KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

MOBESE KAMERALARI ÜZERİNDEN ÖZELLİK TABANLI SORGULAMA SAĞLAYAN DAĞITIK VE GERÇEK ZAMANLI AKILLI TRAFİK SİSTEMİ

ISABEK TASHIEV

KOCAELİ 2018

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

MOBESE KAMERALARI ÜZERİNDEN ÖZELLİK TABANLI SORGULAMA SAĞLAYAN DAĞITIK VE GERÇEK ZAMANLI AKILLI TRAFİK SİSTEMİ

ISABEK TASHIEV

Doç. Dr. Ahmet SAYAR
Danışman, Kocaeli Üniversitesi .....

Dr. Öğr. Üyesi Orhan AKBULUT
Jüri Üyesi, Kocaeli Üniversitesi .....

Doç. Dr. Mehmet Sıddık AKTAŞ
Jüri Üyesi, Yıldız Teknik Üniversitesi .....

Tezin Savunulduğu Tarih: 28.06.2018

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışmasında, araçların tip, renk, hız parametrelerine göre sınıflandırılması ve sınıflandırılmış görüntüleri tip-renk, tip-hız, renk-hız, tip-renk-hız olarak birleştirerek yayınla/kaydol sistemi ile istemcilere aktarılması amaçlanmaktadır. Sisteme kayıtlı kullanıcılara istedikleri özelliklere sahip araçlara ait bilgi/görüntülerin gerçek zamanlı olarak aktarılması gerçekleştirilmeye çalışılmıştır .

Tez çalışmamda desteğini esirgemeyen, çalışmalarıma yön veren, karşılaştığım her zorlukta desteğini ve zamanını esirgemeyen danışmanım Doç. Dr. Ahmet SAYAR'a sonsuz teşekkürlerimi sunarım.

Akademik çalışmalarım sırasında, birçok aşamada beni destekleyen Bilgisayar Mühendisliği Bölümü araştırma görevlileri Süleyman EKEN'e ve Seda KUL'a teşekkür ediyorum.

TUBİTAK projesinde beraber çalıştığım Ali SENTAŞA ve Fatmanur KÜÇÜKAYVAZ'a teşekkürlerimi sunarım.

Kocaeli Üniversitesinde yüksek lisans kazanmamda yardımcı olan Türkiye Burslarına teşekkür ediyorum.

Hayatım boyunca her zaman beni destekleyen, beni motive eden destekçilerim annem Sanavar TASHIEVA'ya, babam Süyorali TASHIEV'e ve kız kardeşlerime teşekkürlerimi sunarım.

Haziran – 2018 Isabek TASHIEV

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

CNN	:	Convolutional Neural Networks (Konvolüsyonel Sinir Ağları)
DNN	:	Deep Neural Networks (Derin Sinir Ağları)
ELU	:	Exponential Linear Unit (Eksponansiyel Doğrusal Birim Katmanı)
HLS	:	HTTP Live Streaming (HTTP Canlı Aktarım)
HOG	:	Histogram of Oriented Gradients (Yönlü Gradyanların Histogramı)
HTTP	:	Hyper-Text Transfer Protocol (Hiper-Metin Transfer Protokolü)
ITS	:	Intelligent Traffic System (Akıllı Trafik Sistemi)
PUB/SUB	:	Publish/Subscribe (Yayınla/Abone Ol)
ReLU	:	Rectified Linear Unit (Düzleştirilmiş Doğrusal Birim Katmanı)
RTMP	:	Real-Time Messaging Protocol (Gerçek Zamanlı Mesajlaşma Protokolü)
RTSP	:	Real-Time Streaming Protocol (Gerçek Zamanlı Aktarım Protokolü)
SIFT	:	Scale-Invariant Feature Transform (Ölçek Bağımsız Özellikler)
SVM	:	Support Vector Machines (Destek Vektör Makineleri)

MOBESE KAMERALARI ÜZERİNDEN ÖZELLİK TABANLI SORGULAMA SAĞLAYAN DAĞITIK VE GERÇEK ZAMANLI AKILLI TRAFİK SİSTEMİ

ÖZET

Trafik yönetim ve bilgi sistemlerinin trafik akışını düzgün bir şekilde sağlayabilmesi için çeşitli algılayıcılar yardımıyla trafikle ilgili bilgileri elde etmesi gerekmektedir. Bu kapsamda, son yıllarda trafik gözlemi ve denetiminde video kameralarının kullanımı çok yaygınlaşmış ve aktif olarak kullanılmaktadır. Görüntü işleme tabanlı video izlemeye dayalı trafik sistemleri yardımıyla plaka tanıma, araç sayısı bulma, trafik yoğunluğu tespiti, araç hızı hesaplanması, şerit ihlalleri ve araç sınıflandırma gibi birçok çalışma yapılabilmektedir.

