şekilde dağıtmaktır. Şekil 2.4, yayınla/abone ol mimarisinin örnek diyagramını göstermektedir.



Şekil 2.4. Genel Yayınla/abone ol mimarisi

Bir Konu, bu işlevselliği mümkün kılan merkezi bileşendir. Yayınla/abone ol mimarisini, bir kullanıcı veya hizmet bunları talep edip alana kadar bireysel mesajların toplu olarak işlendiği tipik mesaj aracılarından ayıran şey budur. Bu yöntem, iletiler için sürekli olarak bir ileti kuyruğunu sorgulamaya gerek kalmadan olay güdümlü hizmetlerin oluşturulmasını sağlar. Ayrıca geliştiricilerin aynı mesajı (veri) kullanarak paralel olarak çalışabilen ve birçok aboneye hizmet verebilen çok sayıda yalıtılmış işlev tasarlamasına olanak tanır. Yayınla/abone ol, Java Message Service (JMS) (URL-2;Edward, 2005), Apache ActiveMQ (URL-3), Apache Apollo (URL-4), Apache Qpid (URL-5), Apache Rocket MQ (Ma ve diğ., 2017), Apache Kafka (URL-6), Amazon Kinesis (URL-7), ve Google Pub/Sub dahil olmak üzere çeşitli teknolojiler kullanılarak uygulanabilir.

Yayınla/abone ol ilk olarak merkezi istemci-sunucu sistemlerinde uygulanmış olsa da, mevcut araştırmalar ağırlıklı olarak dağıtılmış sürümlere odaklanmaktadır. Dağıtılmış yayınla/abone olmanın en önemli faydası, yayıncılar ve abonelerin doğal olarak ayrılmasıdır. Yayıncılar, verilerinin potansiyel tüketicileriyle ve aboneler de potansiyel ilginç veri üreticilerinin konumlarıyla ilgilenmediğinden, yayınlama/abone olma sisteminin istemci arayüzü güçlü ancak basit ve sezgiseldir.

Yayınla/abone olma sistemlerinin birkaç farklı sınıfı vardır. Konuya dayalı yayınla/abone ol (Oki, 1993), her yayınla ilişkili, kapsanan veriler için ilgi alanını gösteren bir konuya sahiptir. Belirli bir konuya abone olan müşteriler, belirtilen konuyla ilgili tüm yayınları alacaktır. Konular, grup iletişimi bağlamında kullanılan grup kavramına benzer (Powell, 1996). İçerik tabanlı yayın/abonelik sistemleri, abonelerin bir yayın içindeki veriler üzerinde kısıtlamalar belirlemesine izin vererek önemli işlevsellik katar.

İçerik tabanlı yayınla/abone ol araştırmasına temel bir katkı SIENA sistemi ile sağlanmıştır (URL-8), SIENA, geniş alan ağı üzerinden verimli ve ölçeklenebilir bildirim yönlendirmesi sağlamaya odaklanır. SIENA, bir aracı ağ mimarisine dayalıdır (bir TCP veya UDP taşıma katmanı üzerinden konuşlandırılır) ve filtreleme tabanlı yönlendirme yaklaşımının tanıtıldığı sistemdir. Bunun yanı sıra, SIENA, abonelik yayılımını sınırlamak için, aralarındaki sınırlama ilişkilerinden yararlanarak teknikler sunar: şu şekilde özetlenebilir σ2 ile eşleşen tüm olaylar aynı zamanda σ1 ile eşleşiyorsa, bir σ1 aboneliği başka bir σ2 aboneliğini içerir. Bir abonelik güncellemesi gerçekleştiğinde, bu aracı daha önce kapsayıcı bir aboneliği yaymışsa, yeni abonelik bir aracı tarafından yayılmaz. SIENA'da tanıtılan olay yönlendirme sürecine yardımcı olacak bir başka teknik de reklamlardır. Bir yayıncı tarafından üreteceği olaylar kümesini bildirmek için bir reklam yayınlanır. İlgili aracılar kümesini daha da azaltmak için yönlendirme yolları oluştururken reklamlar da dikkate alınır. SIENA algoritmaları, uygulama düzeyinde bir ağda içerik tabanlı olayların ve aboneliklerin yönlendirilmesi sorunu (içerik tabanlı yönlendirme sorunu) için bir referans çözüm haline gelmiştir. İçerik tabanlı yönlendirme için genel bir teorik çerçevenin yanı sıra orijinal SIENA algoritması üzerinde bazı varyantlar ve bir performans değerlendirmesi önerilmiştir.

REBECA (Rebeca Event-Based Electronic Commerce Architecture kısaltması - www.gkec.informatik.tu-darmstadt.de/rebeca), Java'da uygulanan ve içerik tabanlı bir adresleme modeli sağlayan nesne yönelimli bir bildirim hizmeti çerçevesidir. Rebecas topolojisi, bildirimleri istemcilere yönlendirmek için birbirleriyle iletişim kuran bağlı olay aracılarının döngüsel olmayan bir grafiği tarafından verilir. Sistem, olay taşmasının yanı sıra farklı filtreleme tabanlı algoritmalarla genişletilmiş genel bir yönlendirme motoru sağlar, yani: a) basit yönlendirme (olayların üreticilerde filtrelenmesine izin veren abonelik taşmasına benzer), b) kimlik yönlendirmesi (ki önceden iletilen bir başka abonelikle aynı olan bir aboneliği iletmekten kaçınır), c) yönlendirmeyi kapsar (benzer şekilde, halihazırda gönderilmiş olan başka bir aboneliğin kapsadığı bir aboneliği iletmekten kaçınır) ve d) birleştirme yönlendirmesi (bu, bir aracının mevcut yönlendirme girdilerinin aboneliklerini birleştirmesine ve iletmesine olanak tanır) sadece bu birleşme). Rebeca, bildirim yönlendirmesini daha da optimize etmek için ek bir mekanizma olarak reklamların kullanımını da destekler ve önceki tüm yönlendirme algoritmalarıyla birlikte kullanılabilir. Rebeca, yapılandırılabilir sistemler, mobilite desteği, konsepte dayalı adresleme, güvenlik yönleri, kapsam belirleme, modülerlik vb. gibi farklı yönleri keşfetmek için araştırma projelerinde yaygın olarak kullanılmıştır. Rebeca, topluluğa açık kaynaklı bir proje olarak sunulmaktadır.

Team (2004), IBM Watson tarafından geliştirilen içerik tabanlı bir yayın/alt sistemdir. Gryphon, bir aracı (broker) ağ mimarisine dayanmaktadır. Brokers overlay topolojisi, tepe noktası aslında düğüm kümeleri ve kenarları bağlantı demetleri olan bir ağaçtır. Düğümlerin ve bağlantıların çoğaltılması, yük dengeleme ve yüksek kullanılabilirlik amaçlarıyla kullanılır. Yayıncılar kök brokerlerde, aboneler ise yapraklarda bağlanır. Bhola ve diğ. (2002)'da olay yönlendirme algoritması, Filtreleme tabanlı sınıfa aittir: olaylar, yayıncılardan abonelere akış yönünde yayılır. Her aracı, yalnızca ilgili bir abonenin akış yönünde mevcut olması durumunda bir olayı yayar. Gryphon algoritmasını düz bir Filtreleme tabanlı yönlendirmeden ayıran özellik, düğümler ve bağlantı arızaları nedeniyle kaybedilebilecek olayların düzenli ve güvenilir bir şekilde tesliminin garantisidir. Klasik Filtreleme tabanlı algoritmada olduğu gibi, bir aracıda bir olayı filtrelemeye veya onu aşağı akışa iletme kararı, aboneliklerin aracılar ağı boyunca yayılmasını gerektirir. Zhao ve diğ., (2004)'nin abonelik yayma algoritması, komisyoncu

çoğaltma, hatalar, mesaj kayıpları ve yedekli bağlantılar üzerinden yayılmaya rağmen abonelik bilgilerinin her aracıda her zaman tutarlı olmasını sağlar.

Konu tabanlı sistemler için alternatif bir yaklaşım, Microsoft Research'te tasarlanmış bir araştırma sistemi olan Scribe (Castro ve diğ. 2002)'de önerilen yaklaşımdır. Scribe, uygulama düzeyinde bir aracılar ağında mesajların verimli, büyük ölçekli bir şekilde yönlendirilmesini sağlayan Pastry (Rowstron ve diğ., 2001) adlı yapılandırılmış bir ağ altyapısı üzerine kurulmuştur. Pastry'deki her aracıya ağda benzersiz bir tanımlayıcı atanır ve mesajlar yalnızca tanımlayıcısını belirterek belirli bir aracıya yönlendirilebilir. Scribe aslında Pastry kullanılarak yazılmış bir uygulamadır ve bunun için bir yayınla/abone ol arayüzünü temsil eder. Abonelik yönlendirme, olay yönlendirme ve bildirim yönlendirme, Pastry'nin yönlendirme yeteneklerinden yararlanılarak Scribe'da uygulanmaktadır. Scribe, yapılandırılmış bir bindirme mimarisi üzerinde buluşma tabanlı bir yönlendirme algoritmasına dayanır. Aşağıda haritalama politikasının nasıl uygulandığını açıklıyoruz. Temel fikir, her konuya rastgele bir tanımlayıcı atanması ve konuya en yakın tanımlayıcıya sahip Pasta düğümünün o konu için hedef aracı haline gelmesidir. Her konu için ilgili hedef aracıya dayanan bir çok noktaya yayın ağacı oluşturulur. Yeni bir düğüm bir özneye abone olduğunda, aboneliği Pastry tarafından yeni aboneyi dahil etmek için ağaç yapısını güncelleyen ilgili hedef aracıya yönlendirilir. Bir konu için bir olay yayınlandığında, olay doğrudan o konu için hedef aracıya yönlendirilerek, Pastry aracılığıyla olay yönlendirme gerçekleştirilir. Hedef komisyoncu, yalnızca konunun tanımlayıcısı tarafından ele alınır. Hedef aracıya bir olay ulaştığında, eşleştirme, doğru çok noktaya yayın ağacının belirlenmesine indirgenir ve bildirimin böyle bir ağaç aracılığıyla yayılmasıyla bildirim yönlendirmesi gerçekleştirilir.

Machanavajjhala ve diğ., (2008), tarafından sunulan hisse senedi fiyat bildirim sistemi çalışmasında yayıncılar alım satım fiyatları yayınlayan borsalardır. Aboneler, belirli bir değişim hacminin üzerindeki tüm teklifler ve en yüksek günlük varyasyona sahip teklifler gibi çeşitli ilgi alanlarına göre teklif almak isteyen yatırımcılar veya diğer finansal kuruluşlardır. Bir dizi ajans, yayıncılardan alınan teklifleri aboneler tarafından tanımlanan kısıtlamalara göre filtreleyerek yayın/alt hizmetleri sağlar. Bu bağlamda yayınlar kamuya açık verilerdir ancak abonelikler çok hassas bilgiler içerebilir. Gerçekten de, bir müşteriden kaynaklanan abonelikleri sızdırmak, yatırım stratejileri hakkında bilgi

verebilir ve rakipler tarafından kullanılabilir. Aboneliklerin aracıya gönderilmeden önce standart teknikler kullanılarak şifrelenmesinin yetersiz olduğuna dikkat çekiyoruz: yayınla/abone ol sistemi, bu aboneliklerde belirlenen kısıtlamalara dayalı olarak yayınları yönlendirebilmelidir. Kısıtlamalar, yayınla/abone ol sistemine tamamen opak olacak şekilde şifrelenirse, yönlendirme imkansızdır ve tüm filtrelemelerin abone tarafında yapılması gerekecektir. Bu nedenle, böyle bir sistemde gizliliğin sağlanması, yayınları doğru bir şekilde yönlendirme yeteneği ile bilgi sızdırma riski arasında bir uzlaşmadır.

E-Sağlık bilgi sistemleri (Ion ve diğ., 2010a; Eze ve diğ., 2010), içerik tabanlı yayınla/abone ol iletişim katmanı olarak yararlanabilen başka bir uygulamadır. Bu tür sistemler, kamu ve özel sağlık sektörlerinin aktörlerinin (doktorlar, hastaneler, klinikler, eczacılar) birbirleri iletişimde olmalarını sağlar. Bu aktörler, vakaların, testlerin vb. zamanında yayılmasını sağlamak için hastalarla ilgili dosyaları paylaşır. Tipik bir yayıncı, kritik durumdaki kişileri kabul eden bir acil durum birimi gönderebilir. Bu durumda yayınlar, tıbbi dosyalarının içeriği ile birlikte hastaların kimliklerini de içerir. Bu sağlık birimleri, yeni vakalar için harekete geçmek, hasta kabul ve tedavi seanslarını organize etmek ve planlamak için yayınla/abone ol sistemine abonelik gönderebilir. Yayının bir kısmı (tıbbi dosya) uçtan uca şifreleme kullanılarak şifrelenebilirken, diğer bazı kısımları yayının yetkili ve ilgili taraflar arasında yönlendirilmesi için kullanılmalıdır. Yayın başlıkları (adı, hastanın adresi, testin doğası vb.) oldukça hassas bilgilerdir. Abonelikler aynı zamanda oldukça hassas bilgilerdir: örneğin hangi hastanın hangi klinikte veya hangi tür rahatsızlık için tedavi edildiğini ortaya çıkarabilirler. Fakat bu tür bilgilerin sızması ciddi sonuçlara yol açabilir: Örneğin bir sigorta şirketinin bu tür verileri yasadışı şekilde aldığı bilgiler doğrultusunda belirli testlerden geçen hastaları kapsamayı reddettiği senaryo düşünülebilir. Ayrıca, e-Sağlık altyapısı, bir emniyet teşkilatının şüpheli bir kazanın kurbanları hakkında bilgi toplaması gibi, yayınla/abone ol iletişim yoluyla diğer sistemlerle bağlantı kurarsa, gizlilik yönetimi daha da karmaşık hale gelebilir. Bu çalışmada da veri sızıntısı durumunun ne kadar ciddi problemlere yol açabileceği görülebilir.

Bahsedilen bu iki senaryonun özelliği, yayınla/abone ol hizmet sağlayıcısının, yayıncıların ve abonelerinkinden farklı bir yönetim alanına ait üçüncü taraf olabilmesidir. Bu üçüncü tarafa hassas verilere erişme konusunda güvenilmeyebilir. Ayrıca,

sanallaştırmanın kullanımı ve genel bulutlarda kaynak yerleştirme ve iletişim üzerinde kontrol eksikliği ciddi gizlilik tehditleri oluşturabilir. Örneğin, araştırmalar hipervizör seviyesindeki istismarların olduğunu göstermiştir (Ristenpart ve diğ. ,2009; Somorovsky ve diğ., 2011) veya CPU önbellek düzeyinde (Liu ve diğ., 2015), sanal makineler bir araya getirildiğinde, genel bir bulut üzerinde çalışan bir sanal makineden özel bilgi toplamak için kullanılabilir. Destek altyapısı saldırılara açık olabileceğinden, yayınla/abone ol aracılar tarafından manipüle edilen bilgilere yetkisiz ve kazara erişim bile kritik veri sızıntılarına neden olmayacak şekilde tasarımla gizlilik sağlanmalıdır.

Zhao ve Sturman (2006), bir yayınla/abone ol sistemde erişim kontrolü sağlamak için bir hizmet modeli sunmuşlardır. Bu çalışma, erişim kurallarını çalışma zamanında değiştirme yeteneğine odaklanır. Birden çok yönetim alanından geçen yayınla/abone ol verilerine erişim, erişim denetimi aracılığıyla düzenlenir ve erişim denetimi ilkelerini değiştirmek için bir sürüm denetimi kullanılır. Bir güvenlik yöneticisi varlığı, yayınla/abone ol sistem istemcilerine verilen haklara ekleme veya değişiklikler yapma sorumluluğuna sahiptir. Abonelere ve yayıncılara, özel olarak yetkilendirilmedikçe yayınla/abone ol verileri üzerinde işlem yapma konusunda güvenilmez. Yayınla/abone ol istemcileri, asıl olarak kimliklerini başarılı bir şekilde doğrularlarsa bağlanma, yayınlama veya abone olma gibi belirli eylemler için haklar elde eder. Bu senkronizasyon, tüm sistem boyunca erişim kontrol kurallarının tutarlılığını garanti eder.

Zhang ve diğ. (2012) filtreleme tabanlı yayınlama/abone olma sistemlerinde, çok sayıda aboneliği destekleyen sistemdeki yüksek abonelik bakım maliyetlerinden muzdarip olduklarını vurgulamışlardır. Bu nedenle yazarlar tarafından sunulan bu çalışmada aboneliklerin birleştirilmesi önerilmektedir. Bununla birlikte, mevcut araştırmalar verimli ve pratik bir bağlantı mekanizmasından yoksundur. Mekanizma hem zaman hem de alan açısından verimlidir ve birleştirme detay seviyesini esnek bir şekilde kontrol edebilir. Bağlantı mekanizması, hem teorik hem de simülasyona dayalı değerlendirme yoluyla doğrulanmıştır.

Aslam ve diğ. (2021), düşük yanıt süresi ile sayısı bilinmeyen abonelikler için nesne algılama modellerinin çevrimiçi sınıflandırıcı yapısını kullanarak multimedya olay işleme için uyarlanabilir bir yaklaşım önermişlerdir. Önerilen model, mevcut nesne

algılama modelleri YOLOv3, SSD300 ve RetinaNet'in hiper parametrelerinin ayarlanmasıyla optimize edilmiştir. Deneyler, performans ve adaptasyon ile eğitim süresi arasındaki ödünleşimin, doğruluktan ödün vererek genel yanıt süresini azaltmak için yararlı olabileceğini göstermişlerdir.

The Open Platform Communication Unified Architecture protokolü gerçek zamanlı senaryolarda görüntü aktarımına yönelik çözümler sunabileceği, sağlam ve ölçeklenebilir kavramlar kullanarak uygulanabilir performanslar elde edebilmektedir. Alexandru Ioana ve diğ. sundukları çalışmada, daha yüksek veri hacimlerini dikkate almayı, çok kanallı User Datagram Protokolü (UDP) tabanlı iletişim ile uzun vadeli veri iletimi bağlamında geliştirilen mekanizmanın sağlamlığını analiz etmeyi amaçlamışlarıdır. Sonuç olarak, araştırmaları,

Melenli ve diğ. (2020), çalışmalarında görüntü işleme ve büyük veri yöntemlerini kullanarak Covid-19 sebebi ile kişiler arası sosyal mesafenin korunmasını gerçek zamanlı olarak gerçeklemişlerdir. OpenCV ve YOLOV3 yöntemleri ile insan tespitini gerçekleştirip kişiler arasındaki mesafeyi ölçmüşlerdir. Akan verinin işlenmesi için Apache Kafka ve Spark kullanılmıştır. Sosyal mesafenin çiğnendiği durumlarda da Oozie workflows ile email notifikasyonu gönderilmektedir. Ek olarak, kullanıcının kameraları tanımlamasını, bildirimler oluşturmasını veya gerçek zamanlı gösterge panellerine erişmesini sağlamak için bir kullanıcı arayüzü oluşturmuşlardır.

Kato ve diğ., (2019) Nesnelerin interneti kapsamında çeşitli sensörlerin yardıyla ve bulut bilişim teknolojilerinin kullanılması ile, sıradan evlerde çok sayıda canlı kaydın toplanmasına ve kullanılmasına yönelik çalışma geliştirmişlerdir. Araştırmalarında, sensörlerden iletilen veri miktarına göre hareketli görüntü tanıma görevlerinin ölçeklenebilir bir şekilde işlenmesini sağlayan yüksek verimli bir dağıtılmış akış işleme altyapısı için bir yapı şeması örnemişleridir. Önerilen dağıtık akış işleme altyapısının bir prototip sistemini dağıtık mesajlaşma sistemi olan Ray ve Apache Kafka kullanarak hayata geçiriyor ve performansını değerlendirmişleridir. Deneysel sonuçlar, önerilen dağıtılmış akış işleme altyapısının yüksek düzeyde ölçeklenebilir olduğunu göstermektedir.

Chen ve diğ. (2017) akan görüntü verilerinin işlenmesi için büyük veri platformunu kullanmışlardır. Verinin toplanmasında Apache Kafka'nın Yayınla/Abone ol modeli, verinin işlenmesinde Apache Spark ile OpenCV, HDFS ve HBase sistemleri de verinin depolanması sırasında kullanılmıştır.

Kim ve diğ. (2018), çalışmasında, kaynakları verimli bir şekilde tahsis eden ve dağıtılmış ortamlardaki akış verilerinin miktarına göre yük dengelemesi yapan RIDE (Real-time massive Image processing platform on Distributed Environment - Dağıtılmış Ortamda Gerçek Zamanlı Büyük Görüntü işleme platformu) adlı yeni bir platform sunmuşlardır. Hem Coarse-grained and Fine-grained paralelliği aynı anda dikkate alarak akış verilerini işleyen bir paralel işleme stratejisi kullanarak iletişim yükünü en aza indirdiğini göstermişlerdir. Coarse-grained paralellik, giriş akışlarının, her biri kendi ilgili çalışan düğümü tarafından işlenen aracı arabelleğinin bölümlerine otomatik olarak tahsis edilmesiyle elde edilir ve bir gruptaki çalışan düğümlerin sayısını gerçek zamanlı olarak kare hızına göre ayarlayan uyarlanabilir kaynak yönetimi ile maksimize edildiği; Fine-grained paralellik, her çalışan düğümde görevin paralel işlenmesiyle elde edilir ve GPU ve gömülü makineler gibi heterojen kaynaklar uygun şekilde tahsis edilerek en üst düzeye çıkarıldığını göstermişlerdir. Sistmeleri bileşenler arasındaki koordinasyon yoluyla gerçek zamanlı görüntü işleme sırasında dinamik hata toleransını desteklemektedir.

Hoque ve Miranskyy (2018), akış verileri üzerinde analiz işlemi gerçekleştirmek için ölçeklenebilir ve sürdürülebilir bir mimari geliştirmişlerdir ve birden çok katmana yayılmış, mikro hizmetlerden ve yayınla-abone ol modelinden (eşzamansız bir iletişim merkezi) oluşan 7 katmanlı bir mimari önermişlerdir. Mimari, kullanım durumlarını karşılamak için verilerin analizine yönelik farklı kullanım durumlarına uyan bir veri alt kümesinin tanımlanması için girdi olarak çeşitli kaynaklardan heterojen akış verilerini kullanmaktadır.

Kul ve diğ. (2017), akıllı trafik yönetim sistemleri için, gerçek zamanlı olarak araçları boyutuna göre sınıflandırılan bir yayınla/abone ol sistemi sunmuşlardır. Görüntüler trafik güvenlik kameralarından gelmektedir. Video çerçevelerinden araç tespit işlemi gerçekleştirilir. Video karelerinden araç nesnelerini çıkarmak için, hareketli bir arka plan/ön plan çıkarma algoritması tarafından algılanır. Tespit edilen araçların geometrik

öznitelikleri (genişlik, uzunluk, yükseklik) ve ikili öznitelikleri (alan oranı, en boy oranı, ana ve yan eksen uzunluğu, elips çapı gibi) çıkarılarak Destek Vektör Makinesi (SVM), Yapay Sinir Ağı (ANN) ve AdaBoost algoritmaları ile sınıflandırılır. Araçlar üç sınıfta kategorize edilir: küçük (sedan, otomobil gibi), orta (minibüs, minivan) ve büyük (otobüs,kamyon). Daha sonra tespit edilen araçlar Tomcat sunucu tabanlı Java Message Service (JMS) kullanılarak konulara yayınlanır. Konular makine öğrenmesinden elde edilen etiketli sınıflar olarak belirlenir.

Akıllı trafik yönetim sistemleri için, gerçekleştirdiğimiz bir başka çalışma da Kul ve diğ., (2021) tarafından sunulan Event-Based Microservices With Apache Kafka Streams: A Real-Time Vehicle Detection System Based on Type, Color, and Speed Attributes çalışmasıdır. Bu makalede sunulan çalışma, işbirliği yapan mikroservis tabanlı trafik gözetleme kameraları tarafından yayınlanan video akışları üzerinden belirli bir aracı izlemeye yönelik yeni bir yaklaşım önermektedir. Yazarlar, istenilen sorgu üzerinden video parçalarını alan, ölçeklenebilir, hataya dayanıklı ve esnek bir trafik izleme sistemi geliştirmenin zorluğunu belirtmişlerdir. Bu makale, yayınla/abone ol modeline dayalı olarak anahtar/değer sorgusu ile video parçalarını alma sorununu araştırmaktadır. Bu nedenle, Olay Tabanlı Mikroservis çerçevesi için senkron ve asenkron iletişim mekanizmasının bir melezini önermişlerdir. Yayınla/abone ol mimarisinin implementi için Apache Kafka çerçevesi kullanılmıştır. Önerilen çerçevede, öncelikle mikroservisler araçları algılar ve araçların tip, renk ve hız özelliklerini çıkararak metadata deposunda depolar. Mikroservisler her özelliği olay olarak yayınlar. Diğer mikroservisler kendi kendine bu olaylara abone olur, ve tüm olasılıkları birleştirerek daha fazla olayın yayınlanmasına yol açar: tip-renk, tip-hız, renk-hız ve tip- renk hızı. Deneysel sonuçlar, önerilen sistemin mesajları gerçek zamanlı olarak filtrelediğini ve yeni mikroservisler mevcut sistemle entegrasyonunu desteklediğini göstermektedir.

Yayınla/abone ol artık büyük ölçüde dağıtılmış etkileşim için en ilgi çekici paradigmalardan biri olarak kabul edilmektedir. Bununla birlikte, bir yayınla/abone ol sistemin olumlu özellikleri (ölçeklenebilirlik gibi) doğrudan paradigmadan miras alınmaz, ancak belirli mimari ve algoritmik çözümler tarafından uygulanması gerekir. Bu gözlemi takiben, tez kapsamında sunulan mimari dağıtık olay yönlendirme için önerilen büyük

miktarda araştırma çalışmasını eleştirel olarak analiz etmeyi, özellikle etkileşim ve bağımlılıklar açısından farklı çözümlerin kritik yönlerine dikkat çekmeyi amaçlamaktadır.

2.4. Mikroservis Mimarisi Kullanım Alanları

Mikroservis mimarisi (Lewis ve Fowler, 2014), her biri kendi sürecinde çalışan ve genellikle bir HTTP kaynak API'si olan hafif mekanizmalarla iletişim kuran küçük hizmetler paketi olarak tek bir uygulama geliştirmeye yönelik mimari bir stil ve yaklaşımdır. Mikroservis mimarisi, her bir mikro hizmetin bağımsız olarak geliştirilmesini, dağıtılmasını, yükseltilmesini ve ölçeklenmesini sağlar. Bu nedenle, özellikle bulut altyapıları üzerinde çalışan ve bileşenlerinin sık sık güncellenmesini ve ölçeklenmesini gerektiren sistemler için uygundur.

Günümüzün modern web tarayıcılarının hızlı yayın döngüleri ile WebSocket ve WebRTC gibi gerçek zamanlı protokoller ve standartlar artık geniş çapta desteklenmektedir ve belge yönetimini, ses ve video akışını kolaylaştırmak için içerik odaklı çevrimiçi işbirliği araçlarının geliştirilmesine; iş grupları arasında görev koordinasyonu ve iletişimini de olanak sağlamıştır. Son yıllarda derin öğrenme mimarilerinin kullanımının yaygınlaşması, kütüphane ve donanım sürücülerinin versiyon uyum problemlerinin yaşanması gibi durumlar araştırmacıları sanallaştırma ve uygulamayı ortamdan bağımsızlaştırma çalışmalarına yönlenmesini sağlamıştır. Mikro hizmetlerin endüstrideki yaygınlığına ve önemine rağmen, araştırma topluluğunda mikroservisler üzerine sadece birkaç makale ve hatta büyük konferanslarda daha az sayıda makale ile konuyla ilgili sınırlı araştırma bulunmaktadır. Mevcut araştırma, tasarım (Hassan ve Bahsoon, 2016), test (Heorhiadi ve diğ., 2016; Schermann ve diğ., 2016; Camargo ve diğ., 2016; Heinrich ve diğ., 2017), dağıtım (Leitner ve diğ., 2016; Hasselbring 2016; Klock ve diğ., 2017) doğrulama dahil olmak üzere mikroservis hakkında çok çeşitli konulara odaklanmaktadır. Mikroservis sistemlerinin hata analizi ve hata ayıklaması hakkında çok az araştırma bulunmaktadır.

Mikroservis mimarisi, küçük hafif mikro hizmetlerin bileşenleştirilmesine, çevik ve DevOps uygulamalarının uygulanmasına, merkezi olmayan veri yönetimine ve mikro hizmetler arasında yönetişime odaklanan daha geniş Hizmet Odaklı Mimari (SOA) alanından doğar. Monolitik mimariden mikro hizmet mimarisine geçişle birlikte mimari karmaşıklık, kod tabanlı koddan mikroservislerin etkileşimlerine geçer. Farklı

mikroservisler arasındaki etkileşimler, ağ iletişimi kullanılarak gerçekleştirilmelidir. Mikroservislerin çağrıları senkron veya asenkron olabilir. Senkronize çağrılar, kesinti süresinin etkisi nedeniyle zararlı olarak kabul edilir. Asenkron çağrılar, asenkron REST çağrıları veya mesaj kuyrukları kullanılarak uygulanabilir. Birincisi daha iyi performans sağlarken ikincisi daha iyi güvenilirlik sağlar. Bir kullanıcı isteği genellikle çok sayıda mikroservis çağrısı içerdiğinden ve her bir mikroservis başarısız olabileceği göz önüne alındığında, mikroservislerin buna göre, yani mikroservisçağrılarındaki olası başarısızlıkları hesaba katarak tasarlanması gerekir.

Mikroservis mimarisi, bir dizi altyapı sistemi ve tekniği ile desteklenir. Spring Boot (URL-9) ve Dubbo (URL-10) gibi mikroservis geliştirme çerçeveleri, REST istemcisi, veritabanı entegrasyonu, harici yapılandırma ve önbelleğe alma gibi ortak işlevler sağlayarak sistemlerinin geliştirilmesini kolaylaştırır. Mikroservis sistemleri, taşınabilirlik, esneklik, verimlilik ve hız için konteyner (örneğin, Docker (URL-11)) tabanlı dağıtım kullanır. Konteynerlar, Spring Cloud (URL-12), Mesos (URL-13), Kubernetes (URL-14) ve Docker Swarm (URL-15) gibi çalışma zamanı altyapı çerçeveleri kullanılarak yapılandırma yönetimi, hizmet keşfi, hizmet kaydı, yük dengeleme ile kümeler tarafından organize edilebilir ve yönetilebilir.

Mimarinin omurgası olan mesaj yolu, yazılım ve donanımdaki bileşenler arasındaki iletişimi kolaylaştırır. Kararlılık ve güvenlik açısından güvenilir olmalı, gerekli iletişim şemalarını desteklemeli ve kullanımı kolay olmalıdır. Görüntü işleme, yapay zeka alanlarında yapılan mikroservis ile geliştirilen çalışmalar aşağıdaki gibi sıralanabilir.

Wu ve diğ. (2020) derin öğrenme modellerinin daha iyi kullanılması ve birden fazla veri kaynağı için model performansının daha hızlı yeniden üretilmesi için Docker tabanlı bir yöntem sunarak, giderek daha fazla biyomedikal bilim insanının yeni teknolojiyi kendi alanlarında uygun bir şekilde kullanılmasını sağlamışlardır. Geliştirdikleri sistemi DDeep3M olarak adlandırmışlardır. DDeep3M, görüntü segmentasyonu için beynindeki 3D optik mikroskopi görüntü kullanılır. 0.96'nın üzerindeki tüm geri çağırma/hassasiyet puanları ile yüksek doğruluk elde etmişlerdir.

Kim ve diğ., (2020) DiPLIP adı verilen derin öğrenme modeli çıkarımına dayalı gerçek zamanlı büyük ölçekli görüntü işleme için dağıtılmış paralel işleme platformu

sunmuşlardır. Tampon katmanı ve akış görüntüsünün boyutuna göre ölçeklenebilir bir paralel işleme ortamı kullanarak büyük ölçekli gerçek zamanlı görüntü çıkarımı için bir şema sağlamaktadır. Kullanıcıların, sanal makine kapsayıcısının dağıtımı yoluyla dağıtılmış bir paralel işleme ortamında yüksek hızlarda gerçek zamanlı görüntüleri işlemek için eğitimli derin öğrenme modellerini kolayca işlemesine olanak tanıyan sistem önermişlerdir.

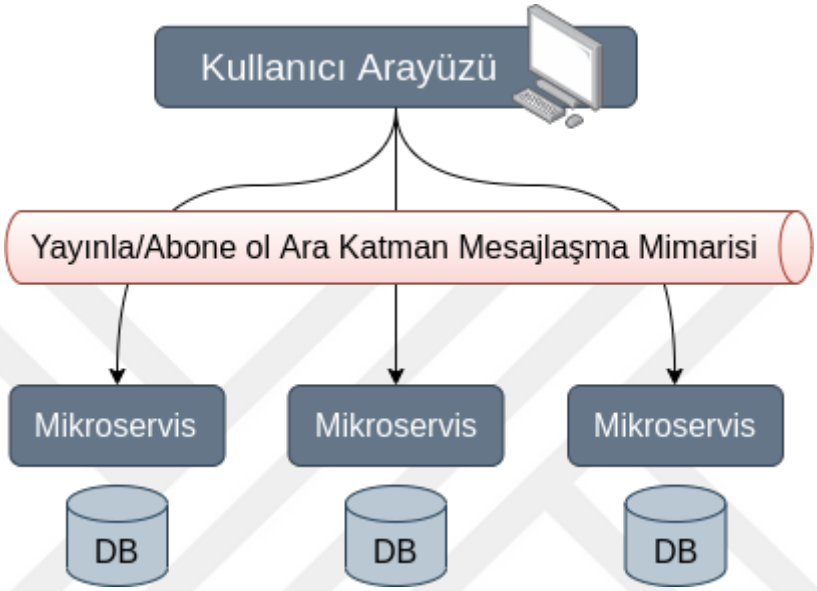
Zendi (2020), bulut hizmetlerinde HTTP, TCP, LwM2M vb. protokoller üzerinden erişilebilen dağıtılmış bir robot kontrol sistemi olarak dağıtılmış bir derin sinir ağı (DDNN) önermiştir. Containerized DNN mimarisiyle geliştirilebilen tek bir düğümde azaltmak için mikro hizmet tabanlı DNN sunulmuştur. Önerilen yöntem, C1-C4 numaralı konteynırlarda eğitilmiştir. 13.7279 saniye boyunca C1'den C4'e kadar olan konteynırlarda gerçekleştirilen DNN'nin eğitim doğruluğu %95,76'ya ve kayıp değeri %1,06'ya eşittir. Önerilen yöntem, 100 IK hareket verisini Socket.IO üzerinden 1,7 kB veriyi 336 milisaniyede ilettiği gözlemlenmiştir.

Gadea ve diğ. (2016), mikroservis mimarisine dayanan, ölçeklenebilir metin düzenleyicisi önermişlerdir. Sistemleri, aynı zamanda metni düzenleyen kişinin yüzünü de algılamaktadır. Mimari, mevcut bilgisayar ağı ve diğer bilgi işlem cihazları arasında sorunsuz dağıtım sağlayan ayrı konteyner olarak bireysel hizmetlerin geliştirilmesine ve test edilmesine izin vermek için Docker'ı kullanmaktadır. Mikro hizmetlerin, birden çok kullanıcının, yüzlerini içeren görüntülerinin eklendiği ve belge içeriğinin bir parçası olarak tanındığı ve belgenin düzenlenmesini sağladığı sistem gerçeklenmiştir.

Mikroservis mimarisinin kullanımı ve hizmetlerin nasıl geliştirileceği büyük önem taşısa da mikro servislerin nasıl iletişim kuracağı da bir o kadar önemlidir. Bu bölümün geri kalanı, Şekil 2.5'te gösterildiği gibi mesajları mikroservislere iletmek için yayınla/abone ol yaklaşımını kullanan çalışmalara odaklanmaktadır.

Yayınla/abone ol ile ilgili yapılan çalışmalar başlığı altında da bahsedilen Kul ve diğ. (2021) tarafından gerçeklenen yayınlama/abone olma modeli ve bir anahtar/değer sorgusu kullanarak video parçalarını döndüren bu çalışmada, microservis çerçevesi için asenkron ve senkronize iletişim mekanizmalarının bir kombinasyonu önerilmiştir. Mevcut platformlardan daha fazla yararlanmak için genel stratejiler oluşturulması

hedeflenmiştir. (i) İlk olarak, mikro hizmetler araçları tanımlar ve daha sonra önerilen çerçevede meta veri deposuna kaydedilen tür, renk ve hız özelliklerini çıkarır. (ii) Mikro hizmetler, her özelliği bir etkinlik olarak kullanılabilir hale getirir. (iii) Diğer mikro hizmetler, tüm olasılıkları birleştirerek daha fazla etkinliğin yayınlanmasını sağlarlar. Haberleşme için Apache Kafka kullanılmıştır.



Şekil 2.5. Mikroservis örnek diyagramı

Alaasam ve diğ. (2019) Apache Kafka Stream API (Kafka stream DSL) kullanılarak gerçek zamanlı üretim verisi işleme ile görevli olan mikro hizmetlerin nasıl oluşturulduğuna dair bir vaka çalışması sunmuşlardır. Ayrıca, hata toleransı, işleme gecikmesi ve ölçeklenebilirlik gibi gereksinimleri karşılayarak Dijital İkiz uygulamasında durum bilgisi olan akış işlemenin nasıl sağlanacağını açıklamışlardır. Çalışmalarının sonuçları, durum bilgisi olan mikro hizmetler oluşturmak için Kafka Stream DSL kullanmanın, durumu yönetmek için makul bir yol ve Dijital İkizde akış işlemeyi desteklemek için kabul edilebilir bir gecikme süresi sağladığını göstermektedir. Ancak, durumu depolamak için ara Kafka konuları ile veri alışverişini yönetmek için oluşturulan trafik miktarı, durum Kafka kümesinde tutulduğu sürece gecikmeyi etkiler.

Yayınla/abone ol mesajlaşma modeli, güçlü ancak basit bir iletişim aracıdır. Dahili ve harici bileşenler arasındaki tüm iletişimi yöneterek, gerçek zamanlı dağıtılmış mikro hizmet tabanlı uygulamalar için temel görevi görür. Büyük ölçekli uygulamalar için mikro hizmetler en iyi seçenektir. Öte yandan, daha küçük programlar genellikle

monolitik bir kod tabanıyla daha iyi çalışabilmektedir. Bağımsız mikro hizmetleri geliştirmek ve sürdürmek kolaydır. Olaya dayalı iletişim kullanıldığında, bir mikro hizmet, daha önce açıklandığı gibi kayda değer herhangi bir şey olduğunda bir olayı yayınlar. Bu olaylara diğer mikro hizmetler tarafından abone olunur. Bir mikro hizmet bir olay aldığında, kendi varlıklarını güncelleyebilir ve belki de yeni olayların yayınlanmasına neden olabilir. Nihai tutarlılık ilkesi buna bağlıdır. Apache Kafka'nın, bileşenler ve iletişim arasındaki sıkı bağlantı (tightly coupled) sorununu çözdüğü için, birden fazla alıcıyla eşzamansız iletişim gerektiğinde umut verici bir yol olduğu gözlemlenmiştir.

2.5. Mikroservis Mimarisi Veri Güvenliği

Artan iletişim taleplerini karşılamak için en umut verici iletişim paradigması, göndericileri ve alıcıları merkezi bir mesaj aracısı kullanarak ayırarak çoktan çoğa iletişim sistemlerinin karmaşıklığını önemli ölçüde azaltan Yayınla/Abone Ol (PubSub) dır (Patrick ve diğ., 2003). Bununla birlikte, iletilen veriler genellikle son derece hassas (Gaove diğ. 2014; Pennekamp ve diğ. 2020) ve gizlilik ve güvenlik gereksinimlerini karşılamak için güvenli iletişim kaçınılmaz olsa da, PubSub güvenli bağlantıların kurulmasını zorlaştırmaktadır. Geleneksel uçtan uca (E2E) güvenli protokollerin, örneğin TLS'ye veya OPC UA Binary Protokolüne dayalı olarak kullanılabildiği bire bir iletişimin aksine, çoktan çoğa iletişim, ikiden fazla ortağın temel materyali üzerine karar vermesini gerektirir ve bu da yerleşik el sıkışma prosedürlerini imkansız hale getirir. Sonuç olarak, merkezi bileşen olarak mesaj broker, PubSub tabanlı endüstriyel iletişimde güvenlik için tehlikeli bir zayıf nokta oluşturur.

Bu güvenlik sorunlarını ele almak için birçok PubSub protokolü, aracıların (broker) belirli konulara erişimi kimliği doğrulanmış ve yetkili istemcilerle sınırlandırmasına olanak tanıyan kimlik doğrulama mekanizmaları (URL-20) sağlar. Bununla birlikte, son araştırmalar, ağırlıklı olarak kullanılan PubSub protokollerinden biri olan MQTT'yi uygulayan İnternet üzerinden erişilebilen birçok aracının erişim kontrolünü doğru şekilde uygulamadığını göstermiştir (URL-18) Bu aracılar, İnternet'teki herkesin iletilen verileri almasına ve iletişime daha fazla koruma uygulanmadığından potansiyel olarak mesajları enjekte etmesine izin verir. Bu istenmeyen durum, herhangi bir yapılandırma kusuru

güvenliği doğrudan etkilediğinden, yanlış yapılandırılmış aracılara karşı koruma ihtiyacını vurgular.

Aracıların (broker) tam olarak güvenli bir şekilde uygulanması ve yapılandırılması varsayılabilir olsa bile, kötü niyetli veya güvenliği ihlal edilmiş bir aracı durumunda, hem gizlilik hem de güvenlik tehlikededir. Bu nedenle, kötü niyetli bir aracı ve ilgili saldırganlar, iletilen her mesajı gizlice dinleyebilir, hatta bu süreçleri kontrol etmek ve rahatsız etmek için mesajları değiştirebilir, enjekte edebilir veya bırakabilir.

Kriptografi, günümüz toplumunda bilgi güvenliğinin önemli bir yönüdür. Kriptografi, bilgiyi dönüştürme ve şifreli konuşmalar yapmak için kodları kullanma tekniğidir, böylece sadece bilginin amaçlandığı kişiler işlenir. Kriptografik algoritmalar çeşitli şekillerde kullanılabilir. İdeal bir dünyada, düşük maliyetli, yüksek performanslı bir şifreleme yöntemine ihtiyaç duyulacaktır. Yaygın olarak kullanılan 3DES, AES, Blowfish, CAST5, RC4 ve RSA kriptografik algoritmalarının maliyet ve performansları yayınla/abone ol iletişim yöntemini güvence altına almak için uyguladı ve bu sebeple bu bölümde bu algoritmaların yayınla/abone ol sistemlerinde kullanımının literatürdeki yerlerine değinilmiştir. Yaklaşımların çoğu, gizli bir anahtarla uçtan uca şifrelemeye dayanır.

Narada Brokering sistemi bağlamında, Pallickara ve diğ. (2003), iletişimin E2E korumalı olduğu güvenli konular aracılığıyla istemcilerin iletişim kurmasını sağlayan bir çerçeve önermektedir. Bu amaçla, bir konu sahibi adına, anahtar yönetim merkezleri, şifreleme anahtarlarını ve erişim belirteçlerini yönetir ve bunları yetkili istemcilere gönderir. Anahtar yönetim merkezleri, diğer (kötü amaçlı) varlıkların gizlice dinlemesini ve mesajları değiştirilmesini önleyen müşteri sertifikalarını kontrol ederek yetkili müşterilere yalnızca anahtar materyali ve erişim belirteçlerini sağlar. Ayrıca tasarımda, göndericiler ve alıcılar hala ayrıdır.

Singh ve diğ. (2015), attributed-based encryption (ABE) kullanarak E2E güvenliğini sağlar. Burada aracı (veya ayrı bir anahtar oluşturma sistemi), sahip oldukları ABE niteliklerine bağlı olarak anahtar sırları yayıncılara ve abonelere dağıtır. Böylece, göndericilerin ve alıcıların ayrılması tasarımla desteklenir. Aracı, anahtar yönetimi için ayrı bir örnekle iletilen mesajlara erişip bunları değiştirebildiğinde bilgi güvenliği

gereksinimini ihlal edecek olsa da, SMQTT kötü niyetli aracıların mesajları değiştirmesini veya gizlice dinlemesini önleyebilir. Bu uygulama kararı, konuşlandırılabilirlik gereksinimini de etkiler. Şimdiye kadar, SMQTT, bırakılan veya yinelenen mesajlara karşı herhangi bir önlem almamıştır.

Borcea ve diğ. (2017), vekil yeniden şifrelemenin, aracının, yayıncılardan alınan mesajları her abone için bağımsız olarak, yani aracının kendisi bir mesajın şifresini çözemeden, habersiz bir şekilde yeniden şifrelemesini sağladığı bir tasarım önermektedir. Bu şema, gönderici/alıcı ayrıştırmasını gerçekleştirebilir ve isteğe bağlı güvenli yan kanallar (güvenilir bir üçüncü taraf kullanılarak gerçekleştirilir) sunarak, ek bir erişilebilir hizmete ihtiyaç duymadan gerekli anahtar materyali dağıtmayı hedefler. Ancak, aracıya, bağlı her abone için her mesajı yeniden şifrelemesi gereken büyük bir ek yük ekler. Ayrıca, mesajların düşmesini veya yinelenmesini tespit etmek için herhangi bir mekanizma sağlamaz ve mesaj bütünlüğünü garanti etmez.

Khurana (2005), güvenli XML belge dağıtımı için bir yayınla/abone ol sistemi hedefleyen vekil yeniden şifrelemeye dayanan bir güvenlik modeli sunar. Tehdit modeli, aracıların yayınla/abone ol iletilerinin hassas bölümlerine erişme konusunda güvenilir olmadığını ve yönlendirmenin yalnızca hassas olmayan alanlar kullanılarak gerçekleştirildiğini varsayar. Sistem, bir proxy güvenlik ve muhasebe hizmeti (PSAS) barındıran güvenilir sunucuların varlığını gerektirir. İşlevsellik. Yayınların yalnızca hassas ve yönlendirilemez kısımları şifrelenir. YBir yayın şifreleme anahtarı ayrıca yayıncıya ait bir genel anahtarla şifrelenir ve şifrelenmiş yayınla birlikte gönderilir. PSAS sunucuları, simetrik yayın şifreleme anahtarını dağıtmaktan sorumludur. PSAS ayrıca, bir yayıncının açık anahtar ile şifrelenmiş mesajları abonelerin açık anahtarlarıyla şifrelenmiş mesajlara dönüştürerek aracı isteklerine göre hareket eder. Bu, kaynak ve hedefler arasında doğrudan anahtar alışverişini önler, ancak hizmet kullanılabilirliğini sağlamak için sürekli olarak bir PSAS sunucusu çoğunluğunun kullanılabilir olmasını gerektirir. Sonuç olarak, bu PSAS sunucuları tek hata noktası oluşturur ve çözümün ölçeklenebilirliğini bozabilir. Gizlilik yalnızca yayınlar için dikkate alınır ve yalnızca yönlendirilemez nitelikler için uygulanır. Yayıncılar, yayın yüklerinin nasıl korunması gerektiğini özel olarak belirtmezler, ancak bunların, yönlendirilemez öznitelikleri şifrelemek için kullanılan

anahtar kullanılarak şifrelenebileceğini varsayılır. Önerdikleri çözümde yayıncı ve abone arasında herhangi bir anahtar paylaşımı olmadan veri paylaşımı gerçekleşmektedir.

Abone ve yayıncının gizli bir anahtar kullanarak, uçtan uca şifreleme ile haberleştiği başka bir teknik Chen ve diğ. tarafından incelenmiştir. yayınla/abone ol altyapısı güvenilir olmadığında, hem olayları hem de ilgi alanlarını yayınla/abone ol ağdan gizli tutmak hedeflenmiştir. Sundukları algoritmalar, n uzunluğunda bir mesaj için O(n) şifreleme işlemleri gerektiren simetrik şifrelemelere dayanmaları bakımından verimli olduğu gözlemlenmiştir.

Opyrchal ve diğ. (2001), abonelerin ve yayıncıların simetrik çift anahtarları paylaşmasına izin veren güvenli sistemler önermektedir. Yayıncı ve abone arasında doğrudan temas olmaması nedeniyle yayınla/abone ol sistemlerinde anahtar paylaşmanın zorluklarını not etmişler ve standart tekniklerin yetersiz olduğunu iddia etmişlerdir.

Srivatsa ve diğ. (2007), yayınla/abone ol iletişim gizliliğini sağlayan bir anahtar yönetim yöntemi önermektedir. Yayıncılar, güvenilir bir merkezi otoriteden şifreleme anahtarlarını ve aboneler de yetkilendirme anahtarlarını alır. Anahtarlar, aralık eşleşmesini kolaylaştırmak için hiyerarşik bir şekilde düzenlenir ve her anahtar bir aralığa karşılık gelir. Bir filtre, bir yetkilendirme anahtarı ile ilgilidir. İlgili çalışmanın sonucu incelendiğinde, performansın anahtar paylaşım mekanizması kadar önemli olduğu görülmektedir.

Fiege ve diğ. (2004), aynı güven düzeyine ve yetkilere sahip varlık gruplarının tanımlanmasına izin veren, kapsamlara dayalı bir güvenlik çözümü sunar. Yayıncılar, aboneler ve aracılar, kapsam adı verilen farklı gruplarda birleştirilir. Bir kapsam içindeki iletişim yalıtılmıştır. Yeni üyelerin bir kapsama kabulü, kapsam kabul kriterlerine göre kimlik doğrulaması gerektirir. Kapsamların tanımı, farklı yönetim alanları arasında kurulan ortak güven ilişkilerine bağlıdır. Bir kapsam, özyinelemeli olarak diğer kapsamların bir üyesi olabilir. Kapsamlar, erişim kontrol ilkelerinin kullanılması yoluyla üyelerine iletilerin görünürlüğünü sınırlamak için kullanılır. Bunlar, sistemdeki varlıklara ait bir dizi öznitelikle ilişkili imzalı bir kimlikten oluşan kimlik bilgilerini temsil eder. Öznitelik sertifikaları ya bir öznitelik yetkilisi tarafından ya da belirli bir kapsama ait olan aracı ağının sahibi tarafından verilir. Düğümler kimlik bilgilerini alır ve bir yayın türünün

yayınını yaparken veya abonelik göndermeye çalıştıklarında bunları kullanır. Aracılar kimlik bilgilerini kontrol eder ve kapsam içinde yayılmasına izin verebilir. Bir aracı, bir abonenin sertifikasındaki öznitelikleri, yayıncının daha önce aldığı reklamlardaki özelliklerle eşleştirmeye çalışır. Bu kontrol başarısız olursa, abonelikler işleme alınmayacaktır. Gizlilik özelliği eserde özel olarak ele alınmamıştır. Ancak erişim kontrol mekanizmaları, güven kapsamı dışında kalan herhangi bir düğüme karşı gizlilik sağlar. Bu, koruma amaçlı mesajların içeriğini değiştirmez, yalnızca yayılmasını kısıtlar.

Wun ve Jacobsen (2007), yayınla/abone ol mimarileri için genel hizmetler sunan bir politika yönetimi çerçevesi sunar. Bir ilke, kurallar tetiklendiğinde yürütülecek eylemleri tanımlayan koşula dayalı kurallar içerir. Kurallar, yayınlar ile abonelikler eşleştirilirken veya hemen sonrasında tetiklenebilir. Ana amaç, anlamsal olarak yayın içeriğine dayanan ilke kurallarını kullanırken eşleştirme işlevselliğini çoğaltacak olan kuralları önceden test etmekten kaçınmaktır. Makale, basit bir güven grupları ayarını ele almaktadır. Güven gruplarını oluşturan varlıklar (yayıncı, abone, aracı) daha sonra tartışıldığı gibi etkileşime girer. Güvenlik zorlaması için yazarlar, her güven grubunun paylaşılan bir grup sırrıyla ilişkilendirildiğini bir ön koşul olarak varsayar. Bu gizli bilgiler, grupla ilgili belirli protokoller kullanılarak kimlik doğrulama ve şifreleme için yayınla/abone ol varlıkları tarafından kullanılır. İlke yönetimi çerçevesi, kimliği doğrulanmış olay kapsamlarını ve güvenlik bölgelerini kullanarak gizliliği korur.

Bu bölümde incelenen çalışmanın önemli bir sınırlaması vardır: güvenilmeyen alanlardaki aracıların (broker) söz konusu alanlardaki hassas alanları kullanarak yayınları yönlendirmesine izin vermez. Örnek verilen sistemlerin nihai olarak aşağıdaki iki çözümden birine dayanması gerekiyor: ya tüm iletiler, gerekli yönlendirilebilir alanların hassas olmadığı hedef etki alanında taşmalı ve filtrelenmelidir ya da tüm abonelikler, abone etki alanlarından tüm yayıncılara çoğaltılmalıdır. Bu baypas çözümleri, ya bant genişliği israfı olacaktır ve zayıf ölçeklenebilirliğe nedeol olabilir ya da artan yönetim karmaşıklığı, üreticiler ve bilgi tüketicileri arasındaki ayrışmanın kaybı ve depolama ek yükü anlamına gelir.

İncelenen çalışmalarda, güvenilir olmayan ağlarda, özellikle güvenilir olmayan Broker ile iletişim sırasında var olan güvenlik açıklarından bahsedilmiş ve şifrelemenin önemi

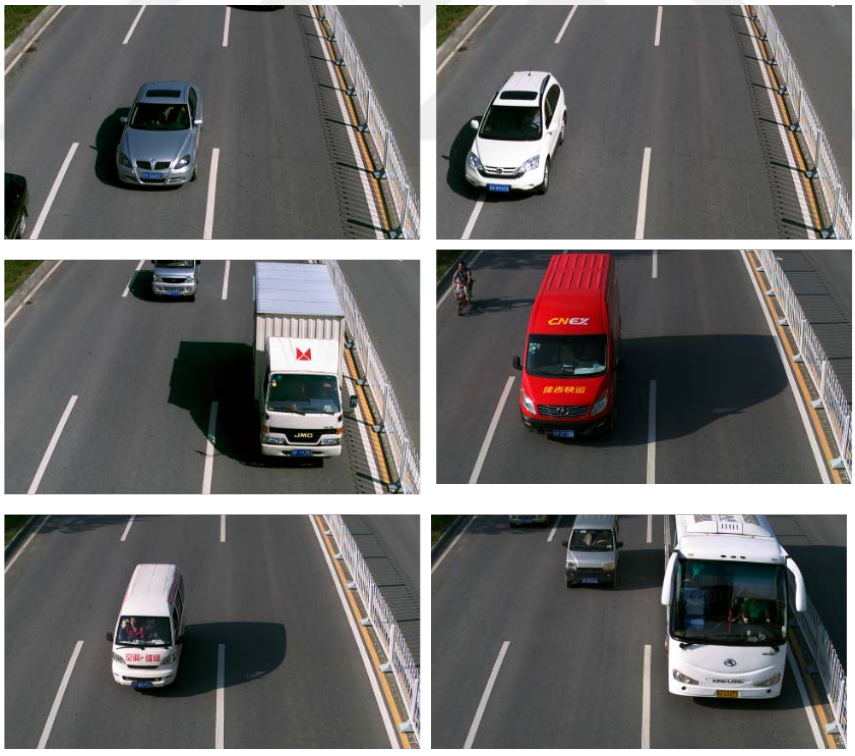
belirlenmiştir. Simetrik ve asimetrik anahtarların kullanımında nasıl bir avantaj ve dezavantaj yaratacağı da gözlemlenmiştir. Kullanılacak anahtar yapısının detayları Yöntem başlığı altında verilmiştir.

Yapılan çalışmalar incelendiğinde, yapay zeka tabanlı, mikroservis mimarisine dayalı, yayınla-abone ol haberleşme kanalı ile haberleşerek katalog-kayıt şeklinde bir çalışma olmadığı gözlemlenmiştir.

3. KULLANILAN VERİ SETLERİ

Filtreleme sırasında araca ait üç özellik seçilmiştir bunlar: tip, renk ve hızdır. Araç tipini belirlemede BIT-vehicle veri kümesi (Dong ve diğ., 2015), araç rengini belirlemede (Chen ve diğ., 2004) ve araç hızını (Luvizon ve diğ., 2014) belirlemede veri kümesi kullanılmıştır.

Pekin Teknoloji Enstitüsü Üniversitesi Medya Hesaplama ve Akıllı Sistemler Laboratuvarı tarafından geliştirilen BIT-vehicle veri kümesi araç tiplerinin belirlenmesinde tercih edildi. Farklı aydınlatma koşullarında (veri kümesinin yaklaşık %10'u gece), farklı çözünürlüklerde (1600x1200 ve 1920x1080) ve farklı görüş açılarında 9.850 araç görüntüsü içerir. Tüm araç görselleri altı kategoride sunulmaktadır: Sedan, SUV, mikrobüs, kamyon, otobüs ve minivan. Her tür için resim sayısı sırasıyla 5922, 1392, 883, 822, 558 ve 476'dır. Şekil 3.1, BIT Araç veri kümesinin bazı örnek görüntülerini içermektedir.



Şekil 3.1. BIT Veri setinden örnek görüntüler

Chen ve diğ. (2014) tarafından yayınlanan veri kümesi. araç renklerini tespit etmek için kullanıldı. Bu veri kümesinde sadece farklı renklere ait araçların görüntüleri bulunmaktadır.

Bunlar 3.442 siyah, 1.086 mavi, 281 camgöbeği, 3.046 gri, 482 yeşil, 1.941 kırmızı, 4.742 beyaz ve 581 sarıdır. Veri seti dengesiz (unbalance) olduğundan dolayı Image Data Generator (Chollet, 2015) kullanıldı. Eğitim görüntülerini farklı işleme yöntemleri veya rastgele döndürme, kaydırma, kesme ve döndürme gibi çoklu işlemler yoluyla yapay olarak oluşturur. Böylece 3442 siyah, 3027 mavi, 2939 gri, 3046 yeşil, 3101 kırmızı, 3056 beyaz ve 4742 sarı olmak üzere eğitim verilerini elde ettik. Şekil 3.2., veri kümesinin bazı örnek görüntülerini içermektedir.

Şekil 3.2. Araç renk veri kümesinden örnek görüntüler

Veri setlerinde yeşil ve gri olan varyanslar diğerlerinden çok daha büyüktür. Bu iki renk daha az birleşik olduğundan, özellikleri özellik uzayında daha dağınıktır ve sınıflandırılması zordur. Bu nedenle, değerlendirilen tüm yöntemlerde bu iki rengin

tanınma oranları diğer renk türlerine göre daha düşük olabilmektedir. Veri seti, sekiz renk tipini kapsayan araç görüntüsü içermektedir.

Görüntüler, 1920 × 1080 çözünürlüğe sahip bir HD kamera tarafından kabaca önden görünümden çekilmiştir. Her görüntü, bir araç dedektörü kullanılarak lokalize edilmiş ve kırpılarak yalnızca bir araç içermektedir. Veri seti, hava durumu, aydınlatma, bakış açısı ve araç tipindeki değişkenliği nedeniyle zorludur. Örneğin, yoğun pus, kar veya aşırı pozlama durumunda önemli bir renk kayması görünebilir. Ayrıca veri kümesinde bulunan araç tipi olarak otomobiller, kamyonlar, sedanlar ve otobüsler bulunmaktadır.

Son olarak, hız hesaplama için yayınlanan veri kümesi kullanıldı. Veri kümeleri, saniyede 30,15 kare hızında (fram per second) 1920×1080 piksel kare çözünürlüğüne sahip, düşük maliyetli, 5 megapiksel CMOS görüntü sensörü tarafından çekilen 20 video içermektedir. Videolar hava ve kayıt koşullarına göre 5 sete bölünmüştür.

Her videonun, her aracın gerçek hızının yanı sıra, her aracın ilk plaka oluşumu için sınırlayıcı kutular içeren, basit bir XML biçiminde ilişkili bir temel doğruluk dosyası vardır. Araç plakaları için temel gerçek, manuel olarak eklenmiştir. Yer gerçeği (groundtruth) hızları, Brezilya ulusal metroloji ajansı (Inmetro) tarafından uygun şekilde kalibre edilmiş ve onaylanmış endüktif döngü dedektörüne dayalı yüksek hassasiyetli bir hız ölçerden elde edilmiştir.

Veri setinde manuel olarak ölçülen temel gerçek hız değerlerinin yanı sıra araçların motosikletler ve motosiklet olup olmadığına dair doğru (true) ya da false (yanlış) değerini tutan boolean tipinde değişen de içermektedir. Bayrak değeri olarak kullanılmaktadır.

Bunun nedeni de yazarlar, motosikletlerin hızının ölçümünün zorluk teşkil ettiğini ifade etmektedirler. Bu durum vakaların sadece %43'ünde (sıradan araçlar için %92'ye kıyasla) hızı ölçümlerinde gerçek hız ölçer için zorlu bir problem olduğu öne sürülmektedir.

Şekil 3.3’te verisetinden örnek bir görüntü ile görüntüde bulunana araçların zemin gerçeğine ait XML dosyasından örnek bir kesit verilmiştir. XML dosyasında, aracın bulunduğu frame numarası, şerit bilgisi, plaka tespit edilmediği, hız değeri, aracın en son kaçıncı framede görüldüğü gibi bilgiler bulunmaktadır.

```
−<GroundTruthRoot>
 −<gtruth>
   −<vehicle iframe="58" lane="1" moto="True" plate="True" radar="False" sema="False">
      <region h="41" w="52" x="493" y="696"/>
    </vehicle>
   −<vehicle iframe="71" lane="3" moto="False" plate="True" radar="True" sema="False">
      <region h="35" w="106" x="1632" y="930"/>
      <radar frame_end="111" frame_start="71" speed="56.65"/>
    </vehicle>
   −<vehicle iframe="107" lane="3" moto="False" plate="True" radar="True" sema="False">
      <region h="27" w="103" x="1563" y="896"/>
      <radar frame_end="147" frame_start="107" speed="53.74"/>
    </vehicle>
   −<vehicle iframe="187" lane="2" moto="False" plate="True" radar="True" sema="False">
      <region h="28" w="108" x="880" y="926"/>
      <radar frame_end="227" frame_start="187" speed="52.13"/>
    </vehicle>
   −<vehicle iframe="245" lane="3" moto="False" plate="True" radar="True" sema="False">
      <region h="32" w="103" x="1722" y="892"/>
      <radar frame_end="285" frame_start="245" speed="49.06"/>
    </vehicle>
```

Şekil 3.3. Hız veri kümesinden örnek görüntüler

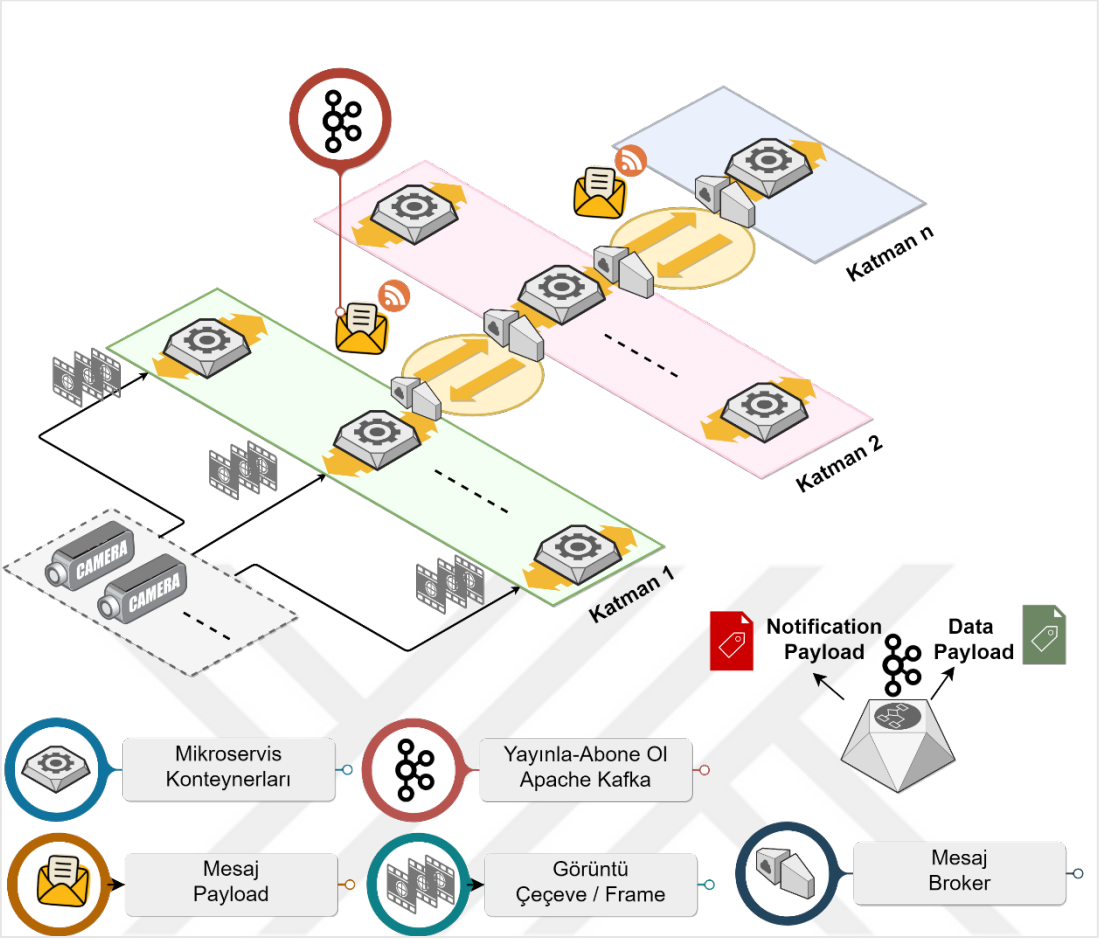
4. YÖNTEM

Önerilen mimari, CCTV görüntüleri gibi büyük ölçekli akış görüntülerini gerçek zamanlı olarak filtrelemeye uygun olarak tasarlanmıştır. Birden fazla kamera kanalından veri alabilir ve işleyebilir. Kameradan alınan verilerin mikroservislere iletimi, işbirlikçi filtreler arasındaki tüm mesajlaşma kuyruk yapılarını kullanarak, kalıcı iletişimi sağlayan yayınla-abone ol mesajlaşma paradigması olan Apache Kafka aracılığı ile yapılır.

Apache Kafka, yüksek verim ve hataya dayanıklı mekanizma avantajlarına sahip, mesaj tabanlı dağıtılmış bir yayınla-abone ol sistemidir. Bu çalışmadaki veri kaynağı türü, kameralar tarafından elde edilen video verisidir. Apache Kafka aboneleri mesajları alır ve ardından akış gruplarını işlemek üzere mikroservislere iletir. Geliştirilen mimarinin çalışma adımı Şekil 4.1'te gösterilmiştir.

Şekil 4.1. Mimari akış adımları

Görevler arasındaki iletişim yükünü en aza indirmek için çok katmanlı bir sistem mimarisi sağlayarak, dağıtılmış paralel ortamda gerçek zamanlı büyük görüntü akışını verimli bir şekilde işlenebilmesi sağlanmıştır. Şekil 4.2'de gösterildiği gibi mimari Sınıflandırıcı, İşbirlikçi-Sınıflandırıcı ve Kompleks-İşbirlikçi-Sınıflandırıcılar olmak üzere üç katmandan oluşur. Mikroservisler yapay zeka sınıflandırıcı tabanlı belirlenen modeller ile filtreleme işlemini gerçekleştirir.



Şekil 4.2. Mimari Genel Bakış

Sınıflandırıcı katmanında bulunan her bir mikroservis, belirlenen özellik kümesi içerisinden tekli kombinasyona göre seçilen özelliklere göre görseller üzerinde filtreleme işlemi gerçekleştirir. İşbirlikçi-Sınıflandırıcı katmanında bulunan mikroservisler, özellik kümesi içerisinden ikili kombinasyona göre seçilen özelliklere göre görseller üzerinde filtreleme işlemi gerçekleştirir. Bunu sınıflandırıcı katmanında bulunan mikroservislerden elde ettiği sonuçlara göre gerçekleştirir. Kompleks-İşbirlikçi-Sınıflandırıcı katmanı ise tüm filtrelerin kombinasyonu ile çalışmaktadır. Bu bölümde sırasıyla:

- Mikroservislerde görev yapacak yapay zeka filtrelerinin tasarımı
- Mikroservis mimarisi tasarımı Monolitik-mikroservis dönüşümü
- Mikroservis mimarisinde kullanılan önemli bir bileşen olan Docker, konteyner ve Mikroservislerin haberleşme kanalı Apache Kafka çerçevesi

- Kanal üzerinden gönderilecek paket yapıları, seri hale getirme ve seri durumdan çıkarma
- Paket güvenliği ve şifreleme
- Sistem izleme

Maddeleri ayrı başlıklar altında açıklanmıştır.

4.1. Yapay Zeka Filtreleri

Sınıflandırma aşamasında üç farklı yapay zeka modeli kullanılmıştır bunlar; aracın tipi, aracın rengi ve arasın hızıdır. Böylece her bir filtrede görev yapacak modül belirlenir. Bu bölüm ağırlıklı olarak derin öğrenme tabanlı nesne sınıflandırma prensibine dayanmaktadır.

Mimaride senaryo olarak araçların tip, renk ve hız özelliklerine göre sınıflandırılması ele alınmıştır.

4.1.1. Araç tip tespit etme filtresinin tasarlanması

Nesne algılama modelleri hem performans hem de hız açısından artarak daha iyi olmaya devam etmektedir. Gerçek zamanlı nesne algılama alanında, YOLOv3 (2018) ve Google Brain ekibi tarafından EfficientDet (2020) popüler bir seçim oldu. COCO'daki standart metriklere göre yeni nesne algılama şampiyonu olduğu gösterilen YOLOv4'ün son sürümü (2020) ile ilerleme halen devam etmektedir.

YOLO (You Only Look Once), Joseph Redmon tarafından Darknet adlı özel bir çerçevede yazılmıştır. Darknet, düşük seviyeli dillerde yazılmış çok esnek bir araştırma çerçevesidir ve bilgisayar vizyonunda bir dizi en iyi gerçek zamanlı nesne dedektörü üretmiştir: YOLO, YOLOv2, YOLOv3 ve YOLOv4.

Bu genel nesne algılama modelleri, çok çeşitli nesneleri ve sınıfları içeren COCO veri kümesinde kanıtlanmıştır ve bu görevde iyi performans gösterebilirlerse yeni veri kümelerine iyi genelleşecekleri düşüncesiyle kanıtlanmıştır. Ancak, araştırmalarda kullanılan derin öğrenme tekniklerinin özel nesneler üzerinde pratikte uygulanması zor olabildiğini göstermektedir.

YOLO, sınırlayıcı kutular çizme ve sınıf etiketlerini tanımlama problemini tek bir uçtan uca türevlenebilir ağda birleştiren ilk nesne algılama ağıydı. YOLO versiyonun 2. si olan YOLOv2, daha yüksek çözünürlük ve bağlantı kutuları dahil olmak üzere YOLO'nun üzerinde bir dizi yinelemeli iyileştirme yapılarak elde edilmiştir. 3. versiyon olan YOLOv3, sınırlayıcı kutu tahmininde omurga ağ katmanlarına bağlantılar ekleyerek ve daha küçük nesnelerdeki performansı arttırılmıştır. YoloV4, YoloV3'ün önemli bir iyileştirmesidir, Omurgada yeni bir mimarinin uygulanması ve Neck değişiklikleri, mAP'yi (Ortalama Hassasiyet) %10 ve FPS (Saniyedeki Kare) sayısını %12 oranında iyileştirmiştir. Ayrıca bu sinir ağını tek bir GPU üzerinde eğitmek daha kolay hale gelmiştir.

Modern bir dedektör genellikle iki bölümden oluşur; özellikleri ayıklayan ImageNet veri kümesinde önceden eğitilmiş bir omurga ve sınıfları ve nesne sınırlayıcı kutularını tahmin eden bir kafa. Kafa genellikle iki türe ayrılır, yani tek aşamalı nesne dedektörü ve iki aşamalı nesne dedektörü. İki aşamalı nesne detektörü önce ilgi bölgeleri oluşturmak için bir Region Proposal Network kullanır, ardından sınırlayıcı kutuları sınıflandırıp oluştururken, tek aşamalı nesne algılayıcı bir giriş görüntüsü alarak ve sınıf olasılıklarını ve sınırlayıcı kutu koordinatlarını öğrenerek bunu bir regresyon problemi gibi ele alır. .Tek aşamalı nesne dedektörü için en temsili model YOLO iken, en temsili iki aşamalı nesne dedektörü R-ConNN serisidir. Son zamanlarda, çeşitli aşağıdan yukarıya ve yukarıdan aşağıya yollardan oluşan farklı aşamalardan özellik haritaları toplamak için Neck adı verilen omurga ve kafa arasına bir katman daha eklenmiştir.

YOLOv4 omurgası incelenirse, omurganın temel amacı, temel özellikleri çıkarmaktır, omurganın seçimi, nesne algılama performansını artıracak önemli bir adımdır. Omurgayı eğitmek için genellikle önceden eğitilmiş sinir ağları kullanılır. YOLOv4 ağı, omurga ağı için CSPDarknet53'ü uygulamıştır.

Tez çalışması kapsamında Problem 2'de belirtildiği üzere, YOLOv4 ve tiny-YOLOv4 tip sınıflandırması için iki farklı mimari kullanılmıştır. YOLO, sınıflandırma sistemlerini özellik çıkarma adımlarından kurtaran özellik çıkarma adımını kendi başına yapar. Ayrıca sistemimizi kamera hareketlerine veya ortamdaki değişikliklere karşı dayanıklı hale getirir. YOLOv4 kullanmanın en önemli avantajı mükemmel hızıdır; saniyede 45 kare

işleyebilir. Derin öğrenme yöntemlerinden biri olan ConNN ailesi, nesneleri görüntü içinde konumlandırmak için öncelikle bölgeleri kullanır. Ağ, görüntünün tamamını gözlemlemez; yalnızca, aranan bir nesneyi içerme olasılığı daha yüksek olabilecek kısımlarına bakar. Öte yandan, YOLOv4 çerçevesi tüm görüntüyü tek bir örnekte alır. YOLOV4, sınırlayıcı kutu koordinatlarını ve sınıf olasılıklarını tahmin eder. Omurga, özellik çıkarma işlemi olarak da bilinen YOLOv4'ün ilk katmanıdır. İki farklı omurga mimarisi kullanıldı:

- Birincisi, 9 evrişim katmanından oluşan Tiny-YOLOv4.
- İkincisi Darknet53, 53 evrişim katmanından oluşan YOLOv4.

Geliştirilen mimaride araç tipleri sınıflandırmasında BIT-vehicle veri kümesi kullanılmıştır. Bu veri kümesinde araç görselleri altı kategoride sunulmaktadır: otobüsler, minibüsler, kamyonlar, kamyonetler, sedanlar ve SUV'ler. Modeli eğitmek için gerekli veri kümesinin elde edilmesinden sonraki sıra verilerin etiketlenmesidir. Görüntüleri etiketlemek için, tespit edilmesi istenilen nesnelerin çevresine sınırlayıcı kutular çizilir. Görüntü etiketleme sırasında:

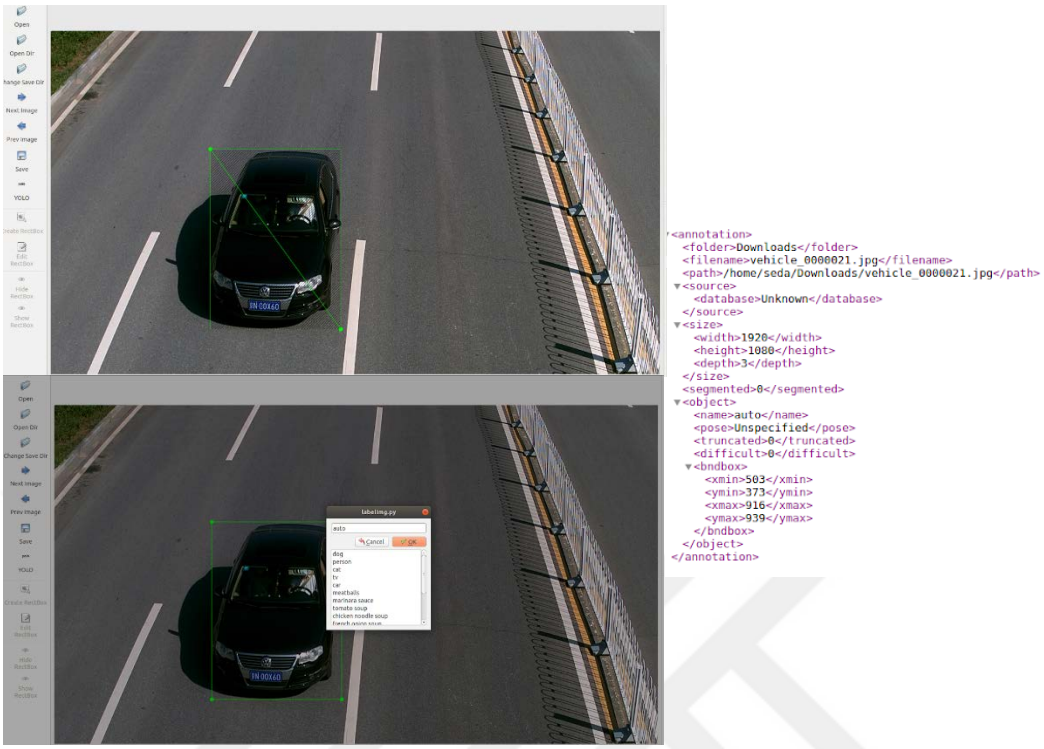
- Her görüntüdeki her ilgili nesne etiketlenir.
- Bir nesnenin tamamının etiketlenmesi önemlidir.
- Sıkı sınırlayıcı kutular oluşturulur. Eksik parça bırakılmamalı ya da fazladan alan eklenmemeli.
- Belirli etiket adları oluşturulur.
- Açık etiketleme talimatları korunur

Etiketleme için CVAT, LabelImg, RectLabel ve Roboflow gibi etiketleme araçları kullanılır. Toplamda 9.850 araç görüntüsü manuel olarak etiketlenmiştir. Şekil 4.3’te nesnenin LabelImg aracı ile sınırlayıcı kutuya alarak etiketlenmesi adımları ve nesneye ait etiket verileri verilmiştir.

Özel bir veri kümesini YOLOv4 için eğitim yapılandırmasını konfigüre adımları şunlardır:

- Input size 288x288
- Learning Rate 0.001

- Batch boyutu 64'e ayarlandı- Batch boyutu, yineleme başına görüntü sayısıdır



Şekil 4.3. Mimari Genel Bakış

- Subdivision değerleri 12'ye ayarlandı- subdivisions, toplu işlerin GPU belleği için bölündüğü parça sayısıdır.
- max_batch değeri hesaplandı- 2000 * sınıf sayısı (6 etiket) = 12000
- steps değerleri max_batches değerinin %80 ve %90'ın hesaplanması ile elde edildi = 9600, 10800
- tüm YOLO katmanlarında num_classes değerini yeni etiket değeri ile değiştirildi
- filters değerini her bir yolo katmanından önceki 3 konvolüsyon öğesinde yer alan değeri, filter=(classes+5)x3 formülü kullanılarak değiştirildi. Böylece filter = 33 elde edildi.

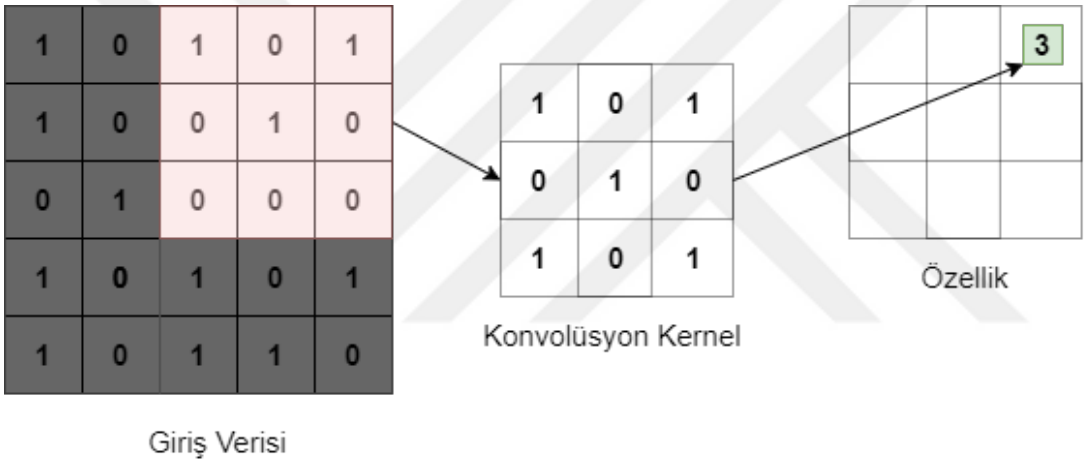
Eğitim doğrulama setinde hesaplanarak ve her 1000 yinelemede bir yazdıracaktır.

4.1.2. Araç renk tespit etme filtresinin tasarlanması

ConNN'ler, çok katmanlı algılayıcıların düzenli versiyonlarıdır. Çok katmanlı algılayıcılar genellikle tamamen birbirine bağlı ağlar anlamına gelir. Tamamen bağlantılı ağlar, bir katmandaki her nöron bir sonraki katmandaki tüm nöronlara bağlıdır. Bu yapı,

onları fazla veriye yatkın hale getirir. Normalleştirmenin veya aşırı öğrenmeyi önlemenin tipik yolları şunları içerir: eğitim sırasında parametreleri cezalandırma (bağlantı ağırlığın azalması gibi) veya bağlantının kesilmesi (bırakma vb.) kullanarak artan karmaşıklık azaltılabilir.

Bir RGB görüntüsü, üç düzleme sahip bir piksel değerleri matrisindir, gri tonlamalı bir görüntü aynıdır ancak tek bir düzlemi vardır. Basit olması için, ConNN'lerin nasıl çalıştığını anlamaya çalışırken gri tonlamalı görüntülere bağlı kalalım. Şekil 4.4 bir konvolüsyonun ne olduğunu göstermektedir. Bir filtre/çekirdek (3×3 matris) olduğunu varsayalım ve özelliği elde etmek için giriş görüntüsüne uygularız. Bu özellik bir sonraki katmana aktarılır.



Şekil 4.4. ConNN Çekirdek Yapısı

Evrişimli sinir ağları, çoklu yapay nöron katmanlarından oluşur. Biyolojik muadillerinin kaba bir taklidi olan yapay nöronlar, çoklu girdilerin ağırlıklı toplamını hesaplayan ve bir aktivasyon değeri veren matematiksel fonksiyonlardır.

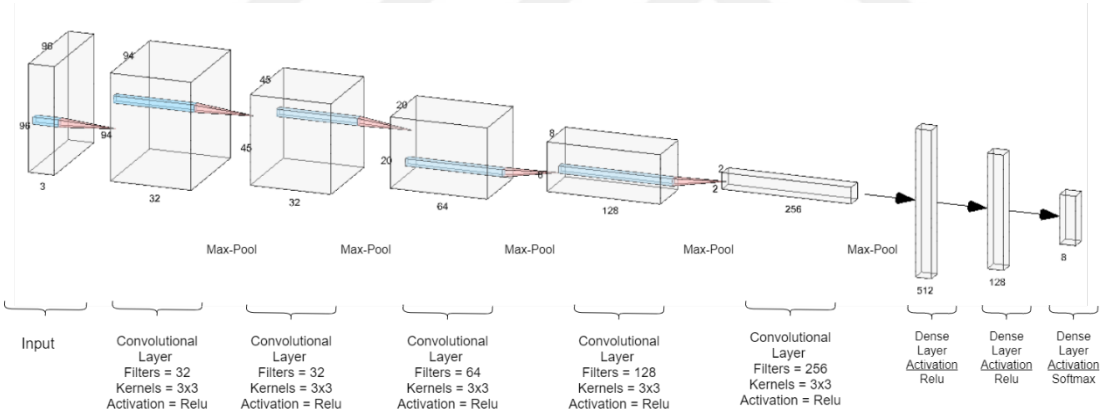
İlk katman genellikle yatay veya çapraz kenarlar gibi temel özellikleri çıkarır. Bu çıktı, köşeler veya birleşik kenarlar gibi daha karmaşık özellikleri algılayan bir sonraki katmana iletilir. Ağın derinliklerine indikçe nesneler, yüzler vb. gibi daha karmaşık özellikleri tanımlayabilir.

Sınıflandırma katmanı, görüntünün bir "sınıfa" ait olma olasılığını belirten bir dizi güven puanı (0 ile 1 arasındaki değerler) üretir. Evrişim Katmanına benzer şekilde, Havuzlama katmanı, Evrişimli Özelliğin uzamsal boyutunu küçültmekten sorumludur. Bu, boyutları

küçülterek verileri işlemek için gereken hesaplama gücünü azaltmaktır. Ortalama havuzlama ve maksimum havuzlama çok sık kullanılır.

Maksimum havuzlama, çekirdeğin kapsadığı görüntünün bir bölümünden bir pikselin maksimum değerini bulur. Maksimum havuzlama ayrıca Gürültü engelleyi olarak da çalışır. Öte yandan, Ortalama Havuzlama, görüntünün Çekirdeğin kapsadığı kısmından tüm değerlerin ortalamasını döndürür. Ortalama Havuzlama, bir gürültü bastırma mekanizması olarak basitçe boyutsallık azaltma gerçekleştirir.

Problem 3'de belirtildiği üzere çözüm için ConNN mimarisi Alexnet mimarisinden esinlerek gerçeklenmiştir. Siyah, mavi, camgöbeği, gri, yeşil, kırmızı, sarı ve beyaz olmak üzere 8 farklı renk sınıfı etikeini sınfılandırılmasında kullanılmıştır. Model mimarisi Şekil 4.5'te ve Tablo 4.1'de verilmiştir. ConNN modelimiz 5 evrişimli, 5 maksimum havuzlama, 1 düzleştirme, 6 bırakma ve 3 yoğun katmandan oluşmaktadır. Kayıp fonksiyonu ve Adam optimizer için ortalama kare hatası kullanıldı.



Şekil 4.5. Araç renk tespitinde kullanılan ConNN mimarisi

Tablo 4.1. ConNN Model mimarisi

Katman Türü	**Katman Parametreleri**
Conv2D	Filtre = 32, Çekirdek Boyutu = (3, 3), Aktivasyon = relu
MaxPooling2D	Havuz Boyutu = (2, 2)
Dropout	0.25
Conv2D	Filtre = 32, Çekirdek Boyutu = (3, 3), Aktivasyon = relu
MaxPooling2D	Havuz Boyutu = (2, 2)

Tablo 4.1. (Devam) ConNN Model mimarisi

Dropout	**0.25**
Conv2D	Filtre = 64, Çekirdek Boyutu = (3, 3), Aktivasyon = relu
MaxPooling2D	Havuz Boyutu = (2, 2)
Dropout	0.25
Conv2D	Filtre = 128, Çekirdek Boyutu = (3, 3), Aktivasyon = relu
MaxPooling2D	Havuz Boyutu = (2, 2)
Dropout	0.25
Conv2D	Filtre = 256, Çekirdek Boyutu = (3, 3), Aktivasyon = relu
MaxPooling2D	Havuz Boyutu = (2, 2)
Dropout	0.25
Dense	Birim= 512, Aktivasyon = relu
Dropout	0.25
Dense	Birim = 128, Aktivasyon = relu
Dropout	0.25
Dense	Birim = 8, Aktivasyon = softmax

Denklem (4.1)'de gösterildiği gibi karesel hata kayıp fonksiyonu kullanılmıştır. ŷ tahmin edilen değer, y ana hedeftir ve N örnek numarasıdır.

$$L(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} (y - \hat{y}_i)^2 \quad (4.1)$$

4.1.3. Araç hız tespit etme filtresinin tasarlanması

Tek bir kameradan araç hızı tespiti, açı, yükseklik vb. kamera parametrelerinin bilinemeyebileceği durumlar ve bunların belirlenmesinin zorlaştığı durumlar için zorlu bir problemdir.

Önerilen sistem, hareket içeren görüntü bölgelerinde araçları verimli bir şekilde bulmak için optimize edilmiş bir hareket dedektörü kullanır. Ayırt edici özellik araç seçilir, birden fazla çerçeve boyunca izlenir ve perspektif bozulması için düzeltilir. Araç hızı, izlenen özelliklerin yörüngeleri bilinen gerçek ölçümleriyle karşılaştırılarak ölçülür. Önerilen

sistem, tek bir düşük maliyetli kamera tarafından farklı hava koşullarında kaydedilen yaklaşık 5 saatlik videoları içeren bir veri kümesi üzerinde test edilmiştir.

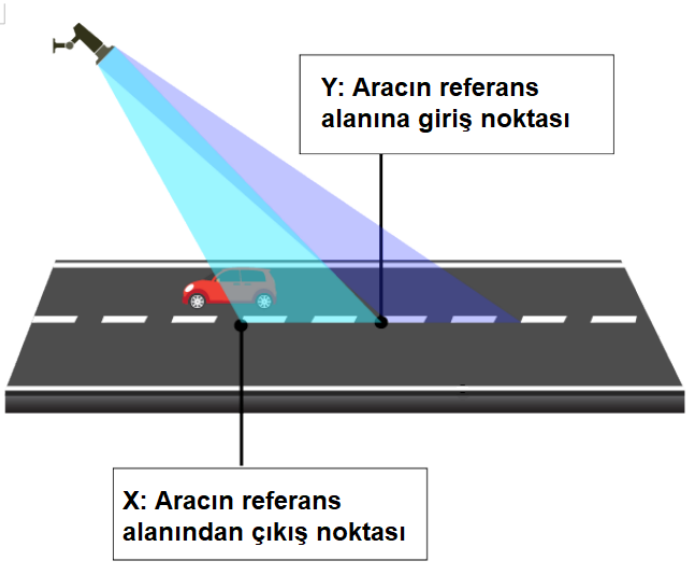
Hesaplamalar, mesafesi bilinen bir yol üzerinde yapıldı. Araçların hızlarının tespit edilebilmesi için öncelikle araç tespit ve takip işlemlerinin gerçekleştirilmesi gerekmektedir. Araç takip işlemi aracın kat ettiği mesafeyi elde ediyor ve ayrıca bu değerin zamanla oranının bize aracın hızını verdiğini biliyoruz. Araçları tespit etmek için YOLOv4 algoritmasını kullandık. Araç tespiti sırasında arka plan çıkarma yöntemi kullanılmadığından araç kuyrukları, yavaş trafik koşulları ve yolda uzun bekleme süreleri işimiz için herhangi bir sorun teşkil etmemektedir.

Videodaki hareketli nesneleri takip etmek için literatürde önerilen birçok algoritmaya izleme algoritmaları denir. Çalışmamızda açık kaynaklı bir görüntü işleme kütüphanesi olan OpenCV (Open Source Computer Vision) kullanılmıştır. OpenCV'de Boosting, Goturn, KCF, MedianFlow ve TLD gibi izleme algoritmaları için uygulamalar bulunmaktadır. Projede MedianFlow algoritması kullanılmıştır. MediaFlow algoritması, izlenen nesneye bir sınırlayıcı kutu gibi davranır ve nesneyi yeni bir çerçevede izlemek için yeni bir sınırlayıcı kutu açar.

Sistemimiz, problem 4'e çözüm için nesne izleme yöntemine göre araç hızlarını hesaplar. Veri seti ile sistemimizin varsayımı, çizgiler arasındaki bilinen mesafe ve fps değerleridir. Şekil 4.6'da gsterildiği üzer araç takibi bilinen bir ortamda ve referans aralığında gerçekleştirilir.

Bu referans noktaları 4,8 metre genişliğinde ve 2,0 metre uzunluğundadır. Aracı çevreleyen sınırlayıcı kutu ile aracın konumu izlenir. Sınırlama kutusu referans alanına girdiği andan itibaren süre tutulur ve araç alandan ayrıldığında ortalama hız hesaplanır.

Son olarak, araç hızları, araçları çevreleyen sınırlayıcı kutu yörüngeleri gerçek ölçümleriyle karşılaştırılarak ölçülür. Bu referans noktaları 4,8 metre genişliğinde ve 2,0 metre uzunluğundadır. Aracı çevreleyen sınırlayıcı kutu ile aracın konumu izlenir. Sınırlama kutusu referans alanına girdiği andan itibaren süre tutulur ve araç alandan ayrıldığında ortalama hız hesaplanır. Hız ölçümü $v = (X\text{-}Y)/süre$ formülü ile hesaplanır. Hız kategorileri 10'ar km/sa olarak tanımlandı.



Şekil 4.6. Araç hız tespitinde kullanılan mimari

4.2. Video Sorgulama Sisteminde Mikroservis Mimarisinin Tasarlanması

Monolitik mimari, tüm işlevlerin tek bir uygulamada kapsüllendiği, yazılım geliştirmenin geleneksel yoludur. Bu tür mimari sıkı bir şekilde birleştirilmiştir ve bu, bileşenlerden biri mevcut değilse, sistemin yürütülemeyeceği veya derlenemeyeceği anlamına gelir. Monolit mimarinin kendine göre avantajları ve dezavantajları vardır. Monolitik Mimarinin faydaları şu şekilde sıralanabilir:

- Her birleşen aynı uygulama üzerinden yürütülürken bunları bağlamak daha kolaydır
- Daha az operasyonel yük; sadece bir uygulamanın kurulması gerekir.

Monolitik mimarinin dezavantajları şunlardır:

- Büyük boyutlu monolitlerin karmaşıklığı nedeniyle bakımı ve geliştirilmesi zordur. Hataların izini sürmek, kod tabanları üzerinden uzun incelemeler gerektirir.
- Monolitler kütüphane ekleme veya güncelleme sonunda derlemeyen/çalıştırmayan veya hatalı davranan tutarsız sistemlerle sonuçlandığı, “bağımlılık cehenneminden” muzdariptir.
- Bir modüldeki herhangi bir değişiklik, tüm uygulamanın yeniden başlatılmasını gerektirir.
- Monolitik uygulamaların dağıtımı, kurucu modellerin kaynaklarındaki çelişkili gereksinimler nedeniyle genellikle problemlere sebep vermektedir.

- Ölçeklenebilirliği sınırlıdır.
- Orijinal uygulamanın aynı dilini ve çerçevelerini kullanmak zorunda olan geliştiriciler için bir teknoloji kilitlenmesini temsil eder.
- Güvenilirlik; herhangi bir bileşendeki hata, tüm uygulamayı potansiyel olarak çökertebilir. Bu sebep ve problemler, yeni çözümleri de beraberinde getirmiştir.

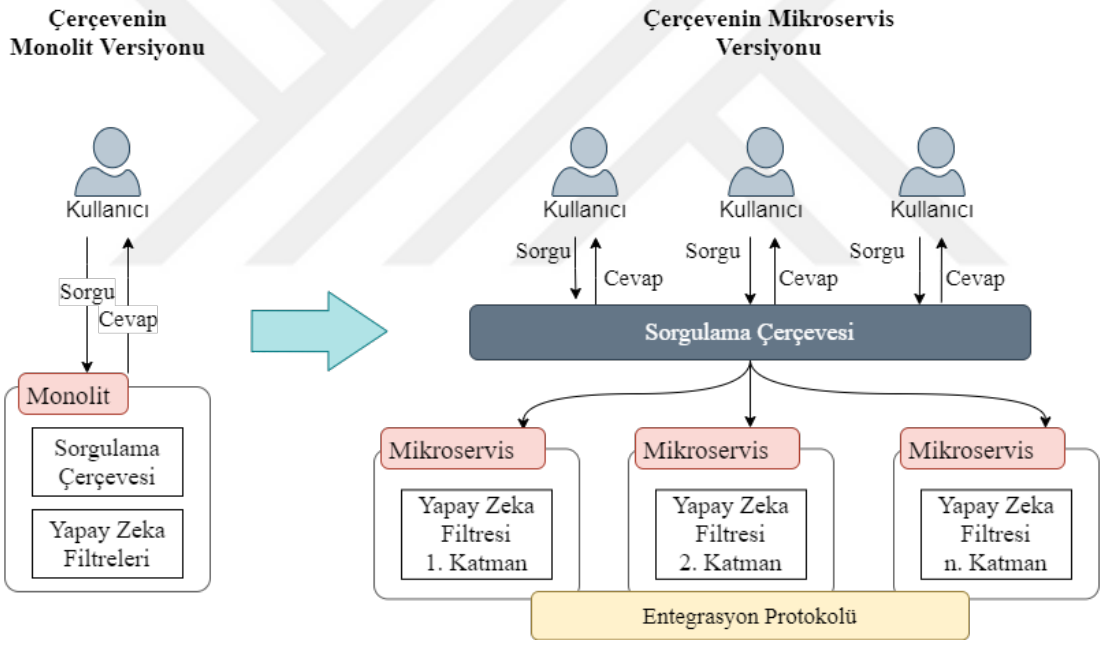
Martin Fowler'a göre, bir mikroservis mimarisi, otomatik dağıtım, dillerin ve verilerin merkezi olmayan denetimi etrafında organize edilmiş bağımsız olarak dağıtılabilir hizmet paketlerinden oluşur (Atchison, 2021). Mikroservis mimarisi, monoliti bir dizi küçük hizmete ayırarak REST API veya mesaj veri yolu gibi hafif mekanizmalar aracılığıyla birbirleriyle iletişim kurmalarını sağlayarak monolitik mimarinin zorluklarını aşmanın alternatif bir yoludur. Mimarinin bileşenleri olarak tasarlanan ve geliştirilen mikroservisler, işletim sistemlerinden, veritabanlarından ve dillerden bağımsız esnek dağıtımların yanı sıra daha kısa geliştirme sürelerine olanak tanır. Mikroservisler, bir uygulamayı bağımsız bireysel hizmetler topluluğu olarak oluşturan bir yazılım mimarisi türüdür. Her hizmet, tek bir iş yeteneğine adanmıştır. Hizmetler birçok programlama dilinde uygulanabilir ve otomatik olarak test edilip dağıtılabilir. Mikroservis modeli, hizmetlerin her birinin bir hizmeti diğerinden ayırmak için tasarlanmış kendi bağımsız süreç bağlamında çalıştığı bir hizmetler paketi olarak dağıtılmış uygulamaları düzenlemeyi savunur. Dahili hizmetler arası iletişim, tescilli arabirimler aracılığıyla uygulanır, bunların uygulamaları ağırlıklı olarak TCP/UDP tabanlı uygulama protokollerine dayanır. Servisler çok dilli programlamayı destekler. Örneğin, hizmetlerin aynı teknoloji yığınını, kütüphaneleri veya çerçeveleri paylaşması gerekmez.

Mikroservisler belirtilen monolitik uygulamalar sorunlarıyla aşağıda sıralanan maddeler şeklinde başa çıkmaktadır:

- Mikroservisler, kod tabanlarını küçülten ve doğal olarak bir hatanın kapsamını sınırlayan sınırlı sayıda işlevsellik uygular.
- Yeni sürümlere kademeli geçişler planlamak mümkündür.
- Bir modülü değiştirmek için tüm sistemin tamamen yeniden başlatılması gerekmez. Yeniden başlatma, yalnızca o modülün mikroservisleri ile ilgilidir.

- Mikroservisler, doğal olarak konteynerleştirmeye uygundur ve geliştiriciler, gereksinimlerine en uygun dağıtım ortamının yapılandırılmasında yüksek derecede özgürlüğe sahiptir;
- Birlikte çalışan mikroservisler mimarisinde uygulanan tek kısıtlama, bunların iletişim kurmasını sağlamak için kullanılan teknolojidir (protokoller, veri kodlamaları vb.). Bunun dışında, mikro hizmetler ek bir kilitleme gerektirmez ve geliştiriciler her bir mikro hizmetin uygulanması için en uygun kaynakları (diller, çerçeveler, vb.) özgürce seçebilirler.

Problem 5'te açıklandığı üzere, Şekil 4.7'de önerilen mimarinin monolit ve mikroservis versiyonları karşılaştırılmıştır ve Mikroservis mimarisinin nasıl olması gerektiği gösterilmiştir.



Şekil 4.7. Monolitik ve mikroservis mimarileri

Mikroservis mimarisinin tasarımında takip edilmesi gereken adımlar:

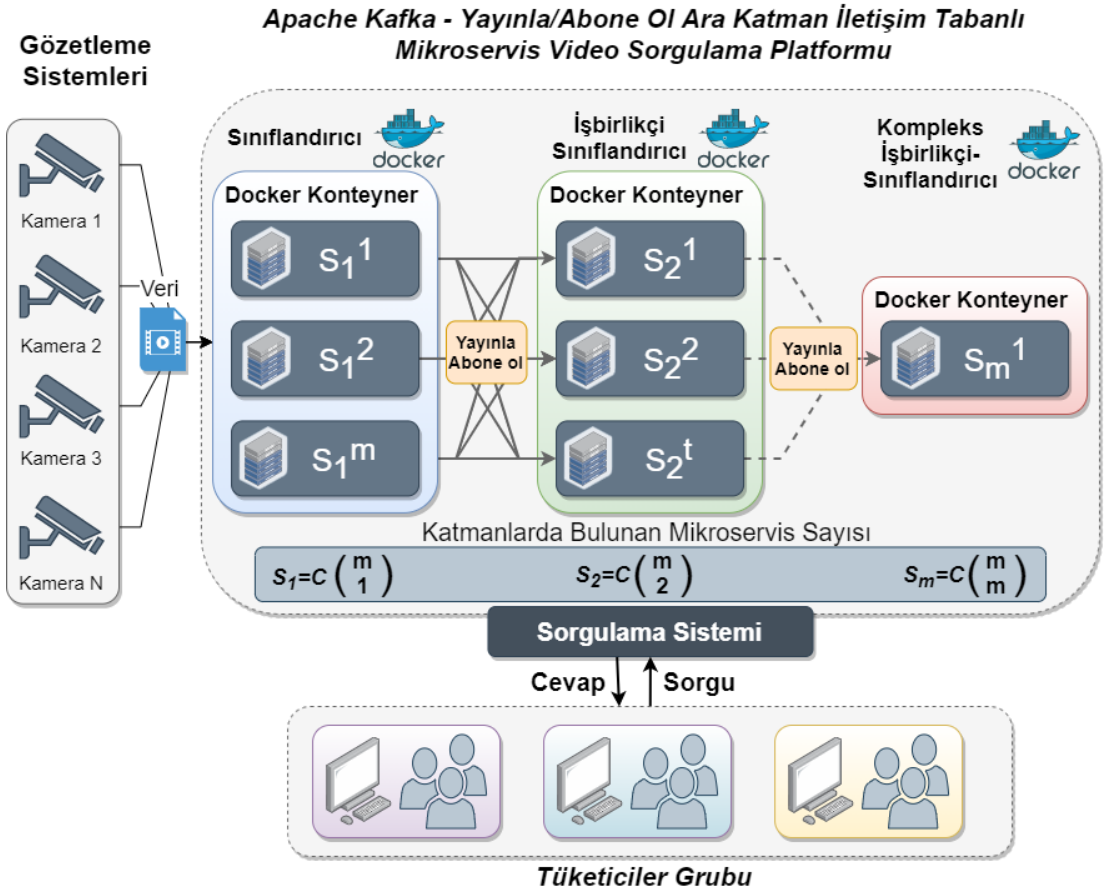
- API Ağ Geçidi programlama
- Hizmetlerin birbiri ile iletişimini tanımlama
- Hizmetlerin birbirini keşfi için protokol oluşturma
- Servis hata toleransını belirleme
- Veri işleme

Bu adımlar birer problem şeklinde ele alınarak çözülmektedir. Şekil 4.7'de mikroservis mimarisinde gösterilen Sorgulama Çerçevesi, önerilen mimaride API Ağ Geçididir. API ağ geçidi, istemciler için sistemi kullanma noktasıdır. İstemciler, hizmetleri doğrudan çağırmak yerine, aramayı arka uçtaki uygun hizmetlere ileten API ağ geçidini çağırır. API ağ geçidi kullanmanın avantajları şunlardır:

- İstemcileri hizmetlerden ayırır.
- API Ağ Geçidi, kimlik doğrulama, günlük kaydı (loglama), SSL sonlandırma ve yük dengeleme gibi diğer kesişen işlevleri gerçekleştirebilir.

Sistem ağ geçidi hakkında detaylı açıklama, sistem izleme bölümünde verilmiştir.

Önerilen sistemi, monolit uygulamadan mikroservis mimarisine dönüştürmede kurallar tanımlandı. Şekil 4.8'da gösterildiği üzere, önerilen mimari: Sınıflandırıcı, İşbirlikçi Sınıflandırıcı ve Kompleks-İşbirlikçi Sınıflandırıcı olmak üzere üç katmana ayrılmıştır.



Şekil 4.8. Önerilen Mikroservis Mimari Şeması

Katmanlar servislerin görev yüklerine göre belirlenmektedir. Sınıflandırıcı katmanında, özellik kümesi içerisinde bulunan her bir özellik için şekilde yapay zeka tabanlı sınıflandırma işlemi gerçekleştirilir. Diğer katmanlar bu servislerin kombinasyonları şeklinde, çoklu özelliklere göre filtreleme yapmaktadırlar. Katman numarası servislerin kaçlı kombinasyonlarına göre sınıflandırma işlemi yapacağını belirtir.

Önerilen sistemde *N* adet kamera ve *m* adet özellik bulunur. *E,* özellik kümesi olmak üzere m adet elemanı bulunur. Bu tez kapsamında örnek olarak kullanılan özellikler araç tipi, rengi ve hızıdır. Özelliğe *a* denirse,

- a_1 = araç tipini, a_2 = araç rengini, a_3 = araç hızını temsil eder.

Böylece özellik sayısını temsil eden *m, m = 3* olacaktır. Özellik kümesi *E, E = {tip, renk, hız}* kümesi olacaktır, böylelikle:

- $E = \{a_1, a_2, \ldots, a_m\}$ şeklinde formüle edilir ve $s(E) = m$ diyebiliriz.

Her bir katman kombinasyon sayısını belirler, *S* katmanı ifade eden semboldür.

- S_1: 1. *Katman* (Sınıflandırıcı). Küme gösterimi,
- S_1 = { m | m>=1, *C(m,1)* }, S1 = { tip, renk, hız}
- S_2: 2. Katman (İşbirlikçi Sınıflandırıcı). Küme gösterimi,
- S_2 = { m | m>=2 ise *C(m,2)* }, S_2= { tip_renk, tip_hız, renk_hız}
- S_m: m. Katman (Kompleks-İşbirlikçi Sınıflandırıcı). Küme gösterimi,
- S_m = { m | m>=1, *C(m,m)* }, *C(m,m)* = 1 olduğundan dolayı son katman her zaman bir elemanlı olacaktır S_3= { tip_renk_hız}

S kümesine ait her bir küme elemanı bir mikroservise karşılık gelmektedir. Notasyonda küme elemanın yanında bulunan alt simge katman sayısını, üst simge mikroservis sırasını belirtmektedir. Katmanlarda bulunan mikroservislerin sırası ile yazımı:

- 1. Katman: S11 = { tip }, S12 = { renk }, S13 = { hız }
- 2. Katman: S21 = { tip_renk}, S12 = { tip_hız}, S13 = { renk_hız }
- m. Katman: S_m^1 = { tip_renk_hız }

Böylelikle,

- Sınıflandırmada kullanılan toplam mikroservis sayısı, 2^m - 1 olacaktır.

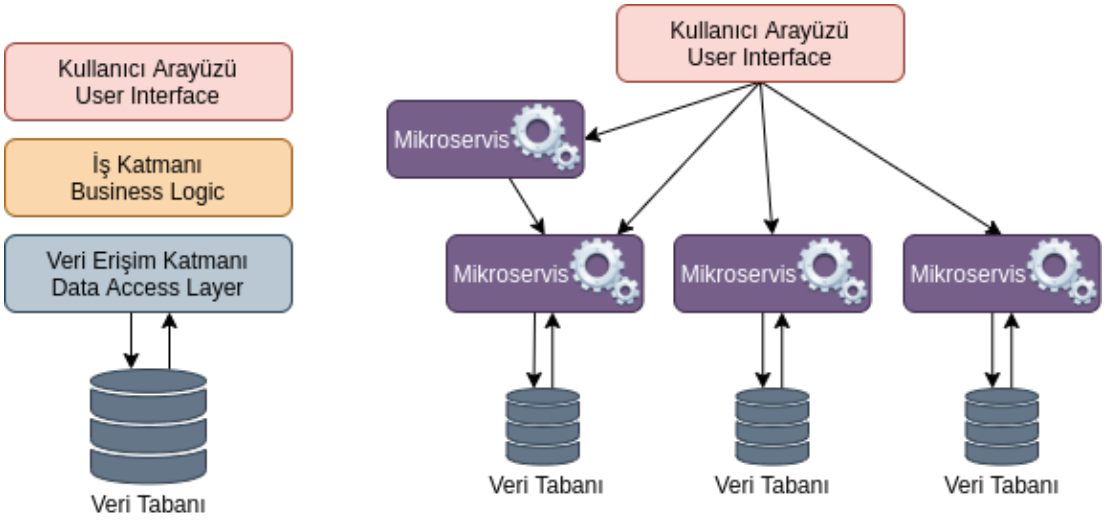
Bu kurallar bütünü ile her yeni gelen servis, görevine göre ilgili katmana yerleştirilecektir. Mikroservislerin nasıl organize şekilde çalıştığı, birbirleri ile haberleştiği bu bölümün alt başlıklarında açıklanmıştır.

4.2.1 Mikroservislerin iletişim protokolü: yayınla-abone ol paradigması

Tek bir işlem üzerinde çalışan monolitik bir uygulamada, program bileşenleri birbirlerini dil düzeyinde fonksiyon veya işlev çağrıları kullanarak çağırırlar. Eğer bu işlem kod içerisinde nesne oluşturma veya referans gösterme ile gerçekleştiriliyor ise bu durum kod içerisinde güçlü bağımlılıklara yol açmaktadır. Monolitik bir uygulamadan mikroservis tabanlı bir uygulamaya geçerken en zorlu problem, iletişim mekanizmasının değişmesi ve yeni haberleşme mimarisinin belirlenmesidir. Uzak Yordam Çağrısı (RPC), bir programın ağdaki başka bir bilgisayarda bulunan bir programdan ağın ayrıntılarını anlamadan hizmet istemek için kullanılabilecek bir protokoldür. Bir prosedür çağrısı bazen bir işlev çağrısı veya bir alt program çağrısı olarak da bilinir. RPC, istemci-sunucu modelini kullanır. Süreç içi yöntem çağrılarından, RPC çağrılarına doğrudan çağrı ile gerçekleştirilen dönüşüm, dağıtılmış ortamlarda iyi performans göstermeyecek ve verimli olmayan bir iletişime neden olacaktır.

Mikroservis tabanlı bir uygulama, genellikle birden çok sunucu veya ana bilgisayar arasında dahi birden çok işlem veya hizmet üzerinde çalışan dağıtılmış bir sistemdir. Her hizmet örneği tipik olarak bir süreçtir. Bu nedenle hizmetler, her hizmetin doğasına bağlı olarak süreçler arası iletişim protokolü kullanarak etkileşime girmelidir. Şekil 4.9'da monolit ve mikroservis uygulamalarının mimarileri gösterilmiştir.

İstemci ve hizmetler, her biri farklı bir senaryoyu ve hedefleri hedefleyen birçok farklı iletişim türü aracılığıyla iletişim kurabilir. Genel olarak, bu iletişim sistemlerini sınıflandırmak için iki kriter vardır: Protokol tipine göre, senkron ve asenkron. Alıcı sayısına göre, bir veya daha fazla.

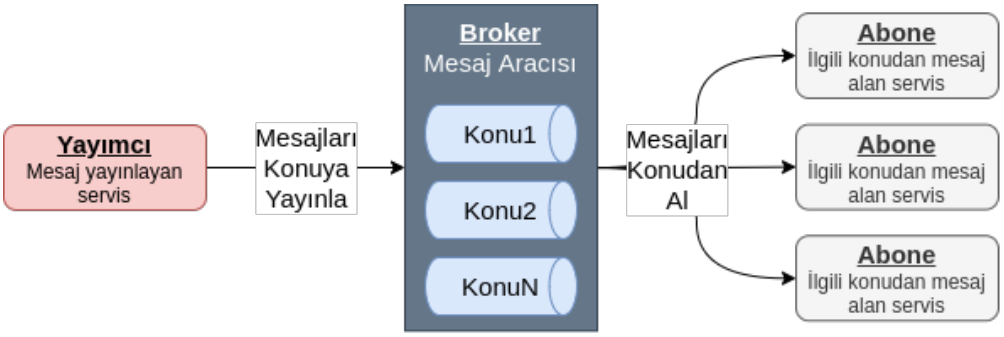


Şekil 4.9. Monolit ve Mikroservis mimari

- Senkron iletişim: Bu modelde bir hizmet, RPC gibi bir protokol kullanarak başka bir hizmetin sunduğu bir API'yi çağırır. Hizmet çağrısında bulunan göndericinin, alıcıdan cevap gelmesini beklediği bu seçenek, senkronize bir mesajlaşma modelidir.
- Asenkron iletişim: Bu modelde, bir hizmet yanıt beklemeden ileti gönderir ve bir veya daha fazla hizmet iletiyi eşzamansız olarak işler.
- Tek alıcı: Her istek tam olarak bir alıcı veya hizmet tarafından işlenmelidir.
- Çoklu alıcı: Her istek birden çok alıcı veya hizmet tarafından işlenmelidir. Bu tür bir iletişim asenkron olmalıdır. Olay odaklı mimari gibi kalıplarda kullanılan yayınlama/abone olma mekanizması buna bir örnektir. Bu, olaylar aracılığıyla birden çok mikro hizmet arasında veri güncellemeleri yayılırken bir olay veriyolu arabirimine veya mesaj aracısına (broker) dayanır.

Önerilen mimaride bir servis birden fazla servisle iletişime geçmektedir. Bu durumda, Şekil 4.10'da gösterilen çoklu alıcısı olan asenkron tabanlı iletişim yapısı olan yayınla/abone ol (pub/sub) paradigması kullanılmıştır.

Yayınla/abone ol (pub/sub) paradigması şu şekilde tanımlanabilir: aboneler (bilgi tüketicileri) abonelikler yoluyla ilgilerini ifade eder ve bunları olay tabanlı sisteme kaydeder; yayıncılar (bilgi üreticileri) olayları sisteme yayınladıklarında; olayların ilgili abonelere iletilmesinden sistem sorumludur. Aboneler ve yayıncılar hiçbir şekilde birbirlerinden haberdar değildirler.



Şekil 4.10. Yayınla/Abone ol akış diyagramı

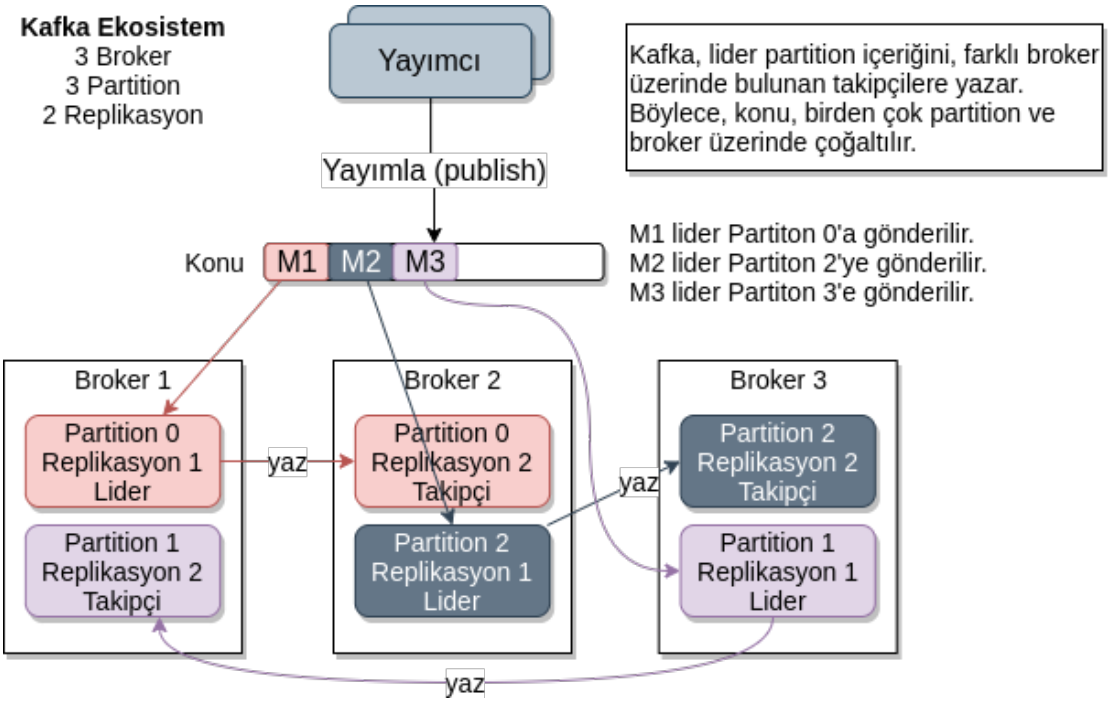
Apache Kafka, yayınla/abone ol paradigması üç temel yeteneği birleştirir: olay akışlarını yayınlama ve bunlara abone olma, olay akışlarını saklama ve olay akışlarını işleme.

Kafka'da kullanılan ana kavramlar ve terminolojiler şu şekilde sıralanabilir:

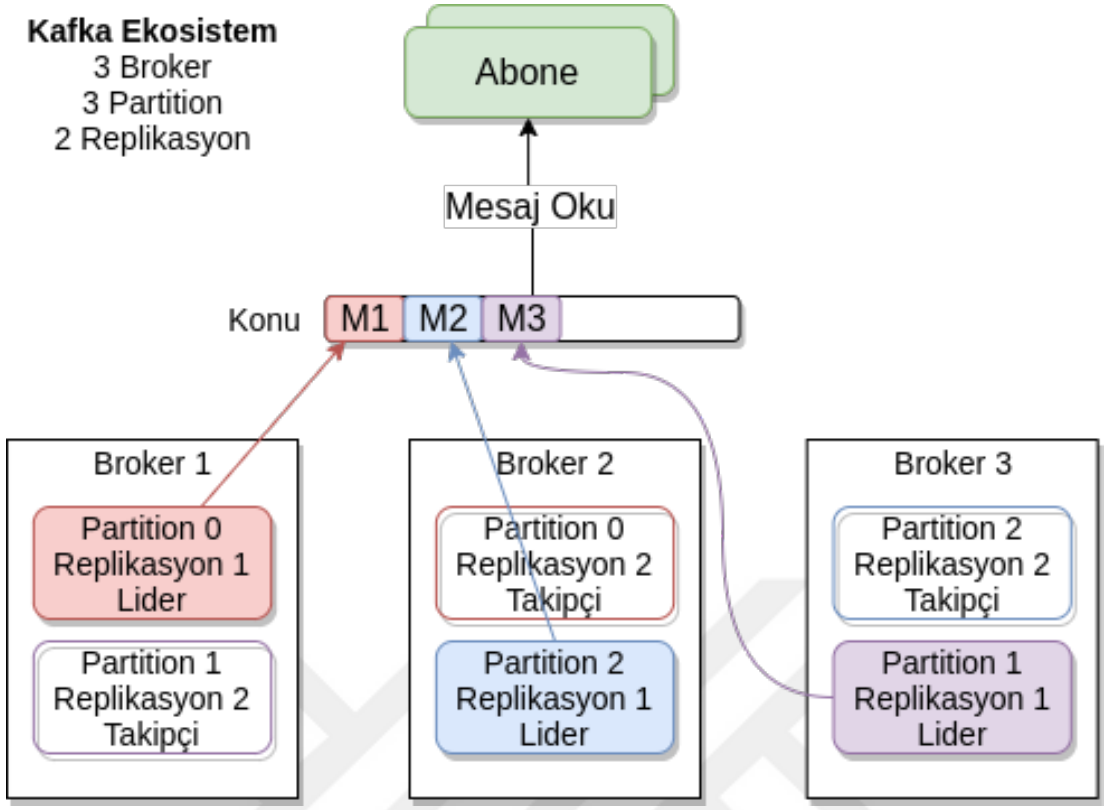
- Olay (Event): Kafka'ya veri okuduğunda veya yazıldığında, bu olaylar şeklinde yapılır. Kavramsal olarak, bir olayın bir anahtarı, değeri, zaman damgası ve isteğe bağlı meta veri başlıkları vardır.
- Yayımcılar (Publishers/Producers) ve Aboneler (Subscribers/Consumers): Olayları Kafka'ya yayınlayan istemci uygulamalardır ve tüketiciler, bu olaylara abone olan (okuyan ve işleyen) uygulamalardır. Kafka'da üreticiler ve tüketiciler birbirinden tamamen ayrılmış ve birbirlerinden bağımsızdır; ölçeklenebilirliği elde etmek için önemli bir tasarım öğesidir.
- Konu (Topic): Olaylar, konularda organize edilir ve kalıcı bir şekilde saklanır (varsayılan süre 2 hafta). Bir konunun kendisine olay yazan sıfır, bir veya çok sayıda yayınlayıcısının yanı sıra bu olaylara abone olan sıfır, bir veya çok sayıda abonesi olabilir. Geleneksel mesajlaşma sistemlerinin aksine, olaylar abonenin mesajı almasından sonra silinmez.
- Partition: Konular Apache Kafka'da bölümlenebilir. Bir konuya veri yayınlandığında mesaj partition alanlarından birine yazılır. Eğer anahtar, değer eşleşmesi var ise anahtar değerine göre, yoksa Round Robin kuralına göre sırası ile partition alanlarına mesaj eklenir. Partition sayesinde bir düğümün kapasitesi ile sınırlı olmayarak birden fazla düğümde veri depolanır, böylece ölçeklenebilirlik sağlanmış olur.

- Replikasyon: Broker'dan birinin arızalanması ve istekleri yerine getirememesi durumunda yalnızca kullanılabilirlik amacıyla verilerin birden çok kopyasına sahip olma işlemidir.
- Broker: Kafka aracısı, Kafka sunucusu olarak da bilinir Kafka aracısı farklı bilgisayar sistemleri arasındaki konuşmaya aracılık etmekten ve mesajın doğru taraflara teslim edilmesini garanti etmekten sorumlu olan bir Mesaj Aracısı olarak tanımlanır. Adından da anlaşılacağı gibi, üretici ve tüketici doğrudan etkileşimde bulunmazlar, Kafka sunucusunu mesaj servislerini değiş tokuş etmek için bir aracı veya aracı olarak kullanırlar. Diğer mesaj sistemlerinden farklı olarak, Kafka brokerleri durumsuzdur. Bu yüzden küme durumlarını korumak için ZooKeeper'ı kullanırlar.
- ZooKeeper: Apache tarafından geliştirilen üst düzey bir yazılımdır. Zookeeper, Kafka küme düğümlerinin durumunu ve ayrıca Kafka konularını, bölümlerini vb. takip eder. Zookeeper'ın sorumlulukları şunlardır: Broker koordinasyonu, lider partition seçme, brokerlerin birbirini tanımasını sağlamak, yeni veya silinmiş broker veya yeni eklenen, değiştirilen konuları keşfetmek.

Şekil 4.11 ve Şekil 4.12 Apache Kafka ekosistemde yayımcı ve abone üzerinden veri yönetimi gösterilmektedir.



Şekil 4.11. Kafka yazma ölçeklenebilirliği - takipçilere eşzamanlı çoğaltma sağlar



Şekil 4.12. Kafka okuma ölçeklenebilirliği- partitions eşzamanlı tüketime olanak tanır

Partition'a kaydedilen mesajların her birine, bölüm içindeki her mesajı benzersiz bir şekilde tanımlayan ofset adı verilen sıralı bir kimlik numarası atanır. Abone bazında tutulan tek meta veri, tüketicinin günlükteki "offset" olarak adlandırılan konumudur. Bu ofset abone tarafından kontrol edilir: normalde bir abone, mesajları okurken ofsetini doğrusal olarak ilerletir, ancak aslında konum abone tarafından kontrol edilir ve mesajları istediği sırayla tüketebilir. Örneğin, bir tüketici yeniden işlemek için daha eski bir ofseti sıfırlayabilir.

Partition birkaç amaca hizmet eder. İlk olarak, kayıtların tek bir sunucuya sığacak bir boyutun ötesine ölçeklenmesine izin verirler. Her bir bölüm, kendisini barındıran sunuculara sığmalıdır, ancak bir konunun, rastgele miktarda veriyi işleyebilmesi için birçok bölümü olabilir. İkincisi, paralellik birimi olarak hareket ederler.

Her Partition'ın "lider" olarak hareket eden bir sunucusu ve "takipçi" olarak hareket eden sıfır veya daha fazla sunucusu vardır. Lider, bölüm için tüm okuma ve yazma isteklerini işlerken, takipçiler lideri pasif olarak çoğaltır. Lider başarısız olursa, takipçilerden biri otomatik olarak yeni lider olur. Her sunucu, bazı bölümleri için bir lider ve diğerleri için

bir takipçi olarak hareket eder, bu nedenle yük, küme içinde iyi dengelenir. Partition atama stratejilerinde, Kafka İstemcileri üç yerleşik strateji sunar: Range, RoundRobin ve StickyAssignor.

- RangeAssignor: varsayılan stratejidir. Bu stratejinin amaçları, çeşitli konuların bölümlerini birlikte yerelleştirmektir. Örneğin, aynı sayıda partition ve aynı anahtar bölümleme mantığına sahip iki konudaki kayıtları birleştirmek için kullanışlıdır. Bunu yapmak için, strateji ilk önce tüm tüketicileri broker koordinatörü tarafından atanan üye_kimliği kullanarak sıraya koyacaktır. Ardından, mevcut konu bölümlerini alfabetik sıraya koyacaktır. Son olarak, her konu için, ilk tüketiciden başlayarak partition atanması gerçekleştirilir.
- RoundRobinAssignor: kullanılabilir partition tüm üyeler arasında eşit olarak dağıtılmak için kullanılabilir. Daha önce olduğu gibi, atayan her bölümü atamadan önce partitipn ve tüketicileri alfabetik sıraya koyacaktır.
- StickyAssignor, tek tip bir dağıtım sağlarken iki atama arasındaki partition hareketlerini en aza indirmeye çalışması dışında RoundRobin'e oldukça benzerdir.

Bu maddelerin dışında da partition ataması için özel strateji uygulanabilmektedir.

Basit bir ifade ile özetlenirse, partition ölçeklenebilirlik için kullanılır ve replikasyon kullanılabilirlik (verinin yedeklenmesi) içindir. Broker, partition ve replikasyon parametreleri kullanıcı tanımlıdır ve belirlenen sayılara göre ekosistemde yer almaktadırlar. Bu sebeple uçtan uca gecikmeyi gönderilen bir mesajın mesajın alınmasına kadar geçen süre izlendi ve haberleşme kanalının verimi bu parametre değerlerine göre ölçüldü. Analiz sonuçları, sonuçlar bölümünde detaylı olarak verilmiştir.

Veriyi almak (tüketmek), yayınlamaktan biraz daha farklıdır. Burada thread yapısı devreye girmektedir. Multithreading, "bir merkezi işlem biriminin (CPU) (veya çok çekirdekli bir işlemcideki tek bir çekirdeğin), işletim sistemi tarafından desteklenen, aynı anda birden çok yürütme thread sağlama yeteneğidir." Çalışmanın daha küçük birimlere bölünebileceği, paralel olarak çalıştırılabileceği durumlarda, veri tutarlılığı üzerinde olumsuz bir etki olmadan, uygulama performansını artırmak için Multithread kullanılabilir.

Apache Kafka'da kullanılan yayımcıya ve aboneye ait örnek kod parçası Algoritma 1'de verilmiştir.

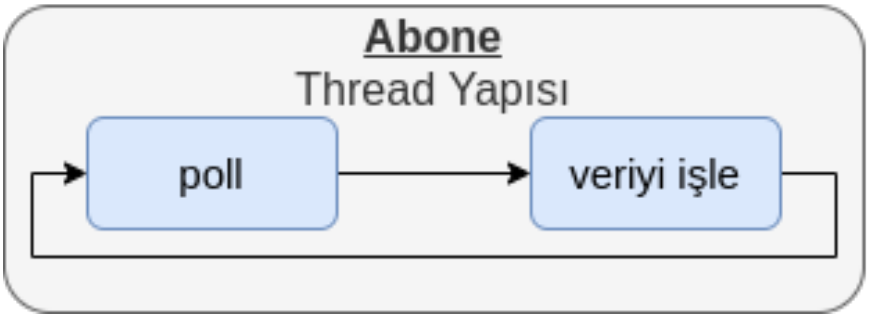
Algoritma 1. Yayınla/Abone ol mimarisinde bir yayımcının mesaj yayınlaması

```
1.  Konfigürasyon ayarları tanımla
2.      broker = "kafka:9092"
3.      Configuration config = {"metadata.broker.list", broker}
4.  Yayımcı tanımlaması
5.      Producer producer(config);
6.  Mesaj içeriği oluşturulur ve veri serileştirilir
7.      vector<MessageBuilder> builders;
8.      builders.payload(Buffer(serialize(chunk)));
9.  Mesaj yayınlanır
10.     producer.produce(builders);
```

Multithread ile bir abone mimarisi uygularken, kafka abonesinin thread güvenli olmadığına dikkat etmek önemlidir. Multithread erişimi düzgün bir şekilde senkronize edilmelidir, bu da zorlu bir görev olduğundan dolayı Single-thread modelinin yaygın olarak kullanılmasının nedeni budur.

Tipik bir single-thread uygulama, poll döngüsünün etrafında toplanmıştır. Şekil 4.13'te gösterildiği üzere temel olarak, iki eylemi tekrarlayan sonsuz bir döngüdür:

- poll yöntemi ile verileri alma
- Alınan verileri işleme



Şekil 4.13. Kafka veri tüketimi için oluşturulan thread

Algoritma 2. Bir abonenin, konudan mesaj alması

```
1. Konfigürasyon ayarları tanımlanır
2. broker = "kafka:9092"
3. Configuration config = {"metadata.broker.list", broker}
4. //Konuya abone ol
5.  consumer.subscribe({topic_name});
6. //Thread döngüsünü başlat
7.  while (!done)
8.          // Try to consume a message
9.           Message msg = consumer.poll();
10.          Buffer &buffer = msg.get_payload();
11.          string chunk = string(buffer);
12.          Payload_Video chunk = parse_protobuf_as_chunk(chunk_s);
13.          consumer.commit(msg)
14. end while
```

Algortima 2, bir abonenin konuya nasıl abone olduğu ve veriyi tüketme algortimasını vermektedir. Karmaşıklık analizi incelendiğinde şu kurallar takip edilir, kod yorumları dikkate alınmaz. Tanımlayıcı ve işlev bildirimleri dikkate alınmaz. Tüm değişkenler ve sabitler işlenen olarak kabul edilir. Aynı programın farklı modüllerinde kullanılan global değişkenler, aynı değişkenin birden çok oluşumu olarak sayılır. Farklı işlevlerde aynı ada sahip yerel değişkenler benzersiz işlenenler olarak sayılır. İşlev çağrıları operatörler olarak kabul edilir. Tüm döngü ifadeleri örneğin, do while, while, for ve tüm kontrol ifadeleri örneğin, if, if else, vb. operatörler olarak kabul edilir.

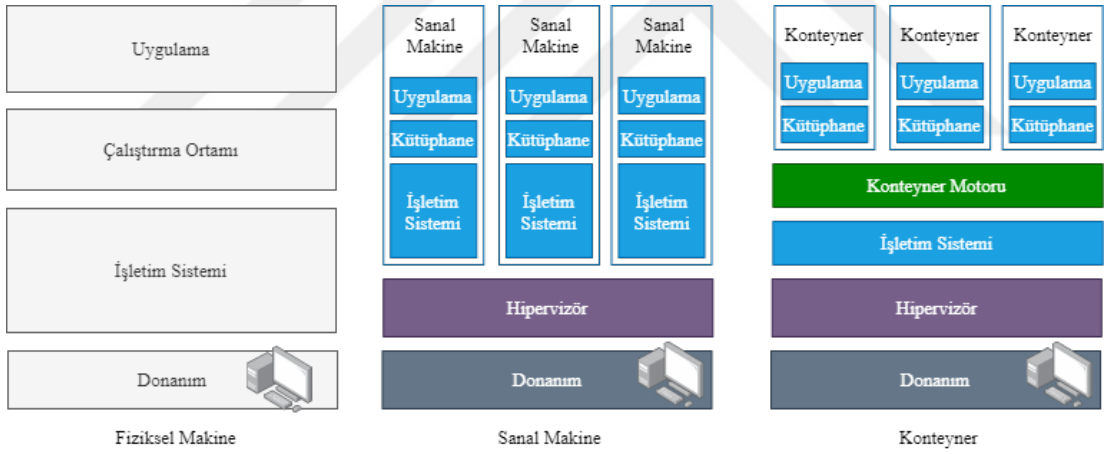
Öyleyse, 7 değişken tanımlama, bir döngü ve 2 atama işlemi gerçekleştiğini söyleyebiliriz. Bu durumda, n döngü sayısını ifade ederse Big-O karmaşıklık analizind, Algoritma 2, O(n) karmaşıklığına sahiptir diyebiliriz.

Veri serileştirme ve ters serileştirme adımları, paket tasarımı yapısı alt konu başlığında açıklanmıştır. Kafka, yüksek performanslı bir TCP ağ protokolü aracılığıyla iletişim kuran sunucular ve istemcilerden oluşan dağıtılmış bir sistemdir. Bulut ortamlarının yanı

sıra, sanal makinelerde ve konteynerlerde devreye alınabilir. Mimarinin konteynerlar aracılığı ile nasıl oluşturulduğu ve yönetildiği bir sonraki bölümde açıklanmıştır.

4.2.2 Mikroservis ve konteyner yönetimi

Mikroservisler ayrı ayrı konuşlandırılabilir, ölçeklenebilir bağımsız birimler olmalıdır. Docker, mikroservislerin çalışması için hafif (lightweight) bir çözümdür ve bir ana makinede konteynerleştirilmiş uygulamaları çalıştırmak için sanallaştırmayı kullanan bir uygulamadır. Konteynerleştirme, kullanıcıların bir ağ üzerinden iletişim kurmalarına izin verirken, uygulamaları tamamen ayrı ayrı oluşturmasına, çalıştırmasına ve test etmesine olanak tanır. Docker, Linux işletim sisteminin kaynak izolasyon özelliklerini kullanarak ek bir soyutlama katmanı ve sanallaştırma otomasyonu sağlar. Docker konteynerleri konuşlandırılmış uygulamalardan oluşur. Şekil 4.14'te gösterildiği üzere, "konteyner tabanlı sanallaştırma" veya "işletim sistemi düzeyinde sanallaştırma" olarak adlandırılır ve sanal makine teknolojisinden farklıdır.



Şekil 4.14. Konteyner ve Sanal Makinenin Mimari Farkı

Konteyner tabanlı sanallaştırma, işletim sistemi düzeyinde sanallaştırma olarak da bilinen ana makine içinde yazılım düzeyinde sanal ortam oluşturan hafif bir sanallaştırma yaklaşımıdır. Kapsayıcılar (misafir sistemler gibi davranan) biçiminde sanal makineler oluştururlar, böylece temel ana bilgisayar işletim sisteminin kaynaklarını paylaşarak hiper yönetici kullanmanın ek yükünü ortadan kaldırır. Konteynerler birbirinden yalıtılmıştır ve kendi yazılımları, kütüphaneleri ve konfigürasyon dosyalarından oluşur. Birbirleriyle tanımlanmış kanallar aracılığıyla iletişim kurabilirler. Tüm Konteynerler tek bir işletim sistemi çekirdeğinin hizmetlerini paylaştığı için sanal makinelerden daha az kaynak

kullanırlar. Docker kullanmanın avantajları arasında hızlı uygulama dağıtımı, makineler arasında taşınabilirlik, bileşenlerin yeniden kullanımı ve sürüm kontrolü, minimum ek yük, paylaşım ve basitleştirilmiş bakım sayılabilir. Tablo 4.2, sanal makine ve konteyneri işletim sistemi, başlama zamanı, depolama ve iletişim yönünden karşılaştırma sonuçlarını göstermektedir.

Tablo 4.2. Sanal Makine ve Konteyner karşılaştırma

	Sanal Makineler	**Konteynerler**
İşletim Sistemi (OS)	Her sanal makine, bir sunucu donanımı (Tip-1 hipervizörleri) veya bir ana işletim sistemi (Tip-2 hipervizörleri) üzerinde çalışır ve çekirdek, tam donanım kaynaklarıyla her VM'ye ayrı ayrı atanır.	Her kapsayıcı, konuk işletim sistemleri arasında paylaşılan çekirdek ve diğer donanım kaynaklarıyla, bir ana bilgisayar işletim sisteminin üzerinde çalışır.
Başlama zamanı	Donanım özelliklerine göre fark etse dahi, Sanal makineyi başlatmak Konteyner'a göre yavaştır	Sanal Makineye göre daha hızlıdır.
Depolama	Hiper yönetici tabanlı sanallaştırma, çekirdek dahil tüm sistem bileşenlerinin kurulması ve çalıştırılması gerektiğinden çok daha fazla depolama alanı gerektirir.	İşletim sistemi ve kernel paylaşımlı olduğundan dolayı, sanal makineye göre daha az depolama alanı gerektirir.
İletişim	Tipik olarak, VM'ler farklı sunucularda çalışıyorsa iletişim için ağ cihazları kullanılır, ancak aynı sunucu olması durumunda sinyaller, soketler, pipe vb. gibi standart IPC mekanizmaları kullanılır.	Konteynerler için de durum aynıdır ancak standart IPC mekanizmaları daha genel olarak kullanılmaktadır.

Her Docker konteyner, Docker konteyner görüntüsünün (image) nasıl oluşturulduğuna ilişkin yönergeleri içeren bir metin dosyasıyla başlar. DockerFile, Docker görüntü oluşturma işlemini otomatikleştirir. Esasen, Docker motorunun görüntü dosyalarını birleştirmek için çalıştıracağı komutların bir listesidir.

Docker görüntüleri (Images) yürütülebilir uygulama kaynak kodunun yanı sıra uygulama kodunun kapsayıcı olarak çalışması için gereken tüm araçları, kütüphaneleri ve bağımlılıkları içerir. Docker görüntüsünü çalıştırdığınızda, konteynerin bir örneği (veya birden çok örneği) olur. Docker konteynerları, Docker görüntülerinin canlı, çalışan örnekleridir. Docker görüntüleri salt okunur dosyalar olsa da, konteynerler canlı, geçici, yürütülebilir içeriktir. Kullanıcılar onlarla etkileşime girebilir ve yöneticiler ayarlarını ve koşullarını ayarlayabilir. Tablo 4.3'te Kafka için kullanılan Dockerfile verilmiştir. Birden çok konteyner kullanılıyorsa, Docker Compose kullanılabilir. Docker Compose,

uygulamaya hangi hizmetlerin dahil edildiğini belirten bir YAML dosyası oluşturur ve konteynerleri tek bir komutla dağıtıp çalıştırabilir.

Tablo 4.3. Örnek Dockerfile yapısı

Dockerfile Kafka

```
FROM openjdk...
ARG kafka_version=2.5.0
ARG scala_version=2.12
....
LABEL org.label-schema.name="kafka" \
      org.label-schema.description="Apache Kafka" \
      .....
ENV KAFKA_VERSION=$kafka_version \
    ....
ENV PATH=${PATH}:${KAFKA_HOME}/bin
...
COPY download-kafka.sh start-kafka.sh broker-list.sh create-topics.sh versions.sh /tmp/
RUN apk add --no-cache bash curl jq docker \
 && chmod a+x /tmp/*.sh \
 ...
COPY overrides /opt/overrides
CMD ["start-kafka.sh"]
```

Docker Compose kullanarak, depolama için kalıcı birimleri tanımlayabilir, temel düğümleri belirlenebilir ve hizmet bağımlılıkları belgelenebilir ve yapılandırabilir. Tablo 4.4'te Docker-Compose dosyasından örnek verilmiştir. Bu dosyada Kafka Broker ve Zookeeper konteynerlerin beraber nasıl çalıştığı ve hangi portlar üzerinden haberleşecekleri açıklanmıştır.

Varsayılan olarak, Kafka sunucusu 9092 numaralı bağlantı noktasında başlatılır. Kafka, ZooKeeper'ı kullanır ve bu nedenle 2181 numaralı bağlantı noktasında ZooKeeper sunucusu başlatılır. Birden fazla kafka kullanılması durumunda docker-compose dosyasında port ayarlaması Tablo 4.5'te gösterilmiştir.

Tablo 4.4. Örnek Docker-Compose dosyası

Docker-Compose Kafka

```
services:
  zookeeper1:
    image: wurstmeister/zookeeper:3.4.6
    ports:
     - 2181:2181
    restart: always
  kafka1:
    image: wurstmeister/kafka:1.0.0
    environment:
      KAFKA_ZOOKEEPER_CONNECT: zookeeper1:2181
      KAFKA_ADVERTISED_HOST_NAME: 30.10.23.18
      ....
    ports:
      - 9092:9092
    external_links:
      - zookeeper1:zookeeper1
```

Şekil 4.15'te ayağa kaldırılan docker-compose dosyasına ait çıktılar verilmiştir. Docker-compose dosyası up komutu ile başlatılır. Stop komutu ile durdurulur. Böylece konteynerlar işleme alınır veya sonlandırılır.

```
seda@seda-Lenovo:~/projects/kafka$ docker-compose up
Starting kafka_zookeeper1_1    ... done
Starting kafka_kafka-manager_1 ... done
Starting kafka_zookeeper2_1    ... done
Recreating kafka_kafka3_1      ... done
Recreating kafka_kafka1_1      ... done
Recreating kafka_kafka2_1      ... done
Attaching to kafka_zookeeper2_1, kafka_kafka-manager_1, kafka_zookeeper1_1, kafka_kafka3_1,
kafka1_1         | waiting for kafka to be ready
kafka2_1         | waiting for kafka to be ready
kafka3_1         | waiting for kafka to be ready
zookeeper1_1     | JMX enabled by default
zookeeper2_1     | JMX enabled by default
zookeeper1_1     | Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg
zookeeper1_1     | 2021-10-17 19:01:10,480 [myid:] - INFO  [main:QuorumPeerConfig@103] - Rea
zookeeper2_1     | Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg
zookeeper1_1     | 2021-10-17 19:01:10,489 [myid:] - INFO  [main:DatadirCleanupManager@78] -
zookeeper1_1     | 2021-10-17 19:01:10,489 [myid:] - INFO  [main:DatadirCleanupManager@79] -
zookeeper2_1     | 2021-10-17 19:01:09,207 [myid:] - INFO  [main:QuorumPeerConfig@103] - Rea
zookeeper2_1     | 2021-10-17 19:01:09,220 [myid:] - INFO  [main:DatadirCleanupManager@78] -
zookeeper2_1     | 2021-10-17 19:01:09,220 [myid:] - INFO  [main:DatadirCleanupManager@79] -
zookeeper2_1     | 2021-10-17 19:01:09,222 [myid:] - WARN  [main:QuorumPeerMain@113] - Eithe
zookeeper1_1     | 2021-10-17 19:01:10,494 [myid:] - WARN  [main:QuorumPeerMain@113] - Eithe
```

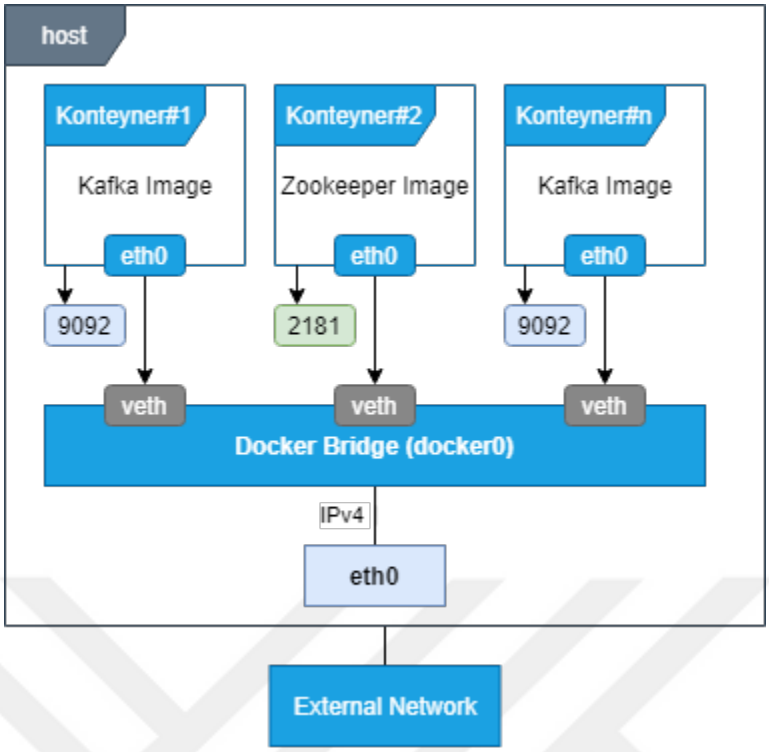
Şekil 4.15. Docker Compose ayağa kaldırılmasında oluşan kayıtlar

Tablo 4.5. Docker Compose

Birden fazla broker olması durumunda port numaralandırma

```
kafka1:
  image: kafka:1.0.0
  environment:
    KAFKA_ZOOKEEPER_CONNECT: zookeeper1:2181
    KAFKA_ADVERTISED_HOST_NAME: 30.10.23.18
    ...
  ports:
    - 9092:9092
  external_links:
    - zookeeper1:zookeeper1
  env_file:
    - ./conf.env
  ...
kafka2:
  image: kafka:1.0.0
  environment:
    KAFKA_ZOOKEEPER_CONNECT: zookeeper1:2181
    KAFKA_ADVERTISED_HOST_NAME: 30.10.23.18
    ...
  ports:
    - 9093:9092
  external_links:
    - zookeeper1:zookeeper1
  env_file:
    - ./conf.env
  ...
kafka3:
  image: kafka:1.0.0
  environment:
    KAFKA_ZOOKEEPER_CONNECT: zookeeper1:2181
    KAFKA_ADVERTISED_HOST_NAME: 30.10.23.18
    ...
  ports:
    - 9094:9092
  external_links:
    - zookeeper1:zookeeper1
  env_file:
    - ./conf.env
  ...
```

Konteynerlerin iletişim kurabilmesi için aynı networkte olmaları gerekmektedir. Docker ile konteynerden konteynere iletişim genellikle Şekil 4.16'da gösterildiği gibi sanal bir ağ (virtual Ethernet devices, veth) kullanılarak yapılır. Tablo 4.5'te verilen Docker-Compose kod parçasında Zookeeper için HOST_PORT ve CONTAINER_PORT 2181, Kafka için HOST_PORT ve CONTAINER_PORT 9092 olarak tanımlanmıştır. HOST_PORT'a atanan port numarası ile hizmete dışından da erişilebilir.

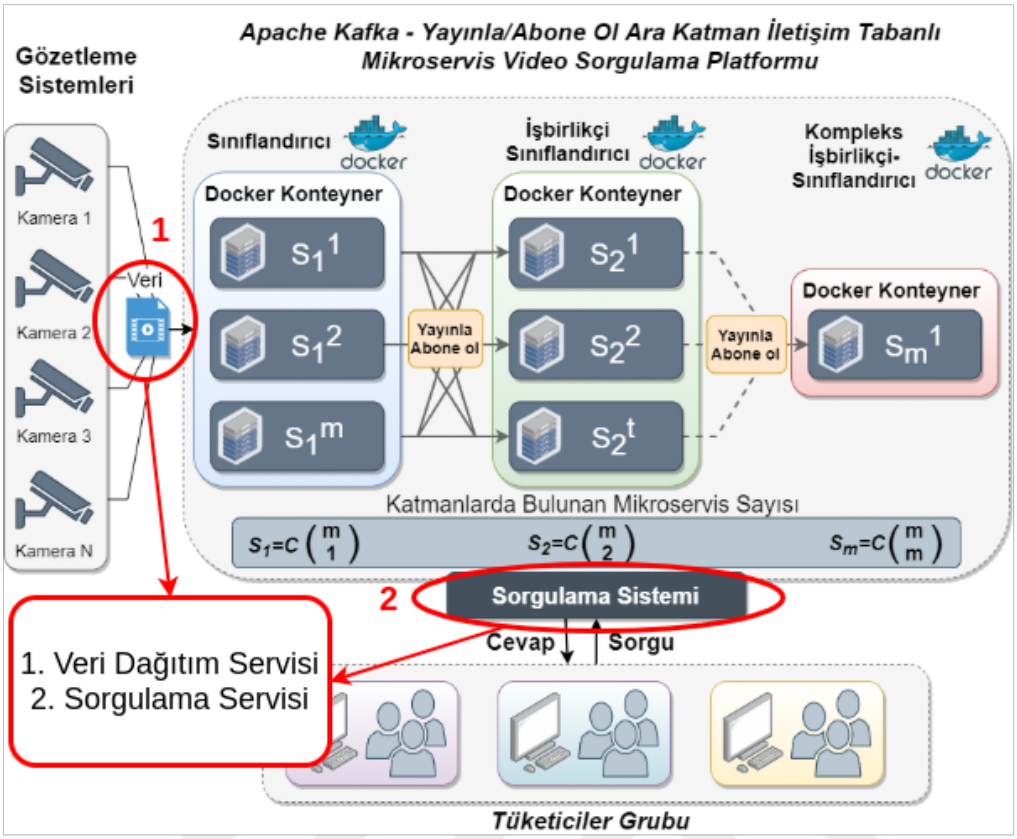


Şekil 4.16. Docker ağ konfigürasyonu

Kafka mimarisindeki partition ve replikasyon birimlerinin performansı etkileyebileceği durumu göz önünde bulundurarak broker, partition ve replikasyon sayısı, Apache Kafka'nın performansına (Records/sn (MB/sn) ve Msgs/sn (records/sn)) göre değerlendirerek seçildi. Performans sonuçlarıp Sonuçlar bölümünde verilmiştir.

Monolitik bir uygulamayı servislere ayırmak önemli bir adımdır fakat servisler arasındaki iletişimin alt yapısını ve kurallarını tanımlamak da bir o kadar önem arz etmektedir. Önerilen mikroservis ekosisteminin içeriğinde:

- Geliştirilen mimaride, 2^n-1 formülü sınıflandırmada görev yapacak mikroservis sayısını verir. Burada n, özellik sayısını belirtir ve bu çalışmada seçilen özellikler: tip, renk ve hız olduğundan, n = 3 olur. Bu durumda ekosistemde 2^3- 1 = 7 adet sınıflandırma servisi bulunur.
- Ayrıca, Şekil 4.17'de kameralardan gelen video görüntüsünü "araç" konusuna yayınlayacak bir Veri Dağıtım servise ve operatörlerin sistem üzerinden görüntü sorgulasını sağlayan web api hizmetini çalıştıracak Sorgulama Servisine ihtiyaç duyulmaktadır.



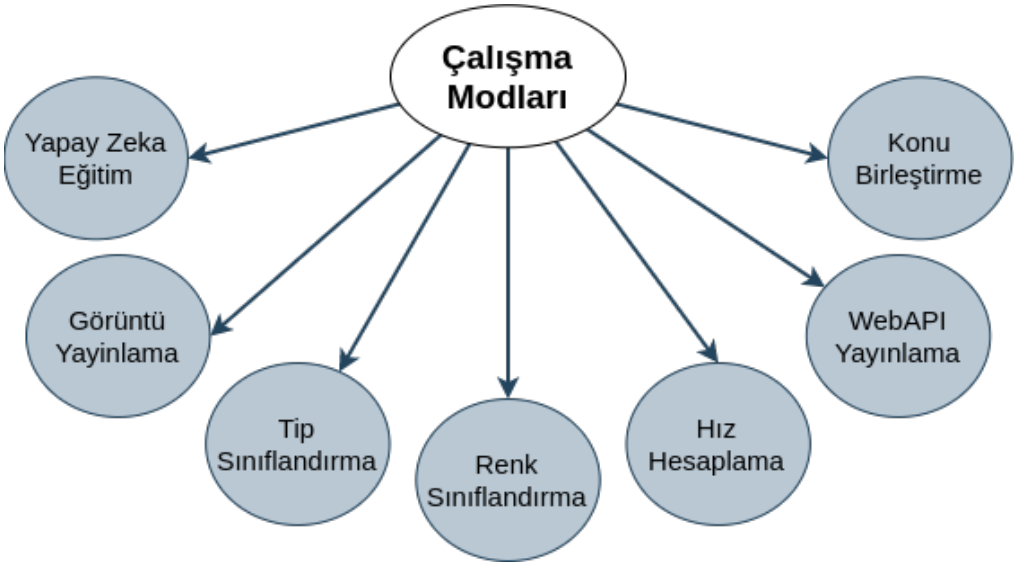
Şekil 4.17. Veri Dağıtım ve Sorgulama Servisi

Bu durumda toplamda mimari 9 mikroservisten oluşur. Bunlardan 8'i sorgulama sistemini yönetimini gerçekleştirir ve operatörler ile doğrudan erişim halinde değildir. Sorgulama Servisi ise bir web api hizmeti sunarak, operatörlerin sistemi kullanmalarını sağlamaktadır.

İletişim kurallarını oluşturma sırasında, servislerin hangi görevleri yaptığı mimari Çalışma Modları ile belirlenir. Şekil 4.18'de çalışma modlarının listesi verilmiştir.

Farklı modlar farklı görevler ile sorumludur. Sırası ile modların görevleri:

- Yapay Zeka Eğitim: Mimari farklı veri setleri ile de uyumlu çalışabilmeli, sadece eğitilen tek bir yapay zeka modeline bağlı kalmaması adına eğitim modu geliştirildi.
- Görüntü Yayınlama: Kameradan alınan görüntüler üzerinde araç tespiti işlemi gerçekleştirir ve mesajları frame konusuna yayınlar.
- Tip Sınıflandırma: frame konusundan alınan görüntü parçaları üzerinde araç tip sınıflandırılması gerçekleştirilir. Sonuçlar araçların tip etiketlerine göre yayınlanır.



Şekil 4.18. Çalışma Modları

- Renk Sınıflandırma: frame konusundan alınan görüntü parçaları üzerinde araç renk sınıflandırılması gerçekleştirilir. Sonuçlar araçların renk etiketlerine göre yayınlanır.
- Hız Hesaplama: frame konusundan alınan görüntü parçaları üzerinde araç hız tespiti gerçekleştirilir. Sonuçlar araçların hız değerlerine göre yayınlanır.
- Birleştirme: Konuların kombinasyonlara göre birleştirilmesinden sorumludur.
- WebAPI Yayınlama: Sonuçların web api ye yayımlanarak kullanıcının gözlemleyebilmesi sağlanmıştır.

Bu bölüm başlığı altında, Apache Kafka terminolojisi ve süreçlerin konteyner haline getirilme aşaması anlatılmıştır. Bir sonraki başlıkta, mesajlaşma için kullanılacak paketlerin yapısı ve paket gönderimi sırasında takip edilecek kurallar açıklanmıştır.

4.2.3. İletişim Kuralları ve Akan Veriye Ait Veri Modelinin Tasarlanması

Monolitik bir uygulamada, hizmetler dil düzeyinde yöntem veya prosedür çağrıları aracılığıyla birbirlerini çağırır. Geleneksel bir dağıtılmış sistem dağıtımında, hizmetler sabit, iyi bilinen konumlarda (hosts/port ve bağlantı noktaları) çalışır ve bu nedenle HTTP/REST veya bazı RPC mekanizmalarını kullanarak birbirlerini kolayca arayabilir. Bununla birlikte, modern bir mikroservis tabanlı uygulama, tipik olarak, bir hizmetin örneklerinin sayısının ve konumlarının dinamik olarak değiştiği sanallaştırılmış veya kapsayıcılı ortamlarda çalışır. Sonuç olarak, bir istemcinin bir hizmete istekte bulunabilmesi için bir hizmet bulma mekanizması kullanması gerekir.

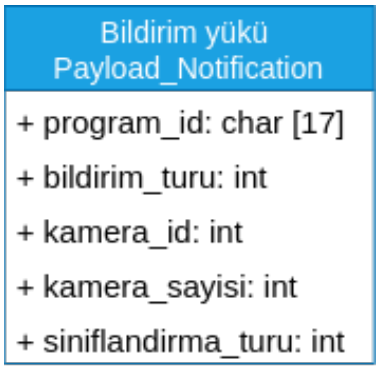
Hizmet keşfi, uygulamaların ve mikroservislerin bir ağ üzerinde birbirlerini nasıl konumlandırdıklarıdır. Uygulamalar, hem adreslerin küresel bir görünümünü koruyan merkezi sunucuları hem de adresleri güncellemek ve almak için merkezi sunucuya bağlanan istemcileri içerir.

Önerilen mimaride, katmanlı bir yapı sunulmuştur. Kompleks-İşbirlikçi servisinin görev yapabilmesi için, İşbirlikçi servislerinin; İşbirlikçi servislerinin görev yapabilmeleri için de Sınıflandırıcı servisinin görev yapması gerekir. Bu mimaride "konuşmadan önce dinle (listen-before-talk)" düzeni kurulmuştur ve böylelikle servisler arasında "el sıkışma" (handshake) adı verilen bir protokol geliştirilmiştir. Servislerin birbirlerine canlı ya da ölü olduklarına dair bildirim göndermeleri sağlanır. Buna bağlı olarak iki farklı haberleşme türü bulunur:

- İki Yönlü El Sıkışma
- Veri alışverişi

Detaylı açıklanmasından önce paket türleri açıklanmalıdır. Yük (payload) nedir, bilgi işlem ve telekomünikasyonda, yük, asıl amaçlanan mesaj olan iletilen verilerin bir parçasıdır. Başlıklar ve meta veriler yalnızca yük dağıtımını etkinleştirmek için gönderilir. (URL-16, URL-17) Haberleşme sırasında kullanılacak paket türleri de bu nedenle ikiye ayrılır:

- 1. Bildirim yükü (notification payload)



Şekil 4.19. Bildirim Yükü

Bildirim yükünde Şekil 4.19'da UML sınıf olarak gösterildiği üzere sırasıyla: her programa atanan 17 karakterden oluşan eşsiz bir id atanır. Birden fazla programın var olduğu bu sistemde mesajların hangi uygulamadan geldiğinin takibi bu şekilde

yapılmaktadır. bildirim_turu değişkeni programın hangi amaca hizmet verdiğini belirtmektedir.

Bildirim türleri UML sınıf diyagramı olarak Şekil 4.20'de gösterilmektedir. Sistem çoklu kamera veri akışını desteklemektedir. Bu durumda verinin hangi kameradan geldiğinin takibi yapılması gerekir. kamera_id, bu ayrımı yapmada kullanılmaktadır. kamera_sayisi, sistemde kaç adet kamera olduğuna dair bilgiyi tutmaktadır. siniflandırma_turu, servisin hangi katmanda görev yaptığını belirtir.

Şekil 4.20. Notifikasyon yükü

Bir servisin bir konuya abone olabilmesi için önce diğer eşleri bulması ve onlarla ağ bağlantıları kurması gerekir. Servis keşfi, uygulama bileşenlerinin birbirini bulmasına yönelik bir yöntemdir. Servis örnekleri sürekli olarak oluşturulup yok edildiğinden, bir mikroservis ortamında servis keşfi bir zorluktur. Bu hizmetlerin konumu ve kullanılabilirliği sürekli olarak güncellenmelidir. Bir mikroservis mimarisinde servis keşfinin önemli bir bileşeni, bir servis kaydıdır. Servis kaydı, kullanılabilir servis örneklerinin bir veritabanıdır. Her servisin anlık kullanılabilir örneklerini ve bağlantı ayrıntılarını içerir. Kayıt defteri, servislerin hala çalışıp çalışmadığını görmek için bir takip mekanizması tutar ve değilse, bunları kayıt defterinden kaldırır.

Yayınla/Abone ol sisteminin kendi başına eşleri keşfetme yolu yoktur. Bunun yerine, kendi adına yeni eşler bulmak için uygulamaya, ortam eş keşfi adı verilen bir sürece güvenir. Akranları keşfetmek için olası yöntemler şunları içerir:

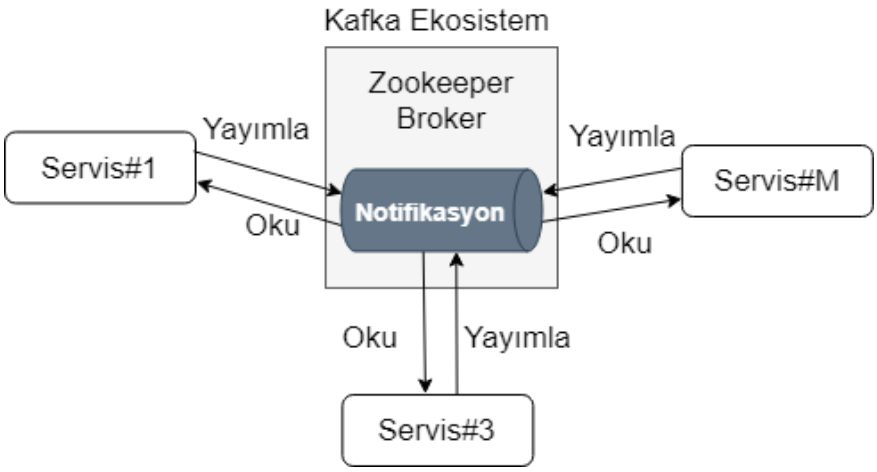
- Dağıtılmış karma tabloları (Distributed hash tables)
- Yerel ağ yayınları (Broadcast)
- Mevcut akranlarla akran listelerini değiş tokuş etme
- Merkezi izleyiciler veya buluşma noktaları
- bootstrap eşlerinin listesi

Servislerin birlikte çalışması ve veri yayınlaması, öncelikle birbirlerinden haberdar olmalarını sağlamıştır. İşbirlikçi sınıflandırıcıları çalışmaya başlamadan önce, önceki katmanın sınıflandırıcılarının ayakta olduğunu anlamak için bir sinyal beklemeye başlarlar. ACK sinyalini aldıktan sonra birleşme sunucuları ilgili konuyu dinlemeye başlar. Algoritma 4, servisler arasındaki konuşmadan önce dinle şeması için sözde kodu verir.

Notifikasyon türleri servisler arasında sinyalleşme amaçlı kullanılmıştır. Servislerin ben canlıyım ya da ölüyüm bildirimlerini birbirlerine iletir ve ağ durumu hakkında bilgi sahibi olunur. Bu durum sayesinde kullanıcının da sistemi sürekli olarak izleyebilmesine olanak sağlanır.

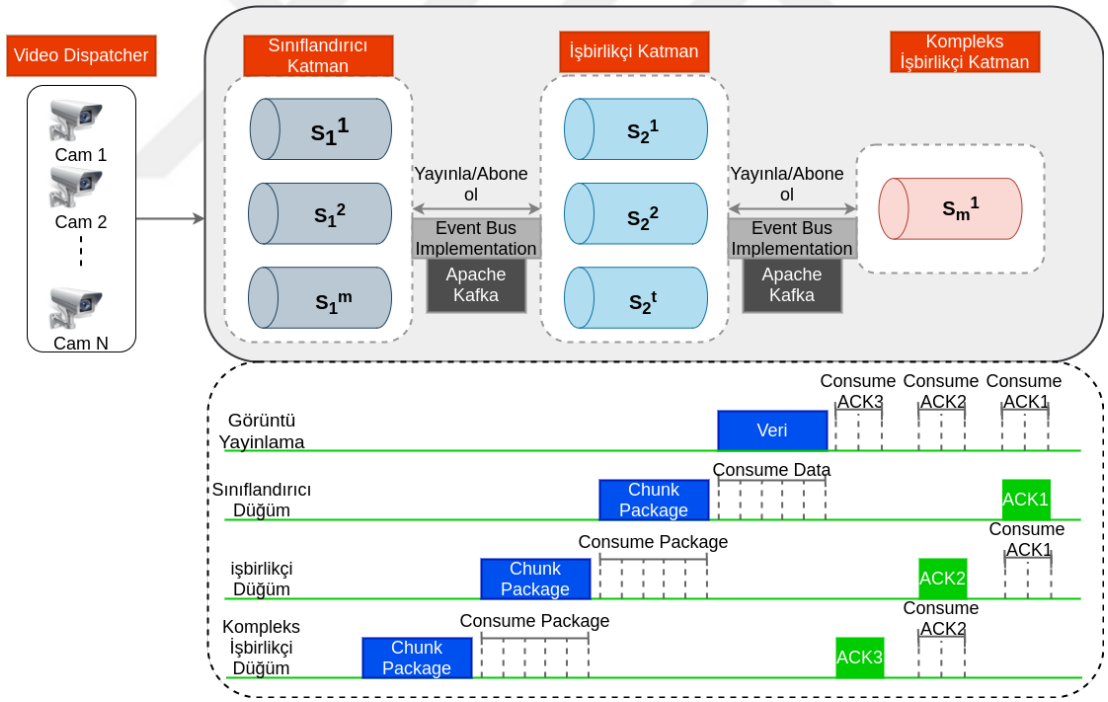
Bu sistemde notifikasyon türleri arasında ACK kullanılmıştır. Veri ağlarında ve bilgisayar veri yollarında ACK, bir iletişim protokolünün parçası olarak mesajın alındığını belirtmek için iletişim süreçleri, bilgisayarlar veya cihazlar arasında iletilen bir sinyaldir. Onaylar göndericiye alıcının durumunu bildirir, böylece kendi durumunu buna göre ayarlayabilir.

Servisler birbirlerinin IP ya da MAC adresini bilmemektedir. Bu nedenle birbirlerine doğrudan mesaj göndererek haberdar olamazlar. Bu durumda tüm servisler “Notifikasyon” adlı konuya abone olarak, adreslerini bilmeden birbirlerinin durumları hakkında bilgi sahibi olurlar. Şekil 4.21’da gösterildiği üzere bu konu üzerinde hem abone hem de üretici olarak görev yaparlar. Notifikasyon ile ilgili paketler bu konuya yayımlanır.



Şekil 4.21. Notifikasyon

Şekil 4.22'de gösterildiği üzere tüm sunucular ACK sinyalini gönderdikten sonra kameralardan yayına başlanır. Sınıflandırıcıların birbirlerine sinyal gönderme adımları Algoritma 3'te verilmiştir.



Şekil 4.22. ACK Sinyali

Her servisin ACK sinyalini nasıl beklediği ve nasıl cevap verdiği açıklanmıştır.

Algoritma 3'de Notifikasyon thread yapısı verilmiştir.

Algoritma 3. Bir abonenin, konudan mesaj alması

```
1.  notifikasyon_Oku_Thread("Notifikasyon", PROGRAM_ID)
2.      brokers = "kafka:9092"
3.      Configuration config = {"metadata.broker.list", brokers}}
4.      // Abone olustur
5.      Consumer consumer(config);
6.      // Konuya abone ol
7.      topic_name = “Notifikasyon”
8.      consumer.subscribe({topic_name});
9.      while (true) {
10.         // mesajlari okumaya calis
11.         Message msg = consumer.poll();
12.         if (msg) {
13.         // Mesaj gelirse oku
14.         const cppkafka::Buffer &buffer = msg.get_payload();
15.         }
16.         else print(“error”)
17.     }
18. //Notifikasyon yayimla
19. notifikasyon_Yayimla_Thread("Notifikasyon", PROGRAM_ID)
20.     brokers = "kafka:9092"
21.     Configuration config = {"metadata.broker.list", brokers}}
22.     Producer producer(config);
23.     // Konuya abone ol
24.     topic_name = “Notifikasyon”
25.     producer.subscribe({topic_name});
26.     while (true) {
27.         Payload_Notification notification;
28.         builder.payload(notification);
29.         producer.produce(builder) }
```

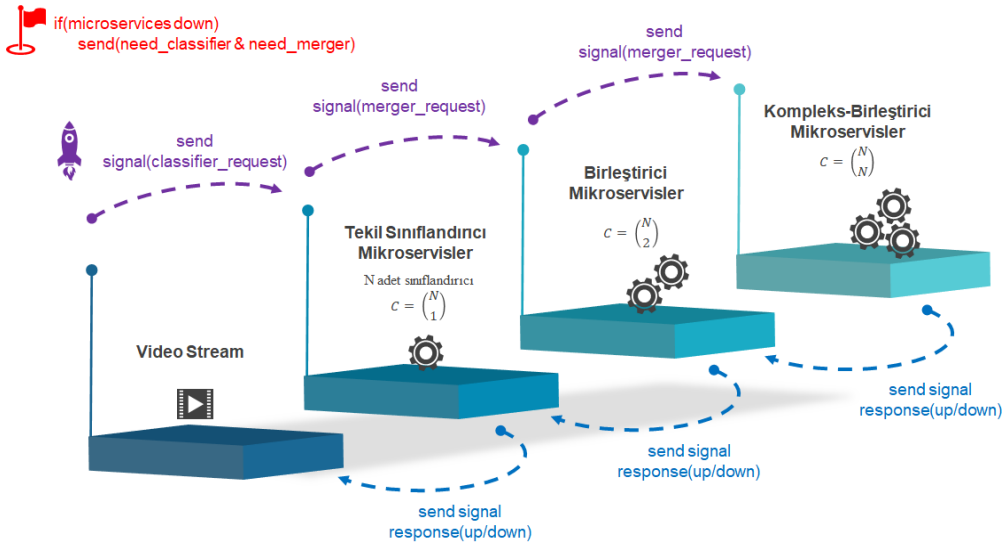
Sırası ile katmanlarda gerçekleşen olaylar: Görüntü yayınlama düğümü, öncelikle diğer servisleri keşfetmeye çalışır (discover mode). Her alınan ACK paketi ile servisler doğrulanır. Algoritma 3 ve Şekil 4.22 servis keşfi modunu göstermektedir.

Algoritma 4. Görüntü Yayınlama servisi (video dispatcher mode)

```
1.  Görüntü Yayınlama Servisini Başlat
2.  send_kamera_canli_signal(PROGRAM_ID, CAMERA_ID)
3.      Payload_Notification notification{};
4.      notification.notification_type = KAMERA_CANLI;
5.      producer.produce("Notifikasyon", notification)
6.  //Sürekli olarak Notifikasyon konusu dinlenir
7.  Thread listen(topic["Notifikasyon"]))
8.  If (tum_servis_mevcut != True)
9.      If (SINIFLANDIRICI != True)
10.         send (SINIFLANDIRICI_ISTEGI)
11.     else
12.         send (SINIFLANDIRICI_ISTEGI_ACK)
13.     If (ISBIRLIKCI != True)
14.         send (ISBIRLIKCI_ISTEGI)
15.     else
16.         send (ISBIRLIKCI_ISTEGI_ACK)
17.     If (KOMPLEKS_ISBIRLIKCI != True)
18.         send (ISBIRLIKCI_ISTEGI)
19.     else
20.         send (ISBIRLIKCI_ISTEGI_ACK)
```

Bu sayade Şekil 4.23'te gösterildiği üzere mikroservisler arasında el sıkışma mekanizması gerçekleştirilmiş olur.

Tüm ACK paketleri alındıktan sonra servisler veri yayımlamaya başlarlar. Bu adımdan sonra, veri dağıtımı için gerekli olan paket (yük) yapısı açıklanmıştır. Paket içinde, mesajlaşma için gerekli alanların ne olduğu tanımlanmıştır.



Şekil 4.23. El Sıkışma

- 2. Veri yükü (data payload): Servislerin hepsi canlı durumuna geçtikten sonra, veri dağıtımı gerçekleşmeye başlar. Video parçaları dağıtımında, video ile ilgili bilgilerinde gönderilmesi gerekmektedir. Veri yükü tasarımı Tablo4.6'da verilmiştir.

Tablo 4.6. Veri Yükü veri yapısı

Veri Yapısı Tanımı
1. **struct** Veri_Yuku {
2. int kamera_id;
3. int chunk_id;
4. int hiz_baslangic_noktası;
5. int hiz_bitis_noktası;
6. int hiz_hesaplama_mesafe;
7. int publisher_type;
8. vector<string> car_ids;
9. vector<int> siniflandirici1;
10. vector<int> siniflandirici2;
11. vector<int> siniflandirici3;
12. vector<**Chunk_Frame**> cerceveler; }

Kamera_id verinin hangi kamera akışından alındığına dair bilgiyi, chunk_id paketin kimliğini, source_id servisin kimliğini, hiz_baslangic_noktası hız hesaplama başlangıç

noktasını, hiz_bitis_noktası hız hesaplama bitiş noktasını, hiz_hesaplama_mesafe hız hesaplama esnasında kullanılan mesafenin uzunluğunu, publisher_type sınıflandırıcı türünü, siniflandirici1, 2 veya 3 araç görüntülerinin hangi sınıflandırıcı türüne ait olduğunu gösterir. Tüm bu meta veriler bize çerçeveler (frame) hakkında ön bilgi verir. Çerçevelerin tutulduğu veri yapısı türüne Chunk denir. Veri yükü tasarımı Tablo 4.7 'de verilmiştir.

Tablo 4.7. Chunk

Veri Yapısı Tanımı
1. **struct** Chunk_Frame {
2. vector<int> obj_frame_id
3. vector<int> obj_x
4. vector<int> obj_y
5. vector<int> obj_genislik
6. vector<int> obj_uzunluk; }

Tanımlamada obj nesneyi ifade eder. Obj_frame_id kimlik ataması için kullanılır. obj_x, tespit edilen nesneyi saran kutunun x koordinatını, obj_y y koordinatını, obj_genislik kutunun genişliğini, obj_uzunluk kutunun uzunluğunu temsil eder.

3.2.4. Mesajların işbirlikçi servisler tarafından birleştirilmesi ve yayınlanması

İşbirlikçi servislerin görevi, 1. Katmanda bulunan sınıflandırıcı servislerin yayınladıkları konuların kombinasyonunu yaparak yeni konular yayınlamaktır. Her araç, Görüntü Yayınlama servisinden yayınlanırken eşsiz bir ID ile yayınlanır. İşbirlikçi servisler birleştirmek istenilen konulara abone olurlar. Burada, görünütüler yeniden sınıflandırma algoritmalarından geçirilmeden, paketlerin ID numaraları kontrol edilerek görüntüler sınıflandırılır.

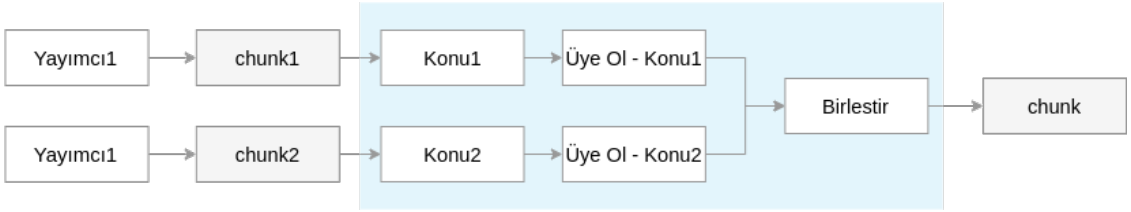
En iyi durum senaryosundan bahsedilirse, Sınıflandırıcı1, Görüntü Yayınlama servisinden görüntüyü alır ve sınıflandırma işlemini gerçekleştirir. Sınıflandırma sonucunu ve yük verilerini (data payload) yazarak paketi Konu1'e yayınlar. Bu paketin ID numarasını 52 kabul edelim. Sınıflandırıcı2 aynı şekilde Görüntü Yayınlama servisinden görüntüyü alır ve kendi yapay zeka modeline göre sınıflandırma işlemini gerçekleştirir. 52 numaralı paketin, Sınıflandırıcı2 tarafından da etiketlendiğini

varsayalım. Sınıflandırıcı2 veriyi konu2'ye yayınlar. Bu durumda 52 numaralı paket iki farklı konuya yayınlanmış olur. Konu1 ve Konu2'yi birleştirmek isteyen işbirlikçi sunucu bu iki konuya abone olur. Paketlerin ID numarası burada belirleyici rol oynar. Aynı ID numarasına sahip bir paket hem konu1 hem de konu2 etiketini taşıyorsa, bu görüntü iki sınıfın da etiketine sahiptir sonucu elde edilir.

Algoritma 4. Görüntü Yayınlama servisi (video dispatcher mode)

```
21. //İşbirlikçi Sınıflandırıcı
22. frame_reader()
23.     // Görüntü Yayınlama, tespit attığı araçları frame konusuna atar
24.     // Her sınıflandırıcı frame konusuna abone olur görüntüleri
25.     string brokers = "kafka:9092";
26.     Configuration config = { {"metadata.broker.list", brokers}}
27.     Consumer consumer(config);
28.     consumer.subscribe({topic_name});
29.     while(!done)
30.     Message msg = consumer.poll()
31.     Veri_Yuku chunk = chunk(msg.get_payload())
32. birlestirme(topic1, topic2)
33.     ortak_chunk = {}
34.     thread.start(frame_reader, topics1, topics2)
35.     if(chunk1.id == chunk2.id)
36.         ortak_chunk.add(chunk1)
37.     end
38.     producer.produce("topic1+topic1", ortak_chunk)
```

Şekil 4.24'te gösterildiği üzere Birleştirme çerçevesi, uygulamanın olayları nasıl işlediğine ilişkin bildirimsel bir yaklaşım sağlar. Potansiyel olarak birden çok temsilci geri araması veya tamamlama işleyicisi kapatması uygulamak yerine, belirli bir olay kaynağı için tek bir işleme zinciri oluşturabilir. Zincirin her bir parçası, önceki adımdan alınan öğeler üzerinde ayrı bir eylem gerçekleştiren bir Birleştirme operatörüdür.



Şekil 4.24. Birleştirme çerçevesi

Bir sonraki başlıkta verilerin network üzerinden iletilebilmesi serileştirme ve seri durumdan çıkarması gösterilmiştir.

4.2.5. Paketlerin Serileştirilmesi

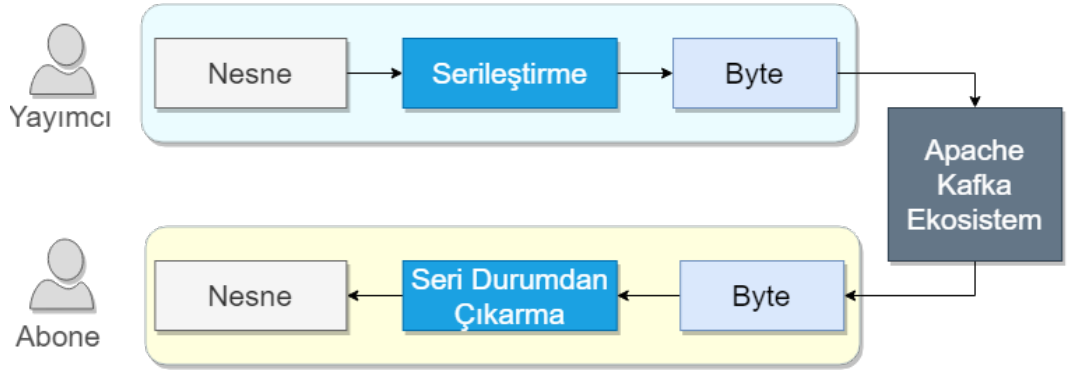
Serileştirme, bir veri nesnesini (veri depolama bölgesi içinde temsil edilen bir kod ve veri kombinasyonu) nesnenin durumunu kolayca iletilebilir bir biçimde kaydeden bir dizi bayta dönüştürme işlemidir. Bu serileştirilmiş formda, veriler başka bir veri deposuna (bellek içi bilgi işlem platformu gibi), uygulamaya veya başka bir hedefe teslim edilebilir. Ortaya çıkan bit dizisi, serileştirme formatına göre yeniden okunduğunda, orijinal nesnenin anlamsal olarak özdeş bir klonunu oluşturmak için kullanılabilir. Serileştirme, geliştiricinin bir nesnenin durumunu kaydetmesine ve gerektiğinde onu yeniden oluşturmasına olanak vererek, veri alışverişinin yanı sıra nesnelerin depolanmasını sağlar. Serileştirme yoluyla, bir geliştirici aşağıdaki gibi eylemleri gerçekleştirebilir:

- Bir web hizmeti kullanarak nesneyi uzak bir uygulamaya gönderme
- Bir nesneyi bir etki alanından diğerine geçirme
- Bir nesneyi bir güvenlik duvarından JSON veya XML dizesi olarak geçirme
- Uygulamalar arasında güvenlik veya kullanıcıya özel bilgilerin korunması

Bu özelliklerden bazılarının kullanışlı olması için mimari bağımsızlığın korunması gerekir. Örneğin şekil 3.25'te gösterildiği üzere, maksimum dağıtım kullanımı için, farklı bir donanım mimarisi üzerinde çalışan bir bilgisayar, serileştirilmiş bir veri akışını, endianlıktan bağımsız olarak güvenilir bir şekilde yeniden oluşturabilmelidir.

Protocol Buffers (Protobuf), yapılandırılmış verileri seri hale getirmek için kullanılan ücretsiz ve açık kaynaklı bir çapraz platform veri formatıdır. Bir ağ üzerinden birbirleriyle iletişim kurmak veya veri depolamak için programlar geliştirmede yararlıdır. Yöntem, bazı verilerin yapısını tanımlayan bir arayüz tanımlama dilini ve yapılandırılmış verileri temsil eden bir bayt akışını oluşturmak veya ayrıştırmak için bu açıklamadan kaynak

kodu üreten bir programı içerir. C++, C#, Dart, Go, Java ve Python gibi birçok popüler dili destekler.



Şekil 4.25. Serileştirme

Protobuf, mesajların yapısını açıklayan proto dosyasını oluşturmasına olanak tanır. Proto dosyaları, mesaj içeriğini okumak ve yazmak için sarmalayıcı (wrapper) sınıflar sağlayan yerel dil kodunda derlenir. Mesajdaki yeni alanlar isteğe bağlı olarak birden çok sürüm olarak ele alınabilir. Mesaj alanları belirtimi için:

- Required (gerekli): mesajda belirtilen bu alandan birine sahip olmalıdır.
- Optional (isteğe bağl)ı: mesajda bu alandan sıfır veya bir tane olabilir (ancak birden fazla olamaz).
- Repeated (tekrarlanan): bu alan, mesajda herhangi bir sayıda (sıfır dahil) tekrarlanabilir. Tekrarlanan değerlerin sırası korunacaktır.

Tablo 4.8, Chunk mesajı için olası bir tanımı göstermektedir. Kamera bilgisi, video parçası, Hız hesaplama için başlangıç, bitiş ve mesafe bilgisi ve sınıflandırıcı bilgisi hakkında alanlar içerir. Protokol Arabelleği, geliştiricinin alan sırasını ve türleri C-benzeri tarzda tanımlamasını sağlar. Ayrıca her alan için varsayılan değerler atamak mümkündür.

Tablo 4.8. Protobuf Yapısı

Chunk protobuf serileştirmesi
message Chunk { int32 kamera_id;

Tablo 4.8. (Devam) Protobuf Yapısı

```
    int32 chunk_id ;
    int32 hiz_baslangic_noktası ;
    int32 hiz_bitis_noktası;
    int32 hiz_hesaplama_mesafe;
    int32 publisher_type;
    repeated string car_ids ;
    repeated int32 siniflandirici1;
    repeated int32 siniflandirici2;
    repeated int32 siniflandirici3;
    message Chunk_Frame {
       uint64 frame_id ;
       repeated int32 obj_frame_id;
       repeated int32 obj_x;
       repeated int32 obj_y;
       repeated int32 obj_genislik;
       repeated int32 obj_yukseklik; }
 }
```

4.2.6 Güvenlik

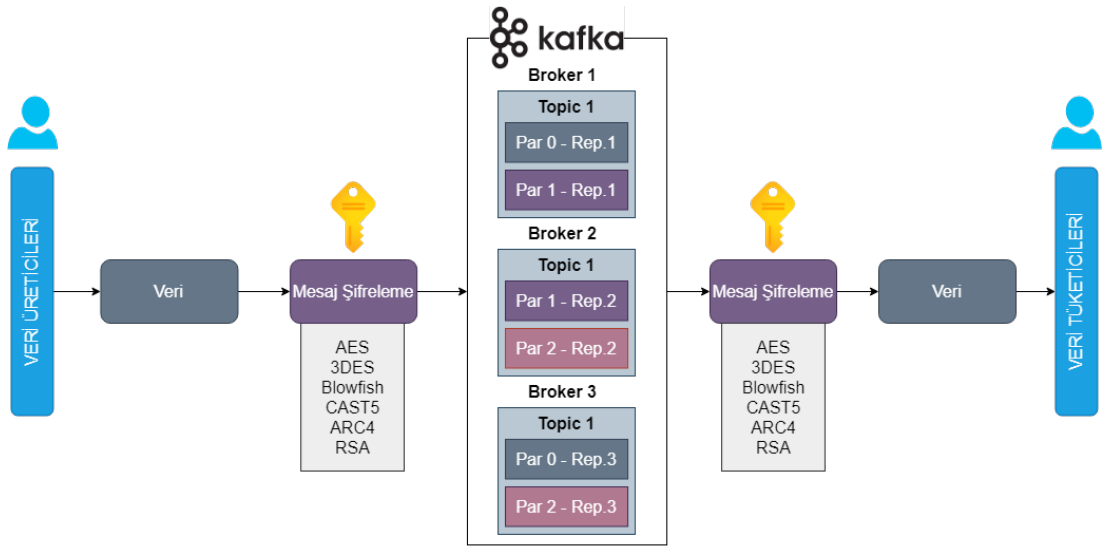
Günümüz dünyasında kriptografi bilgi güvenliğinin hayati bir parçasıdır (Feng ve diğ., 1976). Kriptografi, özellikle iletim ve depolama için dijital verileri istenmeyen erişimlere karşı anlaşılmaz hale getirme bilimidir. Kriptografinin temel bir işlevi olan şifreleme/şifre çözme, bilgiyi gizli tutmanın yaygın olarak kullanılan bir yoludur. Temel iletişim (düz metin), şifreleme sırasında şifreli metin olarak bilinen okunamaz bir forma dönüşür. Şifre çözme (düz metin) sırasında bir şifreli metin orijinal metne dönüştürülür.

İki tür kriptografi vardır: simetrik ve asimetrik (Mikhail ve diğ., 2014). Simetrik anahtar şifrelemesi, verileri yalnızca gönderici ve alıcının bildiği gizli bir anahtar kullanarak şifreleyerek korur. Asimetrik anahtar şifreleme ise verileri korumak için genel ve özel anahtarları kullanır. Özel anahtar bir kişi tarafından tutulur ve başka kimseyle paylaşılmaz. Açık anahtar ise yaygın olarak bilinir.

Kriptografi tekniklerinin hesaplama süresi üç kategoriye ayrılır: şifreleme/şifre çözme, anahtar oluşturma ve anahtar değişimi. Düz metni (mesaj) şifreli metne dönüştürmek ve bunun tersi, şifreleme/şifre çözme süresini hesaplar. Bir anahtarın üretilmesi için gereken süre, simetrik ve asimetrik kriptografi arasında farklılık gösteren anahtarın uzunluğuna göre belirlenir. Bir anahtarın üretilmesi için gereken süre, simetrik ve asimetrik kriptografi arasında farklılık gösteren anahtarın uzunluğuna göre belirlenir. Bir anahtarın değiş tokuş edilmesi için geçen süre, gönderici ve alıcı arasındaki iletişim kanalı tarafından belirlenir (Jincharadze, 2017).

Kriptografik algoritmaların seçilmesi söz konusu olduğunda, her bir algoritmanın güçlü ve zayıf yönlerinin, maliyetinin ve performansının kapsamlı bir şekilde incelenmesi faydalı bilgiler verecektir. İşlemcilerde güvenlik mekanizmalarının kullanılması aşırı yüklenmiş CPU'ları bunaltacaktır. Bu, daha yüksek güç tüketimine, uygulama gecikmelerine veya kaynak taleplerine yol açabilir. Bu nedenle, Apache Kafka iletişimi için AES, 3DES, RSA ve Blowfish gibi çeşitli simetrik ve asimetrik şifreleme/şifre çözme şemalarının etkilerini değerlendirilmiştir.

Bu bölüm, kompaktlık ve performansa dayalı kriptografik algoritmaları değerlendirmek için çeşitli kriterler sunmaktadır. Algoritmaların performansları kriptografik güç, sistem maliyeti ve tepki süresi açısından değerlendirildi. Mesaj bit sayısı ve şifre anahtarı bit sayısına bağlı olarak yöntemin ne kadar süreceği araştırıldı.



Şekil 4.26. Güvenli Yayınla/abone ol adımları

Şekil 4.26'da gösterildiği gibi, sistem çeşitli uzunluklarda mesajlar üretir ve bunlar daha sonra yayınlanmadan önce 3DES, AES, Blowfish, CAST5, RC4 ve RSA algoritmalarından biri kullanılarak şifrelenir. Şifrelenmiş metinler daha sonra Apache Kafka çerçevesi kullanılarak yayınlanır.

Simetrik anahtar algoritmaları, aynı şifreleme anahtarlarını kullanarak düz metni şifreler ve şifreli metnin şifresini çözer. Anahtarlar arasında fark olmayabilir veya aralarında basit bir geçiş olabilir. Bu çalışmada kullanılan simetrik kriptografi teknikleri aşağıda listelenmiştir.

- DES ve Üçlü DES, DES, verileri 64 bitlik bloklar halinde şifreleyen bir blok şifredir. Bu, 64 bit düz metnin DES'e beslendiği ve 64 bit şifreli metnin üretildiği anlamına gelir. Şifreleme ve şifre çözme, küçük farklılıklarla aynı algoritmayı ve anahtarı kullanır. 56 bitlik bir anahtar kullanılır.
- 3DES algoritması, Veri Şifreleme Standardı (DES) şifresiyle verileri üç kez şifreler. 3DES'i kullanmadan önce, kullanıcıların öncelikle üç DES anahtarından oluşan bir 3DES anahtarı K üretmesi ve paylaşması gerekir: K1, K2 ve K3. Bu, gerçek 3DES anahtarının 3x56 = 168 bit uzunluğunda olduğunu gösterir. Üçlü DES sistemleri, tekli DES'lerden çok daha güvenlidir, ancak oldukça yavaştırlar.
- Gelişmiş Şifreleme Standardı (AES) simetrik bir blok şifrelemedir. AES'nin tüm hesaplamaları bit yerine bayt cinsinden yapılır. Sonuç olarak, AES bir düz metin bloğunun 128 bitini 16 bayt olarak yorumlar. Matris işleme için bu 16 bayt, dört sütun ve dört satır halinde düzenlenir. AES-128, 128 bit anahtar uzunluğuna sahip bir mesaj bloğunu şifreler ve şifresini çözerken, AES-192, 192 bit anahtar uzunluğuna ve AES-256'ya 256 bit anahtar uzunluğuna sahip mesajları şifreler ve şifresini çözer. Bu çalışmada 256 bit anahtar uzunluğu kullanılmıştır.
- Blowfish, bir drop-in yedeği olarak DES yerine kullanılabilecek simetrik bir blok şifredir. 32 ila 448 bit arasında değişen değişken uzunluklu bir anahtar kullanır, bu da onu hem yerel hem de uluslararası kullanım için uygun hale getirir.
- CAST5, CAST-128 olarak da bilinen simetrik bir blok şifredir. 128 bite kadar yapılandırılabilir anahtar boyutu ve 8 bayt blok boyutu.
- En yaygın kullanılan akış şifrelerinden biri RC4 şifrelemesidir. 64 bit veya 128 bit anahtarlarla çalışır.

Simetrik şifrelemeden farklı olarak, asimetrik şifreleme iki anahtar kullanır: bir genel ve bir özel. Kriptografik sistemdeki herkesin ortak anahtara erişimi vardır, ancak yetkili kullanıcı özel anahtarı gizli tutar.

1977'de RSA algoritmasını icat eden Rivest, Shamir ve Adleman, hem mesaj şifreleme hem de şifre çözme için kullanılan asimetrik bir şifreleme yöntemi olarak bilinir. Anahtarları güvenli olmayan bir kanal üzerinden iletirken, RSA yaygın olarak kullanılır. Yöntem, asimetrik doğası gereği iki anahtar kullanır. Şifreleme sistemindeki herkesin açık anahtara erişimi vardır, ancak özel anahtar yetkili bir kişi tarafından gizli tutulur.

5. SONUÇLAR VE GELECEK ÇALIŞMALAR

Bu tez çalışmasında sunulan mimarinin deneysel sonuçları kullanılan veri setlerine bağlı olup, doğruluğu bu veri setleri üzerinde denenmiştir. Araç haricinde farklı nesneler ve nesnelere ait sorgulama sisteminin kullanılması durumunda, yapay zeka modelleri istenilen kategoriye göre çevrim dışı olarak eğitilmeli ve mikroservis mimarisine yüklenmelidir. Buna bağlı olarak da yayınlanan konular değiştirilmelidir.

5.1. Deneysel Sonuçlar

Bu bölümde, araç tip, renk sınıflandırma ve hız ölçümü ile ilgili yapılan deneysel sonuçlar, Apache Kafka'nın parametrelere bağlı performans analizleri ve veri şifrelemenin yayınla/abone ol mimarisine etkileri verilmiş ve sonuçları analiz edilmiştir.

Deneylerimiz Intel® Core™ i9-9900KF CPU @ 3.60GHz × 16, GeForce RTX 2080 Ti/PCIe/SSE2, Ubuntu 18.04 OS ve Dört çekirdekli ARM Cortex-A72 CPU, Broadcom 2711 64-bit SoC ve 8GB LPDDR4 RAM özelliklerine ait Raspberry Pi 4 (68.63 x 94.09 x 26.63 mm) ve 1.5 GHz hızında bir makinede yapıldı. Programlama dili olarak C ++ kullanıldı.

Deneysel sonuçlar, Giriş bölümünde sırasıyla verilen problemlere karşılık önerilen çözümlerin performasını göstermektedir.

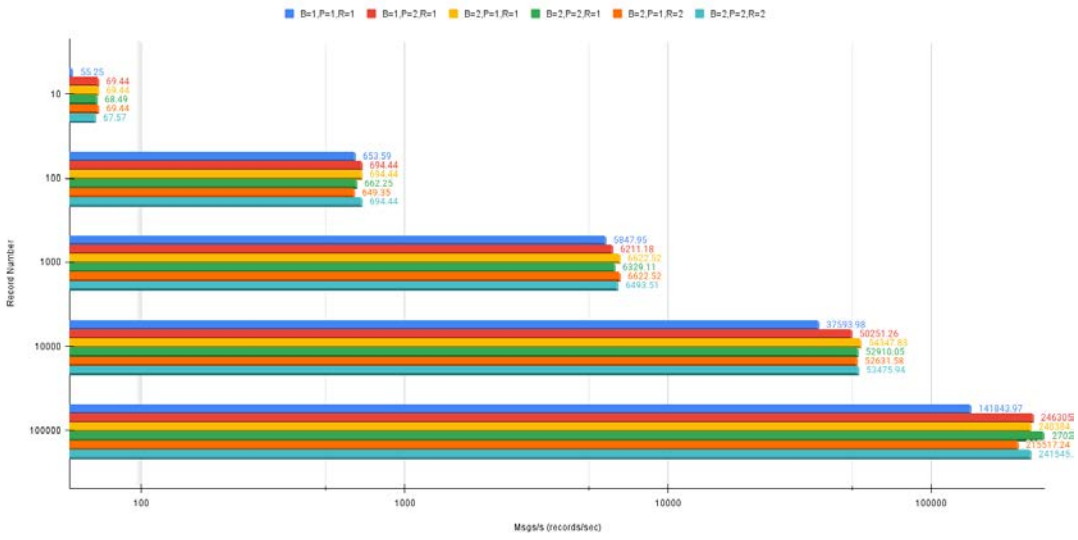
Problem 1'de belirtildiği üzere, yayınla/abone ol haberleşme mimarisini analiz etmek üzere, Kafka Cluster mimarisi içerisinde broker, zookeeper, replication ve partition parametrelerinin üretici (producer) ve tüketici (consumer) üzerindeki performansları incelendi. Toplamda 5 kombinasyon oluşturuldu. Bu kombinasyonlar local ve docker üzerinde olmak üzere iki farklı ortamda denendi. Her bir senaryo için üretici sırasıyla 10, 100, 1000, 10000 ve 100000 adet mesaj gönderdi. Aynı şekilde tüketici de 10, 100, 1000, 10000 ve 100000 adet mesaj aldı. Oluşturulan senaryolar Tablo 5.1'de verilmiştir.

Bu testler üreticinin verimini vurgulayacaktır. Bu testler sırasında hiçbir tüketici çalıştırılmaz, bu nedenle tüm mesajlar kalıcıdır, ancak okunmaz. Şekil 5.1. verilen grafikler, üretici çıktısının aracıların sayısına (broker), konu bölümlerine (partition) ve

replikasyonlara (replication) bağlı olarak değiştiğini göstermektedir. Partition 2, Broker 2 ve Replication 1 olduğunda verim sonuçlarının daha iyi olduğu gözlemlenmiştir.

Tablo 5.1. Kafka parametreleri ile oluşturulan kombinasyonlar

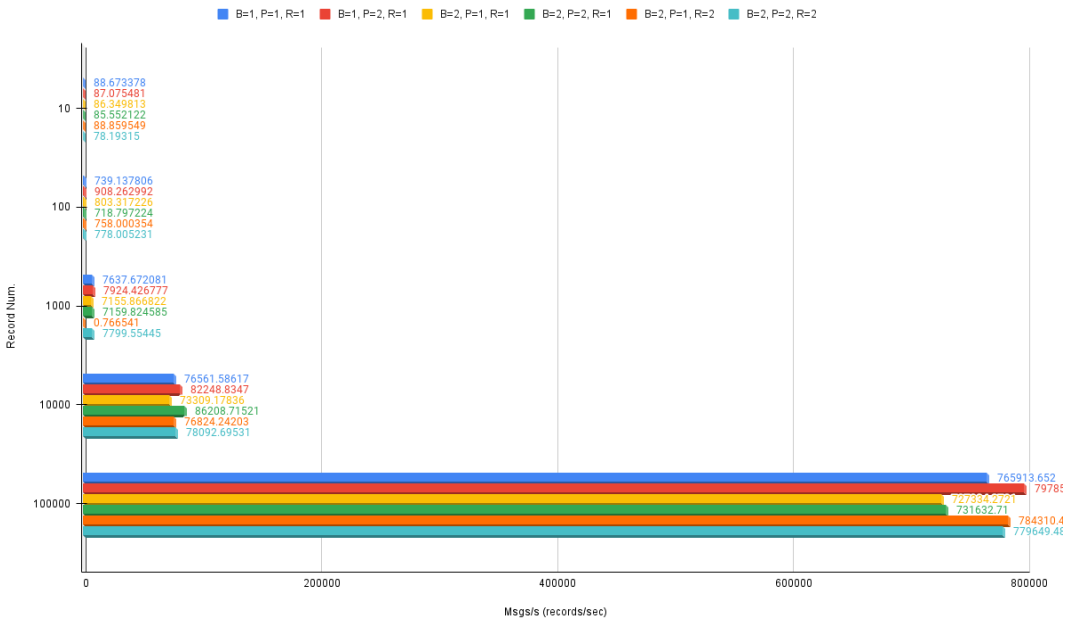
Broker Sayısı	Zookeeper Sayısı	Partition Sayısı	Replikasyon Sayısı	Adlandırma
1	1	1	1	B1Z1P1R1
1	1	2	1	B1Z1P2R1
2	1	1	1	B2Z1P1R1
2	1	2	1	B2Z1P2R1
2	1	1	2	B2Z1P1R2
2	1	2	2	B2Z1P2R2
2	2	1	1	B2Z2P1R1
2	2	2	1	B2Z2P2R1
2	2	1	2	B2Z2P1R2
2	2	2	2	B2Z2P2R2



Şekil 5.1. Üretici verimliliği (mb/sn - mesaj sayısı) grafiği

Tüketici çoğaltma (replication) faktörüne bakmaksızın yalnızca bir kopyadan okuduğundan çoğaltma faktörünün bu testin sonucunu etkilemeyecektir. Şekil 5.2'de

gösterildiği üzere sonuçlar B1P2R1 kombinasyonun nispeter tüketicide daha iyi bir performans verdiğini göstermektedir.



Şekil 5.2. Tüketici verimliliği (mb/sn- mesaj sayısı) grafiği

Problem 2'de verilen çözüm için, araç tipi tespitinde, Test veri kümesindeki modelin doğruluğunu ölçmek için kullanılan Union (IOU), Ortalama Hassasiyet, Geri Çağırma ve F-Skoru metrikleri üzerinden kesişme. Tablo 5.2, YOLOv4, YOLOv4-Tiny için test sonucumuzu gösterir ve bunları Sang tarafından önerilen model sonuçlarıyla karşılaştırır. YOLOv4-Tiny ile %98,39'luk daha yüksek AP elde ettik. Modelimiz BIT-Vehicle Dataset'in tüm araç sınıfında daha iyi performans göstermektedir. Tablo 4.3, her çerçeve için Yolov4, Yolov4-Tiny Modeli için Yürütme Süresini göstermektedir.

YOLOV4 ile %98 Hatırlama, %96 F1 puanı ve %86 Ortalama IoU puanları elde ettik. Tiny-YOLOV4 ile %98 Geri Çağırma, %96 F1 puanı ve %83.01 Ortalama IoU puanları elde edildi.

Problem 3'te verilen çözüm için, araç renk tespitinde ise Renk modelinin doğruluğunu ölçmek ve performansı karşılaştırmak için RGB ve HSV olmak üzere iki farklı renk alanı kullandık. Veri setimizi %80-20 oranlarına göre eğitim ve test olarak ayırdık. Giriş şeklimiz 96x96x3 çözünürlüğe yeniden boyutlandırıldı.

Tablo 5.2. YOLOV4, YOLOV4-Tiny PERFORMANS ve BIT-Araç veri kümesi üzerinde farklı yaklaşımların karşılaştırılması

Araç Sınıfı	YOLOV 4	Tiny YOLOV4	YOLOv2 Araç (Sang ve diğ., 2018)	Model Comp (Sang ve diğ., 2018)	Faster R-CNN + ResNet (Girshick ve diğ., 2014)
Otobüs	97,80%	97,80%	97,54%	97,43%	90,62%
Mikrobüs	96,43%	98,33%	93,76%	94,47%	94,42%
Minivan	95,06%	98,18%	92,18%	90,86%	90,67%
Sedan	99,12%	99,19%	98,48%	97,46%	90,63%
SUV	96,57%	97,04%	94,62%	93,05%	91,25%
Kamyon	97,23%	99,81%	92,09%	91,69%	90,07%
Ortalama	97,04%	98,39%	94,78%	94,16%	91,28%

Tablo 5.3. Yolov4, Yolov4-Tiny Model Her Çerçeve İçin Yürütme Süresi

	YOLOV4	Tiny YOLOV4
Ort. zaman (ms)	263,29	87,58

YOLOV4 ve Tiny-YOLOV4 eğitiminin performans sonuçları 10000 epoch tamamlandı.

Tablo 5.4, RGB ve HSV renk uzayları ile test sonucumuzu gösterir ve bunları Chen ve diğerleri (Chen ve diğ. 2014), tarafından önerilen model sonuçlarıyla karşılaştırır. Rachmadi (Rachmadi ve diğ., 2015). HSV renk alanı ile %0,9563 gibi daha yüksek bir ortalama doğruluk elde ettik. Modelimiz Mavi, Cyan, Yeşil, Kırmızı ve Sarı renklerde daha iyi performans göstermektedir.

Tablo 5.5, her çerçeve için RGB ConNN Modeli ve HSV ConNN modeli için yürütme süresini gösterir.

Problem 4'te verilen çözüm için, araç hız tespitinde sistemimiz, takip yöntemine göre hızları hesaplar. Veri seti ile sistemimizin varsayımı, çizgiler arasındaki bilinen mesafe ve fps değerleridir. Araç takibi bilinen bir ortamda ve referans aralığında gerçekleştirilir. Bu referans noktaları 4,8 metre genişliğinde ve 2,0 metre yüksekliğindedir. Aracı

çevreleyen sınırlayıcı kutu ile aracın konumu izlenir. Sınırlama kutusu referans alanına girdiği andan itibaren süre tutulur ve araç alandan ayrıldığında ortalama hız hesaplanır.

Tablo 5.4. Araç Renk Tespiti ConNN Mimari

Araç Renk Sınıfı	Model Doğruluğu RGB	Model Doğruluğu HSV	Chen Model Doğruluğu (Chen ve diğ. 2014).	Rachmadi Model Doğruluğu (Rachmadi ve diğ., 2015)
Siyah	0,9723	0,9560	0,9730	0,9738
Mavi	0,9513	0,9579	0,9535	0,9410
Camgöbeği	0,9973	0,9986	0,9787	0,9645
Gri	0,8371	0,8555	0,8466	0,8608
Yeşil	0,9468	0,9544	0,7884	0,8257
Kırmızı	0,9884	0,9923	0,9878	0,9897
Beyaz	0,9547	0,9453	0,9423	0,9666
Sarı	0,9752	0,9905	0,9553	0,9794
Ortalama	0,9529	0,9563	0,9282	0,9447

RGB ile 0.9534 f1-skoru, 0.9554 kesinlik ve 0.9529 geri çağırma puanları, HSV renk uzayları ile 0.9563 f1, 0.9575 kesinlik ve 0.9568 geri çağırma puanları elde ettik.

Tablo 5.5. Her Çerçeve İçin Yürütme Süresi

	RGB ConNN Model	HSV ConNN Model
Ort. zaman (ms)	24,5	25,65

Hesaplanan hız değerleri zemin gerçeği ile karşılaştırılarak hız ölçüm performansı değerlendirilmiştir. ABD standartlarına göre, bir ölçüm eksi 3 km/sa, +artı 2 km/sa hata aralığında olmalıdır.

Sonuçlar Tablo 5.6'da verilmiştir.

Tablo 5.6. Hız Tespiti

Gerçek Hız (km/sa)	Hesaplanan Hız (km/sa)
56,65	59,2
53,74	55

Tablo 5.6. (Devam) Hız Tespiti

52,13	56,23
49,06	51,47
50,51	53,6
48,89	50,03
59,57	62,37
48,83	50,81
56,01	59,65
51,12	55,2
41,97	42,67

Problem 5 ve 6 için verilen çözüm, Yöntem başlığı altında açıklanmıştır.

Problem 7, haberleşme kanalını güvenli yapmada ise yayınlama/abone olma sistemlerinde simetrik ve asimetrik algoritmaların performansını karşılaştırmak için şifreleme zamanı, şifre çözme zamanı ve yayıncıların üretme zamanı gibi özellikleri değerlendiririz Simetrik algoritmalar 3DES, AES, Blow-fish, CAST5 ve RC4'ü içerirken asimetrik algoritmalar RSA'yı içerir. Testlerimizi 5 kez çalıştırdık ve sonuçların ortalamasını aldık. Tablo 1 sonucu göstermektedir. Çalışmalarımız için değerlendirme metriklerimiz aşağıdaki gibidir:

Şifreleme Süresi: Şifreleme süresi, bir şifreleme tekniğinin düz metinden bir şifreli metin oluşturması için geçen süredir. Tablo 5.7, farklı anahtar ve veri boyutlarında 3DES, AES, Blowfish, CAST5, RC4 ve RSA'nın şifreleme süresini göstermektedir.

Şifre Çözme Süresi: Şifre çözme süresi, bir şifreleme tekniğinin bir şifreli metinden bir düz metin üretmesi için geçen süredir. Üretme Süresi: Bir Apache Kafka yayıncısının bir şifreli metin yayınlaması için geçen süre, üretim süresi olarak bilinir.

AES, 3DES, Blowfish, CAST5 ve RC4 algoritmalarının çeşitli dosya boyutlarında şifreleme ve şifre çözme süreleri Tablo 5.7'de gösterilmiştir. Tablo 4.7, AES algoritmasının şifreleme ve şifre çözme süresinin DES algoritmasına göre daha düşük olduğunu göstermektedir.

Tablo 5.7. Anahtar-Veri Bit Boyutlarına Dayalı Simetrik Şifreleme Algoritmaları için Şifreleme, Şifre Çözme ve Yayınlama Süresi

Kriptografi Algoritmaları	Anahtar Boyutu (bits)	Veri Boyutu (bits)	Şifreleme Süresi (sn)	Şifre Çözme Süresi (sn)	Verilerin Yayınlanma süresi (sn)
AES	256	1048576	0,003093	0,00317	0,028336
		2097152	0,006143	0,006188	0,041272
		4194304	0,012455	0,012502	0,634794
		8388608	0,024557	0,024972	0,046605
3DES	64	1048576	0,012819	0,013016	0,036797
		2097152	0,025869	0,025876	0,070912
		4194304	0,054368	0,054006	0,633884
		8388608	0,108439	0,108207	0,217624
3DES	128	1048576	0,012809	0,013035	0,095825
		2097152	0,025848	0,02587	0,187665
		4194304	0,054381	0,053003	0,378478
		8388608	0,108435	0,115457	0,21152
3DES	192	1048576	0,012814	0,013182	0,09534
		2097152	0,025855	0,026008	0,186994
		4194304	0,054402	0,053995	0,376833
		8388608	0,10865	0,108265	0,216971
Blowfish	256	1048576	0,002961	0,003185	0,045933
		2097152	0,006131	0,006195	0,089047
		4194304	0,015044	0,01389	0,491844
		8388608	0,029762	0,029944	0,070624
CAST5	128	1048576	0,003072	0,003337	0,048027
		2097152	0,006341	0,006466	0,092652

Tablo 5.7. (Devam) Anahtar-Veri Bit Boyutlarına Dayalı Simetrik Şifreleme Algoritmaları için Şifreleme, Şifre Çözme ve Yayınlama Süresi

CAST5	128	4194304	0,015379	0,015317	0,513155
		8388608	0,030488	0,030913	0,121954
RC4	256	1048576	0,001005	0,001375	0,04726
		2097152	0,002048	0,002634	0,089339
		4194304	0,007748	0,006496	0,509371
		8388608	0,015262	0,015494	0,048748

Table 5.8. Anahtar-Veri Bit Boyutlarına Dayalı Asimetrik Şifreleme Algoritmaları için Şifreleme, Şifre Çözme ve Yayınlama Süresi

Kriptografi Algoritmaları	Anahtar Boyutu (bits)	Data Size (bits)	Şifreleme Süresi (sn)	Şifre Çözme Süresi (sn)	Verilerin Yayınlanma süresi (sn)
RSA	1024	64	0,000315	0,002605	0,056182
		128	0,000178	0,00275	0,000489
		256	0,000165	0,00274	0,000694
		512	0,000158	0,002734	0,00048
RSA	2048	64	0,000371	0,002335	0,000982
		128	0,000335	0,013883	0,000999
		256	0,000329	0,013973	0,000852
		512	0,000326	0,014146	0,002398
RSA	4096	64	0,000899	0,095471	0,001329
		128	0,000883	0,095663	0,003702
		256	0,000874	0,095751	0,002139
		512	0,00087	0,096316	0,00256

AES'ler ayrıca Blowfish ve CAST5 algoritmalarından biraz daha hızlıdır. Öte yandan, RC4 algoritmasının şifreleme ve şifre çözme süresi AES'den daha düşüktür. Üretim süresi, anahtarın ve dosyanın boyutuna bağlı olarak değişir.

Tablo 4.8, çeşitli dosya boyutları için RSA algoritmasının şifreleme ve şifre çözme sürelerini gösterir. Sonuçlar, metin dosyalarının boyutu arttıkça, onları şifrelemek, şifresini çözmek ve yayınlamak için geçen sürenin de arttığını ortaya koymaktadır.

Literatürde derin öğrenme yöntemleri kullanılarak veri akışı işleme ile ilgili çeşitli çalışmalar bulunmaktadır. Ancak, bu çalışmaların çoğu tek bir bilgisayara odaklandı. İş birliği yapan trafik gözetleme kameraları tarafından gerçek zamanlı olarak yayınlanan video akışları üzerinden belirli bir aracı izleyen bir çerçeve öneriyoruz. Ağ arasındaki mikro hizmetler arasında veri aktardık. Ayrıca kullanıcının video akışları arasında metin sorguları kullanmasını sağlıyoruz. Bu çerçeve ile Apache Kafka'dan yararlanan kritik görev gerçek zamanlı uygulamalarda video alımı için olası kullanım örneklerini tartıştık. Kafka, makine öğrenimi altyapısı için harika bir tamamlayıcı araçtır ve analitik modelin gerçek zamanlı olarak çıkarılması, izlenmesi ve uyarılması gibi alanlarda fayda sağlayabilir.

Deneysel sonuçlar, tasarımımızın hataya karşı dayanıklı olduğunu, verilen veri setleri dahilinde akan veri üzerinde sorgulama yapılabileceğini ve bu sorguların "ve", "veya" şeklinde kompleks, karmaşık yapılabileceğini göstermektedir.

5.2. Gelecek Çalışmalar

Bu araştırma, akan veride sorgulama yapabilme süreçlerinde yardımcı olan video gözetleme sistemleri üzerine gelecekteki çalışmalar için genişletilebilir bir temel olarak hizmet edebilir. Devam eden bir araştırma alanı olarak, çözümün genelleştirilmesi adına zorluklarını ele almak için filtreleme alanda çözümlerin daha geniş bir şekilde uyarlanmasını sağlamak için daha fazla araştırma çabası gerektirmektedir.

Bu araştırmanın ana sınırlaması, prototipin geliştirilen modüllerinin önerilen bileşenlere dahilinde ve kullanılan veri setleri kapsamında kısıtlı işlevselliğe sahip olmasıdır. Daha fazla araştırma, önerilen çerçeveyi gerçek zamanlı sorgulama sistemlerinde yardım sağlayacak şekilde genişletebilir.

Ayrıca, gelecekteki araştırmalar, yayınla/abone ol haberleşme kanalını ele alarak önerilen çözümlerin gizlilik ve güvenlik yönlerine de odaklanmalıdır.

KAYNAKLAR

Abdelbaki, H. M., K. Hussain, E., Gelenbe, A. (2001). laser intensity image based automatic vehicle classification system, I*EEE Conf. Intelligent Transportation Systems*, Oakland, CA, USA, 25-29 Ağustos. 2001.

Afifah, F., Nasrin, S., Mukit, A. (2019). Vehicle Speed Estimation using Image Processing, *Journal of Advanced Research in Applied Mechanics*, *48*(1), 9-16.

Alaasam, A. B. A., Radchenko, G.,Tchernykh, A. Stateful Stream Processing for Digital Twins: Microservice-Based Kafka Stream DSL, *2019 International Multi-Conference on Engineering, Computer and Information* Sciences *(SIBIRCON)*, Novosibirsk, Rusya, 21-27 Ekim 2019.

Aliane N, Fernandez, J., Bemposta, S., Mata, M. (2011). Traffic violation alert and management, *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference*, Washington, DC, ABD, 5-7 Ekim 2011.

Aslam, A., Curry, E. (2021). Investigating response time and accuracy in online classifier learning for multimedia publish-subscribe systems, *Multimed Tools Appl*, *80*(1), 13021–13057.

Astley, M. (2021). Achieving scalability and throughput in a publish/subscribe system. IBM,https://dominoweb.draco.res.ibm.com/783b3bb9ffd5dfaf85256e36004fd04c.html, (Ziyaret Tarihi: 10 Temmuz 2021).

Atchison, L. (2021). Microservice Architectures: What They Are and Why You Should Use Them, https://blog.newrelic.com/technology/microservices-whatthey-are-why-to-use-them/, (Ziyaret Tarihi: 12 Mayıs 2021).

Athanesious, J.J., Suresh, P. (2012). Systematic survey on object tracking methods in video, *Int J* Adv *Res Comput Eng Technol (IJARCET), 1*(8), 242–247.

Baek, N., Park, SM., Kim, KJ., Park, SB. (2007). Vehicle Color Classification Based on the Support Vector Machine Method, *Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Technique*, Qingdao, China, 21-24 Ağustos 2007.

Banavar, G., Chandra T. B., Mukherjee, Nagarajarao J., Strom R. E., Sturman D. C. (1999). An Efficient Multicast Protocol for Content-Based Publish-Subscribe Systems, *IEEE 33rd International Conference on Distributed Computing Systems*, Austin, Texas, 5 Haziran 1999.

Bell, D., Xiao, W., James, P. (2020). Accurate Vehicle Speed Estimation from Monocular Camera Footage, ISPRS Annals of the Photogrammetry, *Remote Sensing and Spatial Information Sciences, 2*(1), 419-426.

Birman, K., Joseph, T. (1987). Exploiting virtual synchrony in distributed systems, Proceedings *of the Eleventh ACM Symposium on Operating Systems Principle,* NY, Amerika Birleşik Devletleri, Kasım 1987.

Borcea, C., Polyakov, Y., Rohloff K., Ryan G. (2017). PICADOR: End-to-end encrypted Publish–Subscribe information distribution with proxy re-encryption. F*uture Generation Computer System*s, *71*(1), 177–191.

Camargo, A., Salvadori, I., Mello R. S., Siqueira F. (2016). An architecture to automate performance tests on microservices, *the 18th International Conference on* Information *Integration and Web-based Applications and Services (iiWAS '16)*, NY, Amerika Birleşik Devletleri, 28 - 30 Kasım 2016.

Campailla, A., Chaki S., Clarke, E.M., Jha, S. (2001). Veith H., Efficient filtering in publish-subscribe systems using binary decision diagrams, *Proceedings of the 23rd International Conference on Software Engineering*, Toronto, Kanada, 12-19 Mayıs 2001.

Chakravarti, R., Meng, X. (2009). A Study of Color Histogram Based Image Retrieval, Sixth *International Conference on Information Technology: New Generations*, Las Vegas, NV, Amerika Birleşik Devletleri, 27-29 Nisan 2009.

Chan, M. N., Tint, T. (2021). A Review on Advanced Detection Methods in Vehicle Traffic Scenes, 2021 *6th International Conference on Inventive Computation* Technologies *(ICICT)*, Coimbatore, Hindistan, 20-22 Ocak 2021.

Chang, C., Liu, W., Zhang, H. (2001). Image retrieval based on region shape similarity, *Storage and Retrieval for Media Databases*, *4315*(1), 31–38.

Chang, S.K., Hsu, A. (1992). Image information systems: where do we go from here?, IEEE *Transactions on Knowledge and Data Engineering*, *4*(5), 431 - 442.

Chauhan, N., Goyani, M. (2013). Enhanced Multistage Content Based Image Retrieval, *IJCSMC*, *2*(5), 175–179.

Chen, H., Luo, F., Zhao, L., Li Y. (2017). Design and Implementation of Real-Time Video Big Data Platform based on Spark Streaming, *International Conference on Computer Science and Application Engineering (CSAE 2017)*, Shanghai, Çin, 21-23 Ekim 2017.

Chen, P., Bai, X., Liu, W. (2014). Vehicle color recognition on an UrbanRoad by feature context, *IEEE Trans. Intell. Transp. Syst*., *15*(5), 2340-2346.

Chen, Z., Ellis T., Velastin, S. A. (2011). Vehicle type categorization: A comparison of classification schemes, 1*4th IEEE Annual Conference on Intelligent Transportation Systems*, Washington, DC, Amerika Birleşik Devletleri, 5-7 Ekim 2011.

Cheng, G., Guo, Y., Cheng, X., Wang, D., Zhao J. (2020). Real-Time Detection of Vehicle Speed Based on Video Image,1*2th International Conference on Measuring Technology and Mechatronics Automation*, Phuket, Tayland, 28-29 Şubat 2020.

Cheng, H. H., Shaw, B. D., Palen, J., Lin, B., Chen, B., Wang Z. (2005). Development and field test of a laser-based nonintrusive detection system for identification of vehicles on the highway, I*EEE Transactions on Intelligent Transportation Systems*, *6*(2), 147-155.

Cheung, S. (2005). Traffic measurement and vehicle classification with single magnetic sensor, J*ournal of the Transportation Research Board*, *1917*(1), 173-181.

Chollet, F. (2021). Building Powerful Image Classification Models Using Very Little Data. Keras Blog, Keras, https://blog.keras.io/building-powerful-image-classification-modelsusing-very-little-data.html (Ziyaret Tarihi: 10 Mayıs 2021).

Chua, T.S., Tang, J., Hong, R., Li H., Luo, Z., Zheng, Y. (2009). NUS-WIDE: a real-world web image database from National University of Singapore, *Proceedings of the ACM International Conference on Image and Video Retrieval*, Santorini, Yunanistan, 8-10 Temmuz 2009.

Chuanping, H., Xiang, B., Li, Qi, Chen, P., Xue, G., Mei, L. (2015). Vehicle Color Recognition with Spatial Pyramid Deep Learning, *IEEE Transactions on Intelligent* Transportation *Systems*, *16*(5), 2925 - 2934.

Curry, E. (2004). Message-Oriented Middleware, Mahmoud Q. H. (Ed.), *Middleware for Communications* (1st ed.) (1-28). New York: CRC Press.

Dehghan, A., Masood, S.Z., Shu, G., Ortiz, E. (2021). View independent vehicle make, model and colorrecognition using convolutional neural network, Arxiv, https://arxiv.org/abs/1702.01721 (Ziyaret Tarihi: 10 Nisan 2021).

Diffie, W., Hellman, M. (1976). New directions in cryptography, *IEEE Trans. Inform.* Theory, *22*(6), 644–654.

Dong, Y., Pei, M. Qin, X. (2014). Vehicle Color Recognition Based on License Plate Color, *2014 Tenth International Conference on Computational Intelligence and* Security, Kunming, Çin, 15-16 Kasım 2014.

Dong, Z., Jia, Y. (2013). Vehicle type classification using distributions of structural and appearance-based features, *2013 IEEE International Conference on Image Processing*, Melbourne, VIC, Avustralya, 15-18 Eylül 2013.

Dong, Z., Wu, Y., Pei, M., Jia, Y. (2015). Vehicle Type Classification Using a Semisupervised Convolutional Neural Network, *IEEE Trans. Intell. Transp. Syst*., *16*(4), 2247–2256.

Duanmu, X. (2010). Image retrieval using color moment invariant, *Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations (*ITNG*)*, Las Vegas, NV, Amerika Birleşik Devletleri, 12-14 Nisan 2010.

Espinosa, J.E., Velastin, S.A., Branch, J.W. (2017). Vehicle Detection Using Alex Net and Faster RCNN Deep Learning Models: A Comparative Study, *Advances in Visual Informatics*, Bangi, Malezya, 28-30 Kasım 2017.

Eugster, P. T., Pascal, A. F., Guerraoui, R., Kermarrec, A.M. (2003). The Many Faces of Publish/Subscribe, *ACM Computing Surveys, 35*(2), 114-131.

Eugster, P.T., Guerraoui, R., Sventek, J. (2000). Distributed Asynchronous Collections: Abstractions for Publish/Subscribe Interaction, *European Conference on* Object-*Oriented Programming*, Cannes, Fransa, 12-16 Haziran 2000.

Fabret, F., Jacobsen, H. A., Llirbat, F., Pereira J., Ross, K. A., Shasha, D. (2001). Filtering algorithms and implementation for very fast publish/subscribe systems, *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, Kaliforniya, ABD, 21 - 24 Mayıs 2001.

Fang, J., Meng, H., Zhang, H., Wang, X. (2007). A Low-cost Vehicle Detection and Classification System based on Unmodulated Continuous-wave Radar, *IEEE* Intelligent *Transportation Systems Conference*, Bellevue, WA, USA, 30 Eylül - 3 Ekim 2007.

Fang, J., Yue, H., Li, X., Wu, J. (2011). Cui, Z., Color identifying of vehicles based on color container and BP network, *International Conference on Business Management and Electronic Information (BMEI)*, Guangzhou, 13-15 Mayıs 2011.

Fergus, R., Fei-Fei, L., Perona, P., Zisserman, A. (2005). Learning object categories from Google's image search, *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Pekin, Çin, 17-21 Ekim 2005.

Fiege, L., Zeidler A., Buchmann A., Kilian-Kehr R., Muhl G., Darmstadt T. (2004). Security aspects in publish/subscribe systems, *In Proceedings of the 3rd International* Workshop *on Distributed Event-Based Systems (DEBS'04)*, Edinburgh, Birleşik Krallık, 28 Mayıs 2004.

Gadea, C., Trifan, M., Ionescu, D., Cordea, M., Ionescu, B. (2016). A microservices architecture for collaborative document editing enhanced with face recognition, *11th* International *Symposium on Applied Computational Intelligence and Informatics (SACI)*, Timisoara, Romanya, 12-14 Mayıs 2016.

Gaikwad. R., Neve J. R. (2016). A comprehensive study in novel content based video retrieval using vector quantization over a diversity of color spaces, *International* Conference *on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, Jalgaon, Hindistan, 22-24 Aralık 2016.

Gajda, J., Sroka, R., Stencel, M., Wajda, A., Zeglen, T. (2001). A vehicle classification based on inductive loop detectors, *18th IEEE Instrumentation and Measurement Technology Conference*, Budapeşte, Macaristan, 21-23 Mayıs 2001.

Gao, J., Liu, J., Rajan, B., Nori, R., Fu, B., Xiao, Y., Liang, W., Chen, C. L. P. (2014). SCADA communication and security issues, *Security and Communication Networks*, *7*(1), 1-7.

Gaszczaka, A., Breckon, T., Hana, J. (2013). Real-time people and vehicle detection from UAV imagery, *Intelligent Robots and Computer Vision XXX: Algorithms and Techniques*, California, Amerika Birleşik Devletleri, 4 - 6 Şubat 2013.

Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, *IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, ABD, 23-28 Haziran 2014.

Guo, J. M., Prasetyo, H., Chen, J. H. (2015). Content-based image retrieval using error diffusion block truncation coding features, *IEEE Transactions on Circuits and Systems for Video Technology*, *25*(3), 466–481.

Guo, X., Zhang, Q., Yang, L., Dong, Q. (2005). Extraction of Complex Background with Moving Objects, *Automation Panorama*, *22*(6): 70-71.

Gupte, S., Masoud, O., Martin, R. F. K. (2002). Papanikolopoulos N. P., Detection and classification of vehicles, *IEEE Trans. Intelligent Transportation Systems*, *3*(1), 37-47.

Hadi, R. A., Sulong, G., George, L. E. (2014). Vehicle detection and tracking techniques: A concise review, *Signal Image Process.*, *5*(1), 1-12.

Hassan, S., Bahsoon, R. (2016). Microservices and their design trade-offs: A self-adaptive roadmap, *2016 IEEE International Conference on Services Computing (*SCC*)*, San Francisco, CA, ABD, 27 Haziran-2 Temmuz 2016.

Hasselbring, W. (2016). Microservices for scalability: Keynote talk abstract, Proceedings *of the 7th ACM/SPEC on International Conference on Performance Engineering*, Delft, Hollanda, 12 - 16 Mart 2016.

Hedeya, M. A., Eid A. H., Abdel-Kader R. F. (2020). A Super-Learner Ensemble of Deep Networks for Vehicle-Type Classification, *IEEE Access*, *8*(1), 98266-98280.

Heinrich, R., Hoorn, A. V., Knoche, H., Li, F., Lwakatare, L.E., Pahl, C., Schulte, S., Wettinger, J. (2017). Performance engineering for microservices: Research challenges and directions, *8th ACM/SPEC International Conference on Performance Engineering (ICPE)*, L'Aquila, İtalya, 22-26 Nisan 2017.

Hirwane, T. (2017). Semantic based Image Retrieval, *International Journal of Advanced Research in Computer and Communication Engineering ISO*, *6*(4), 3297:2007.

Hollink, L., Little, S., Hunter, J. (2005). Evaluating the application of semantic inferencing rules to image annotation, *K-CAP '05: Proceedings of the 3rd international conference on Knowledge capture*, Banff Alberta, Kanada, 2 - 5 Ekim 2005.

Hoque, S. (2018). Miranskyy A., Architecture for analysis of streaming data, *2018 IEEE International Conference on Cloud Engineering (IC2E)*, Orlando, Florida, ABD, 17-20 Nisan 2018.

Hsieh, J.W., Chen, L.C., Chen, S.Y., Chen, D.Y., Alghyaline S., Chiang H.F. (2015). Vehicle Color Classification Under Different Lighting Conditions Through Color Correction, *IEEE Sensors Journal*, *15*(2), 971–983.

Huang, J., Kumar, S. R., Mitra, M., Zhu, W.J., Zabih R. (1997). Image indexing using color correlograms, *IEEE Comput. Vis. Pattern Recog*., San Juan, Porto Riko, ABD, 17-19 Haziran 1997.

Huang, Y., Garcia-Molina, H. (2001). Publish/subscribe in a mobile enviroment, *Wireless Networks*, *10*(6), 27-34.

Huo, Z., Xia, Y., Zhang, B. (2016). Vehicle type classification and attribute prediction using multi-task RCNN, 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Datong, Çin, 15-17 Ekim 2016.

Ioana, A., Burlacu, C., Korodi, A. (2021). Approaching OPC UA Publish–Subscribe in the Context of UDP-Based Multi-Channel Communication and Image Transmission, *Sensors*, *21*(4), 1296-1315.

Ion, M., Russello, G., Crispo, B. (2010). An implementation of event and filter confidentiality in pub/sub systems and its application to e-health, *In Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS'10)*, Chicago Illinois, ABD, 4 - 8 Ekim 2010.

Ion, M., Russello, G., Crispo, B. (2010). Supporting publication and subscription confidentiality in pub/sub networks, *In Security and Privacy in Communication Networks. Lecture Notes in Computer Science*, Singapur, Singapur, 7-9 Eylül 2010.

Jincharadze E. (2017). Critical analysis of some cryptography algorithms, *Scientific and Practical Cyber Security Journal*, *1*(2), 60-68

Karthikram (2014). Tag Based Image Retrieval (Tbir) Using Automatic Image Annotation, *International Journal Of Research In Engineering And Technology*, *3*(1), 148-152.

Kato, K., Takefusa, A., Nakada, H., Oguchi, M. (2019). Construction Scheme of a Scalable Distributed Stream Processing Infrastructure, *34th International Conference on Computers and Their Applications (CATA 2019)*, Hawaii, ABD, 18-20 Mart 2019.

Kato, T. (1992). Database architecture for content-based image retrieval, *Symposium on Electronic Imaging: Science and Technology*, San Jose, CA, ABD, 1 April 1992.

Khurana, H. (2005). Scalable security and accounting services for contentbased publish/subscribe systems, *2005 ACM symposium on Applied computing*, New Mexico, ABD, 13-17 Mart 2005.

Ki, Y.K., Baik, D. K. (2006). Model for Accurate Speed Measurement Using Double-Loop Detectors, *The IEEE transactions on Vehicular Technology*, *55*(4), 1094-1101.

Kim, K. (2018). Vehicle Color Recognition via Representative Color Region Extraction and Convolutional Neural Network,*2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, Prag, Çek Cumhuriyeti, 3-6 Temmuz 2018.

Kim, Y.K., Kim, Y., Jeong, C.S. (2018). RIDE: real-time massive image processing platform on distributed environment, *J Image Video Proc*., *39*(2018), 1-13.

Kim, Y. K., Kim, Y. (2020). DiPLIP: Distributed Parallel Processing Platform for Stream Image Processing Based on Deep Learning Model Inference, *Electronics*, *9*(10), 1664-1681.

Kleyko, D., Hostettler, R., Birk, W., Osipov E. (2015) Comparison of Machine Learning Techniques for Vehicle Classification Using Road Side Sensors, *ITSC '15: Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems*, DC, Amerika Birleşik Devletleri, 15 -18 Eylül 2015.

Klock, S., Werf, J.M.E.M. van der, Guelen J.P, Jansen S. (2017). Workload-based clustering of coherent feature sets in microservice architectures, in Proc. *IEEE Int. Conf. Softw. Archit*., Göteborg, İsveç, 3-7 Nisan 2017.

Koide, R., Kitamura, S., Nagase, T., Araki, T., Araki, M., Ono, H. (2006). A Punctilious Detection Method for Measuring Vehicles' Speeds, *The International Symposium on Intelligent Signal Processing and Communication System (ISPACS 2006)*, Yonago, Japonya, 12-15 Aralık 2006.

Koyuncu, H., Koyuncu, B. (2018). Vehicle Speed detection by using Camera and image processing software, *The International Journal of Engineering and Science*, *7*(9), 64-72.

Kul, S., Eken, S., Sayar, A. (2017). Distributed and collaborative realtime vehicle detection and classification over the video streams, *Int. J. Adv. Robot. Syst.*, *14*(4), 1-12.

Kul, S., Eken, S., Sayar, A. (2016). Measuring the Efficiencies of Vehicle Classification Algorithms on Traffic Surveillance Video, *International Conference on Artificial Intelligence and Data Processing*, Malatya, Türkiye, 17 - 18 Eylül 2016.

Kul, S., Tashiev, I., Şentaş, A., Sayar, A. (2021). Event-Based Microservices With Apache Kafka Streams: A Real-Time Vehicle Detection System Based on Type, Color, and Speed Attributes, *IEEE Access*, *9*(1), 83137-83148.

Kurtz, C., Depeursinge, A., Napel, A., Beaulieu, C. F., Rubin, D. L. (2014). On combining image-based and ontological semantic dissimilarities for medical image retrieval applications, *Medical Image Analysis*, *18*(7), 1082- 1100.

Lai, A.H.S., Fung, G.S.K., Yung, N.H.C. (2001). Vehicle Type Classification from Visual-Based Dimension Estimation, *Intelligent Transportation Systems*, Oakland, CA, ABD, 25-29 Ağustos. 2001

Lauzon, F. Q. (2012). An Introduction to Deep Learning, *11th International Conference on Information Science, Signal Processing and their Applications*, Montreal, QC, Kanada, 2-5 Temmuz 2012.

Lee, J.T., Chung, Y. (2017). Deep learning-based vehicle classifcation using an ensemble of local expert and global networks, *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, HI, ABD, 21-26 Temmuz 2017.

Leitner, P., Cito, J., Stockli, E. (2016). Modelling and managing € deployment costs of microservice-based cloud applications, *UCC '16: Proceedings of the 9th International Conference on Utility and Cloud Computin*g, Şanghay, Çin, 6 - 9 Aralık 2016.

Lewis, J., Fowler M. (2021). Microservices a definition of this new architectural term, Martin Flower, http://martinfowler. com/articles/microservices.html, (Ziyaret Tarihi: 10 Mayıs 2021).

Li, W., Duan, L., Xu, D., Tsang, I.W.H. (2011). Text-based image retrieval using progressive multi-instance learning, *2011 International Conference on Computer Vision*, Barselona, İspanya, 6-13 Kasım 2011.

Li, X., Zhang, G., Fang, J., Wu, J., Cui, Z. (2010). Vehicle color recognition using vector matching of template, *In Third International Symposium on Electronic Commerce and Security*, Guangzhou, Çin, 29-31 Temmuz 2010.

Lin, H. Y., Li, K.J. (2004). Motion Blur Removal and Its Application to Vehicle Speed Detection, *The IEEE International Conference on Image Processing (ICIP 2004)*, Singapur, 24-27 Ekim 2004.

Liu, F., Yarom, Y., Ge, Q., Heiser, G., Lee, R. B. (2015). Last-Level Cache Side-Channel Attacks are Practical, *2015 IEEE Symposium on Security and Privacy*, San Jose, CA, ABD, 17-21 Mayıs 2015.

Liu, T., Chi, H., Hong, C., Meng-shou, Z. (2014). Moving object detection in dynamic background, *Proceedings of the 33rd Chinese Control Conference*, Nanjing, Çin, 28-30 Temmuz 2014.

Luvizon, D. C., Nassu, B. T., Minetto, R. (2016). A Video-Based System for Vehicle Speed Measurement in Urban Roadways, *in IEEE Transactions on Intelligent Transportation Systems*, *18***(**6) 1393-1404.

Machanavajjhala A., Vee E., Garofalakis M., Shanmugasundaram J., *Proceedings of the VLDB Endowment*, 2008, **1**(1), 451–462.

Mahto P., Garg P., Seth P., Panda J. (2020). Refining YOLOv4 for Vehicle Detection, *International Journal of Advanced Research in Engineering and Technology*, *11*(5), 409-419.

Mailaivasan, G., Karthikram, P. (2014). Tag based image retrieval (TBIR) using automatic image annotation, *International Journal of Research in Engineering and Technology*, *3*(3), 148-153.

Manzoor, M. A., Morgan, Y. (2017). Vehicle Make and Model Classification System using Bag of SIFT Features, *7th IEEE Annual Conference on Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, ABD, 9-11 Ocak 2017.

Melenli, S., Topkaya, A. (2020). Real-Time Maintaining of Social Distance in Covid-19 Environment using Image Processing and Big Data, *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, İstanbul, Türkiye, 15-17 Ekim 2020.

Mikhail, M., Abouelseoud, Y., Elkobrosy, G. (2014). Extension and application of El-Gamal encryption scheme, *World Congress on Computer Applications and Information Systems (WCCAIS)*, Hammamet, Tunisia, 17-19 Ocak 2014.

Min, Z., Jian, J., Dihua, S., Yi, T. (2018). Vehicle Detection Method Based On Deep Learning and Multi-Layer Feature Fusion, *2018 Chinese Control And Decision Conference (CCDC)*, Shenyang, Çin, 9-11 Haziran 2018.

Mitra S., (2021) Acharya T., Data Mining: Multimedia, Soft Computing, and Bioinformatics, Wiley, https://www.wiley.com/en-us/ Data+Mining%3A+ Multimedia%2C+Soft+Computing%2C+and+Bioinformatics-p-9780471460541, (Ziyaret Tarihi: 10 Haziran 2021).

Nguyen, H. (2019). Improving Faster R-CNN Framework for Fast Vehicle Detection, *Mathematical Problems in Engineering*, *2019*(1), 1-12.

Oki, B., Pfluegl, M., Siegel, A., Skeen, D. (1993). The Information Bus: an architecture for extensible distributed systems, *Proceedings of the fourteenth ACM symposium on Operating systems principles*, Carolina, ABD, 5 - 8 Aralık 1993.

Opyrchal, L., Prakash, A. (2001). Secure distribution of events in content-based publish-subscribe systems, *10th USENIX Security Symposium*, Washington, D.C., ABD, 13-17 Ağustos 2001.

Osman, Z., Chahine, S. A. (2002). Novel speed detection scheme, *The 14th International Conference on Microelectronics*, Beyrut, Lübnan, 13 Aralık 2002.

Pallickara, S., Fox, G. (2003). NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids, *International Middleware Conference*, Rio de Janeiro, Brezilya, 16-20 Haziran 2003.

Pallickara, S., Pierce, M., Gadgil, H., Fox, G., Yan, Y., Huang, Y. (2006). A Framework for Secure End-to-End Delivery of Messages in Publish/Subscribe Systems, *In Proceedings of the 2006 7th IEEE/ACM International Conference on Grid Computing (GRID '06)*, Barselona, İspanya, 28-29 Eylül 2006.

Pelegri, J., Alberola, J., Llario, V. (2002). Vehicle Detection and Car Speed Monitoring System using GMR Magnetic Sensors, *The IEEE Annual Conference in the Industrial Electronics Society (IECON 02)*, Sevilla, İspanya, 5-8 Kasım 2002.

Pennekamp, J., Buchholz, E., Lockner, Y., Dahlmanns, M., Xi T., Fey M., Brecher, C., Hopmann, C., Wehrle, K. (2020). Privacy-Preserving Production Process Parameter Exchange, *Annual Computer Security Applications Conference*, Austin, ABD, 7-11 Aralık 2020.

Pereira, J., Fabret, F., Llirbat, F., Shasha, D. (2000). Efficient matching for Web-based publish/subscribe systems, *Cooperative Information Systems*, Eilat, İsrail, 6-8 Eylül 2000.

Phung, T.T., Ly, N. Q., Vo, T., Ho, T.N. (2021). Deep Feature Learning Network for Vehicle Retrieval, *ICMLSC'21: 2021 The 5th International Conference on Machine Learning and Soft Computing*, Da Nang Vietnam, 29 - 31 Ocak 2021.

Pillai, U. K. K., Valles, D. (2020). Vehicle Type and Color Classification and Detection for Amber and Silver Alert Emergencies Using Machine Learning, *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, Vancouver, BC, Kanada, 9-12 Eylül 2020.

Pornpanomchai, C., Kongkittisan, K. (2009). Vehicle speed detection system, *2009 IEEE International Conference on Signal and Image Processing Applications*, Kuala Lumpur, Malezya, 18-19 Kasım 2009.

Pumrin, S., Dailey, D.J. (2002). Roadside Camera Motion Detection for Automated Speed Measurement, *the IEEE 5th International Conference on Intelligent Transportation Systems*, Singapore, 6 Eylül 2002.

Rachmadi, R. F., Purnama, I. K. E. (2021). Vehicle Color Recognition using Convolutional Neural Network, arxiv, https://arxiv.org/abs/1510.07391, (Ziyaret Tarihi: 25 Haziran 2021).

Rad, A., Dehghani, A., Karim, M.R. (2010). Vehicle speed detection in video image sequences using CVS method, *International Journal of Physical Sciences*, *5*(17), 2555-2563.

Rakibe, R.S., Patil, B.D. (2013). Background subtraction algorithm based human motion detection, *International Journal of Scientific and Research Publications*, *3*(5), 2250–3153.

Ristenpart, T., Tromer, E., Shacham, H., Savage, S. (2009). Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds, *In Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09),* NY, Amerika Birleşik Devletleri, 9 - 13 Kasım 2009.

Roung, W.H.H., Wu, S. (2012). A Semantic Image Retrieval Framework Based on Ontology and Naive Bayesian Inference, *International Journal of Multimedia Technology*, *2***(**2), 36-43.

Rowstron, A., Druschel, P. (2001). Pastry, Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001),* Heidelberg, Almanya, 12-16 Kasım 2001.

Rui, Y., Huang, T. S. (1999). Image Retrieval: Current Techniques, Promising Directions, and Open Issues, *Journal of Visual Communication and Image Representation*, *10*(1), 39–62.

Sande, K. V., Gevers, T., Snoek, C. (2011). Empowering visual categorization with the GPU, *IEEE Trans. Multimedia*, *13***(**1), 60–70.

Sande, K. V., Gevers, T., Snoek, C. (2010) Evaluating color descriptors for object and scene recognition, *IEEE Trans. Pattern Anal. Mach. Intell*., *32*(9), 1582–1596.

Sang, J., Wu, Z., Guo, P., Hu, H., Xiang, H., Zhang Q., Cai B. (2018). An Improved YOLOv2 for Vehicle Detection, *Sensors*, *18*(12), 4272-4287.

Schermann, G., Schoni, D., Leitner, P., Gall, H. C. (2016). Supporting continuous deployment with automated enactment of multi-phase live testing strategies, *17th Int. Middleware Conf., 2016,* Trento, İtalya, 12-16 Aralık 2016.

Şentaş, A., Tashiev, İ., Küçükayvaz, F., Kul, S., Eken, S., Sayar, A., Becerikli, Y. (2018). Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification, *Evolutionary Intelligence*, *13*(1), 83–91

Şentaş, A., Tashiev, İ., Küçükayvaz, F., Kul, S., Eken, S., Sayar, A., Becerikli, Y. (2018). Performance Evaluation of Support Vector Machine and Convolutional Neural Network Algorithms in Real-Time Vehicle Type Classification, *The 6th International Conference on Emerging Internet, Data & Web Technologies. Lecture Notes on Data Engineering and Communications Technologies,* Tiran, Arnavutluk, 15-17 Mart 2018.

Shao, H., Wu, Y., Cui, W., Zhang, J. (2008). Image retrieval based on MPEG-7 dominant color descriptor, *The 9th International Conference for Young Computer Scientists ICYCS 20,* Hunan, Çin, 18-21 Kasım 2008.

Shetty, A., Shetty, K., Krithika, A. (2014). Review on Asymmetric Cryptography – RSA and ElGamal Algorithm, *International Journal of Innovative Research in Computer and Communication Engineering*, *2*(5), 98-105.

Shi, J., Qu, X., Feng, Y., Wang, C. (2014). A Vehicle Detection Method Based on Improved YOLOv3, *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC),* Chongqing, Çin, 12-14 Mart 2021.

Shisong, Z., Toshio, K., (2006). Feature Point Tracking for Car Speed Measurement, *the IEEE Asia Pacific Conference on Circuits and Systems (APCCAS 2006),* Singapur, 4-7 Aralık 2006.

Singha, M., Rajan, M. A., Shivraj, V. L. (2015). Balamuralidhar P., Secure MQTT for Internet of Things (IoT), *2015 Fifth International Conference on Communication Systems and Network Technologies*, Gwalior, Hindistan, 4-6 Nisan 2015.

Singha, M., Hemachandran K. (2012). Content based image retrieval using color and texture, *Signal & Image Processing: An International Journal*, *3*(1), 39–57.

Somorovsky, J., Heiderich, M., Jensen, M., Schwenk, J., Gruschka, N., Ia, L. L. (2011). All your clouds are belong to us: Security analysis of cloud management interfaces, *In Proceedings of the 3rd ACM Workshop on Cloud Computing Security,* New York, Amerika Birleşik Devletleri, 21 Ekim 2011.

Son, J.W., Park, S.B., Kim, K.J. (2007). A convolutional kernel method for color recognition, *International Conference on Advanced Language Processing and Web Information Technology*, Henan, Çin, 22-24 Ağustos 2007.

Spagnolo, P., Leo, M., Distante, A. (2006). Moving object segmentation by background subtraction and temporal analysis, *Image Vis. Comput*., *24***(**5), 411–423.

Srivatsa, M., Liu, L. (2007). Secure event dissemination in publish–subscribe networks, *Proceedings of the 27th International Conference on Distributed Computing Systems*, Toronto, Kanada, 25-27 Haziran 2007.

Tajar, T., Ramazani, A., Mansoorizadeh, M. A. (2021). Lightweight Tiny-YOLOv3 vehicle detection approach, *Journal of Real-Time Image Processing, 18*(1), 2389–2401.

Tamura, H., Yokoya, N. (1984). Image database systems: A survey, *Pattern Recognition*, *17*(1), 29-43.

Terpstra, W.W., Behnel, S., Fiege, L., Zeidler, A., Buchmann P. (2003). A peer-to-peer approach to content-based publish/subscribe, *Proceedings of the 2nd International Workshop on Distributed Event-Based Systems,* New York, Amerika Birleşik Devletleri, 8 Haziran 2003.

Tourani, A., Soroori, S., Shahbahrami, A., Khazaee, S., Akoushideh, A. (2019). A Robust Vehicle Detection Approach based on Faster R-CNN Algorithm, *4th International Conference on Pattern Recognition and Image Analysis (IPRIA),* Tehran, Iran, 6-7 Mart 2019.

Urazghildiiev, I. (2002). A vehicle classification system based on microwave radar measurement of height profiles, *2002 International Radar Conference*, Edinburgh, Birleşik Krallık, 15-17 Ekim 2002.

URL-1: https://www.confluent.io/blog/secure-kafka-deployment-best-practices/ (Ziyaret Tarihi: 14 Hazira 2021).

URL-2: https://www.oracle.com/java/technologies/java-message-service.html (Ziyaret Tarihi: 10 Hazira 2021).

URL-3: https://activemq.apache.org (Ziyaret Tarihi: 10 Temmuz 2021).

URL-4: http://activemq.apache.org/apollo/, (Ziyaret Tarihi: 15 Temmuz 2021).

URL-5: https://qpid.apache.org/ , (Ziyaret Tarihi: 5 Nisan 2021).

URL-6: https://kafka.apache.org/, (Ziyaret Tarihi: 3 Mayıs 2021).

URL-7: http://aws.amazon.com/kinesis/ , (Ziyaret Tarihi: 15 Ağustos 2021).

URL-8: http://www.cs.colorado.edu/users/carzanig/siena/, (Ziyaret Tarihi: 15 Ağustos 2021).

URL-9: http:// projects.spring.io/spring-boot/ (Ziyaret Tarihi: 15 Ağustos 2021).

URL-10: http://dubbo.io/ (Ziyaret Tarihi: 15 Ağustos 2021).

URL-11: https://docker. com/ (Ziyaret Tarihi: 18 Ağustos 2021).

URL-12: http://projects.spring.io/spring-cloud/ (Ziyaret Tarihi: 15 Ağustos 2021).

URL-13: http://mesos. apache.org/ (Ziyaret Tarihi: 12 Mayıs 2021).

URL-14: https://kubernetes.io/ (Ziyaret Tarihi: 12 Mayıs 2021).

URL-15: https://docs.docker.com/swarm/ (Ziyaret Tarihi: 18 Ağustos 2021).

URL-16: Pcmag.com. (Ziyaret Tarihi: 12 Temmuz 2021).

URL-17: Techterms.com. (Ziyaret Tarihi: 12 Temmuz 2021).

URL-18: https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/mqtt-and-coap-security-and-privacy-issues-in-iot-and-iiot-communication-protocols (Ziyaret Tarihi: 15 Ağustos 2021).

URL-19: https://d0. awsstatic.com/whitepapers/microservices-on-aws.pdf (Ziyaret Tarihi: 12 Haziran 2021).

URL-20:http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.pdf (Ziyaret Tarihi: 14 Haziran 2021).

URL-21: https://kafka.apache.org/, (Ziyaret Tarihi: 6 Haziran 2021).

Viola, P., Jones, M. (2004). Robust real-time face detection, *Int J Comput Vision, 57*(1), 137–154.

Wang, H. H., Mohamad, D., Ismail, N. (2009). Image Retrieval: Techniques, Challenge, and Trend, *International Journal of Computer and Information Engineering, 3*(12), 2908 - 2910.

Wang, H.H., Mohamad, D., Ismail, N.A. (2010). Approaches, challenges and future direction of image retrieval, *Journal of Computing*, *2*(1), 193–199.

Wang, X, Zhang, W., Wu, X., Xiao, L., Qian, Y., Fang, Z. (2019). Realtime vehicle type classifcation with deep convolutional neural networks, *Journal of Real-Time Image Processing*, *16*(1), 5-14.

Wang, X., Huo, H., Fang, T. (2006). Moving Vehicles' shadow detection with fast normalized cross-correlation, *Journal of Computer Applications, 26*(9), 2065-2067.

Wang, Y. C., Han, C. C., Hsieh, C. T., Fan, K. C., Vehicle Type Classification from Surveillance Videos on Urban Roads, *2014 7th International Conference on Ubi-Media Computing and Workshops*, Ulan Batur, Moğolistan, 12-14 Temmuz 2014.

Wicaksono, D. W., Setiyono, B. (2017). Speed Estimation On Moving Vehicle Based On Digital Image Processing, *International Journal of Computing Science and Applied Mathematics, 3*(1), 21-26.

Wilder, J. L., Milenkovic, A., Jovanov, E. (2008). Smart Wireless Vehicle Detection System, *2008 40th Southeastern Symposium on System Theory (SSST),* New Orleans, LA, ABD, 16-18 Mart 2008.

Wu, J., Liu, Z., Li, J., Gu, C., Si, M., Tan, F. (2009). An algorithm for automatic vehicle speed detection using video camera, *2009 4th International Conference on Computer Science & Education*, Nanning, Çin, 25 - 28 Temmuz 2009.

Wu, K., Xu, T., Zhang, H., Song, J. (2011). Overview of video-based vehicle detection technologies, *6th International Conference on Computer Science Education*, Singapore, 3-5 Ağustos 2011.

Wu, X., Chen, S., Huang, J., Li, A., Xiao, R., Cui, X. (2020). DDeep3M: Docker-powered deep learning for biomedical image segmentation, *Journal of Neuroscience Methods, 342*(1), 1-7.

Wu, X., Sun, S., Chen, N., Fu, M., Hou, X. (2018). Real-time vehicle color recognition based on YOLO9000, *CSPS: International Conference in Communications, Signal Processing, and Systems,* Dalian, Çin, 14-16 Temmuz 2018.

Wun, A., Jacobsen, H. A. (2007). A policy management framework for content-based publish/subscribe middleware, *In Middleware 2007 Lecture Notes in Computer Science, 4834*(1), 368–388.

Xu, F., Lu, G. (2009). Moving Object Detection Based on Ameliorative Surendra Background Update Arithmetic, *Shanxi Electronic Technology*, *5*(1), 39-40.

Xu, Z., Cao, J. (2009). Vehicle Color Extraction Based on First Sight Window, *In 1st International Conference on Information Science and Engineering (ICISE)*, Nanjing, Çin, 26 - 28 Aralık 2009.

Yu, S., Jiang, L., Xu, Z., Lan, Z., Xu, S., Chang, X., Li, X., Mao, Z., Gan, C., Miao, Y., Du, X., Cai, Y., Martin, L.J., Wolfe, N., Kumar, A., Li, H., Lin, M., Ma, Z., Yang, Y., Meng, D., Shan, S., Sahin, P.D., Burger, S., Metze, F. (2014). InformediaTRECVID 2014 MED and MER, *NIST TRECVID Video Retrieval Evaluation Workshop 2014*, Florida, ABD, 16-Ocak-2014.

Yue, M., Ruiyang, Y., Jianwei, S., Kaifeng, Y. (2017). A MQTT Protocol Message Push Server Based on RocketMQ, *2017 10th International Conference on Intelligent Computation Technology and Automation*, Changsha, Çin, 9-10 Ekim 2017.

Zarchi, M. S., Monadjemi, A., Jamshidi, K. (2014). A semantic model for general purpose content-based image retrieval systems, *Computers & Electrical Engineering*, *40*(7), 2062-2071.

Zendi, I. (2020). A Microservices-based for Distributed Deep Neural Network of Delta Robot Control System, *2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat)*, Batam, Endonezya, 17-18 Aralık 2020.

Zhang, C., Zhou, P., Li, C., Liu, L. (2015). A Convolutional Neural Network for Leaves Recognition Using Data Augmentation, *IEEE International Conference on Computer & Information Technology; Ubiquitous Computing & Communications*, Liverpool, Birleşik Krallık, 26-28 Ekim 2015.

Zhang, Z., Xu, C., Feng, W. (2016). Road Vehicle Detection and Classification based on Deep Neural Network, *7th IEEE International Conference on Software Engineering and Service Science*, Pekin, Çin, 26-28 Ağustos 2016.

Zhang, S., Shen, R. (2012). Subscription merging in filter-based publish/subscribe systems, *International Conference on Graphic and Image Processing*, Singapur, 5-7 Ekim 2012.

KİŞİSEL YAYIN VE ESERLER

Kul, S., Eken, S., Sayar, A. (2015). Service oriented warning system for detection of abondened object in video surveillance, *2015 23nd Signal Processing and Communications Applications Conference (SIU),* Malatya, Türkiye, 16-19 Mayıs 2015.

Kul, S., Eken, S., Sayar, A. (2016). Measuring the Efficiencies of Vehicle Classification Algorithms on Traffic Surveillance Video, *International Conference on Artificial Intelligence and Data Processing (IDAP16),* Malatya, Turkey, 17-18 September 2016.

Kul, S., Eken, S., Sayar, A. (2016). Evaluation of Real-Time Performance for BGSLibrary Algorithms: A Case Study on Traffic Surveillance Video, *2016 6th International Conference on IT Convergence and Security (ICITCS),* Prag, Çek Cumhuriyeti, 26-29 Eylül 2016.

Kul, S., Eken, S., Sayar, A. (2017). Distributed and collaborative real-time vehicle detection and classification over the video streams, *Int. J. Adv. Robot. Syst*., *14*(4), 1729-8814.

Kul, S., Eken, S., Sayar, A. (2017). A Concise Review on Vehicle Detection and Classification, *The Third International Workshop on Data Analytics and Emerging Services (DAES) with the International Conference on Engineering and Technology*, Antalya, Turkey, 21-23 August 2017.

Kul, S., Eken, S., Sayar, A. (2017). Konvolüsyonel Sinir Ağı Kullanarak Gerçek Zamanlı Araç Tipi Sınıflandırması, *1.Ulusal Bulut Bilişim ve Büyük Veri Sempozyumu B3S'17*, Antalya, Türkiye, 19-20 Ekim 2017.

Şentaş, A., Tashiev, İ., Küçükayvaz, F., **Kul S.,** Eken S., Sayar A, Becerikli Y. (2018). Performance Evaluation of Support Vector Machine and Convolutional Neural Network Algorithms in Real-Time Vehicle Type Classification. *The 6th International Conference on Emerging Internet, Data & Web Technologies. Lecture Notes on Data Engineering and Communications Technologies*, Tiran, Arnavutluk, 15-17 Mart 2018.

Şentaş, A., Tashiev, İ., Küçükayvaz, F., **Kul S**., Eken S., Sayar A, Becerikli Y. (2018). Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification, *Evolutionary Intelligence, 13*(1), 83–91.

Korkmaz, A., Elik, F., Aydin, F., Bulut, M., **Kul, S**., Sayar, A. (2018). Modeling Trajectory Data as a Directed Graph, *MIKE (2018),* Cluj-Napoca, Romanya, 20 - 22 Aralık 2018.

Kul, S., Durdu, P. O., Akbulut, O. (2019). Performance Comparison of EEG Channels in Emotion Recognition, *2019 27th Signal Processing and Communications Applications Conference (SIU),* Sivas, Türkiye, 24-26 Nisan 2019.

Kul, S., Tashiev, I., Şentaş, A., Sayar (2021). A., Event-Based Microservices With Apache Kafka Streams: A Real-Time Vehicle Detection System Based on Type, Color, and Speed Attributes, *IEEE Access*, *9*(1), 83137-83148.

Kul, S., Sayar, A. (2021). Entity Name Recognition in Job Postings and Resumes, 2021 3rd *International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, Karabük, Türkiye, 11-13 Haziran 2021.

Fulya, E., **Kul, S**., Sayar, A. (2021). Product-Based Reasonable Price Suggestion System in the Market with Crowd-Source and Full-Text Search, *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA),* Karabük, Türkiye, 11-13 Haziran 2021.

Gülmez, G., Cebbar, K., **Kul, S.,** Sayar, A. (2021). A Framework for Using Contactless Technologies in Operating Rooms, *2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA),* Kocaeli, Türkiye, 25-27 Ağustos 2021.

Denizgez, T. M., Kamiloğlu, O., **Kul, S.,** Sayar, A. (2021). Guiding Visually Impaired People to Find an Object by Using Image to Speech over the Smart Phone Cameras, *2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA),* Kocaeli, Türkiye, 25-27 Ağustos 2021.

.

ÖZGEÇMİŞ

Seda KUL, ilk ve orta öğrenimini Bursa Gemlik Cumhuriyet İlköğretim Okulu'nda, Lise öğrenimini Kocaeli İzmit Yahya Kaptan Lisesi'nde tamamladı. 2011 yılında girdiği Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü'nden 2015 yılında yüksek onur derecesi ile bölüm ikincisi olarak mezun oldu. 2015 yılında Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda başladığı Yüksek Lisans eğitimini 2017 yılında tamamladı. 2018 yılında Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda doktora eğitimine başlamıştır. TÜBİTAK BİLGEM'de Araştırmacı olarak çalışmaktadır.