Mobese kameraları veya diğer gözetleme kameralarından gelen video görüntüleri çok büyük boyutlara ulaştığı için bunları işlemek tek bir bilgisayarda çok zaman almakta ve doğru bilginin doğru zamanda elde edilmesini zorlaştırmaktadır.

Bu problemler dikkate alınarak bu tez kapsamında;

- Konu tabanlı olarak görüntüler içeriklerine göre kademeli olarak filtrelenerek farklı makinalara iş yönlendirilmektedir.
- Tüm video görüntüleri üzerinde tek bir bilgisayar ile işlem yapmak yerine, işbirlikçi çalışan çok sayıda makinanın iş bölümünü içeren ölçeklenebilir dağıtık sistem altyapısı oluşturulmuştur. Bu yapı konu tabanlı yayınla-kaydol orta katmanı ile gerçeklenmiştir.
- Son kullanıcılar farkı özelliklere göre (tip, renk, hız) arama kriterlerini belirleyek ve belirli konulara kaydolan kullanıcılara gelen görüntüler üzerinde daha az arama uzayında arama ve işlem yapmak zaman tasarrufu sağlanmıştır.
- Gerçek zamanda gerçeklenmesi imkânsız olan bazı video görüntü işleme algoritmalarının, gerçek ya da gerçeğe yakın zamanda gerçeklenebilmesi sağlanmıştır.

Geliştirilen sistemin; alt yapı planlaması, trafik yönetimi ve trafik suçları ile mücadele gibi farklı alanlara fayda sağlayacağı düşünülmektedir.

Anahtar Kelimeler: Akıllı Trafik Yönetimi, Dağıtık Sistemler, Konu Tabanlı Yayınla/Kaydol Modeli, Konvolüsyon Sinir Ağları, Video İşleme.

DISTRIBUTED AND REAL-TIME INTELLIGENT TRAFFIC SYSTEM WHICH PROVIDES FEATURE-BASED QUERYING ON SURVEILLANCE CAMERAS

ABSTRACT

Traffic management and information systems need to obtain information about traffic with various sensors to control the traffic flow properly. In this context, videos are very actively used in traffic surveillance and control in recent years. With the help of image processing based on video surveillance system in traffic management systems, many studies are done. Some of these are license plate recognition, finding the number of vehicles, traffic density detection, vehicle speed calculation, detection of lane violations and vehicle classification.

The most important reason not to be able to put real-time video processing applications into practice is that video data from surveillance cameras get very large in size, therefore using central processing techniques to process data and get the required information in right time and correctly is not an easy task.

Within the scope of this thesis, we deal with following issues by taking aforementioned problems into account,

- Real-time streaming video data is going to be filtered and forwarded to the right processing node by using publish-subscribe messaging middleware.
- Instead of classical approaches to centralized computation, a distributed scalable network of collaborating computation nodes is developed to process streaming real-time video data coming from traffic surveillance cameras.
- End users have a capability of searching for a vehicle defined in properties such as type, color, and speed.
- Some video processing algorithms which are impossible to realize in real-time is possible to realize and to use in real-world applications.

The proposed architecture can be utilized in different areas such as infrastructure planning, traffic management and traffic offenses.

Keywords: Intelligent Traffic Management, Distributed Systems, Topic-Based Publish/Subscribe Model, Convolutional Neural Networks, Video Processing

GİRİŞ

Akıllı Ulaşım Sistemlerini (AUS/ITS) genel olarak ulaşımı kolaylaştıracak çözümler olarak söylemek mümkündür [1]. Bu kapsamda bir AUS referans alındığında karşımıza sekiz tane temel konu başlığı çıkmaktadır. Bunlar; trafik yönetim sistemi, yolcu bilgilendirme ve rehberlik, toplu taşıma yönetimi, taşıt güvenlik ve kontrol, ticari taşıma yönetimi, afet ve acil durumların yönetimi, yol bakım yönetim, veri arşiv yönetimi diye sayılabilir [2]. AUS'un önemine binaen Enerji ve Tabii Kaynaklar Bakanlığı tarafından 2012-2023 yılları için hazırlanan Enerji Verimliliği Strateji Belgesi'nde "Ulaşımda enerji verimliliğinin arttırılması ve ağ verimliliğinin sağlanması için bilgi ve iletişim teknolojilerinin kullanıldığı akıllı trafik yönetimi uygulamaları ve akıllı ulaştırma sistemlerinin yaygınlaştırılması" eylem planı olarak ele alınmış ve Ulaştırma, Denizcilik ve Haberleştirme Bakanlığı sorumlu, Kalkınma Bakanlığı da işbirliği yapılacak kuruluş olarak belirlenmiştir [3]. Bir trafik yönetim sisteminde yeri geldiği zaman trafik videolarının analiz edilmesi ve istenen özelliklere sahip verilere ulaşılması gerekliliği kaçınılmazdır. Yukarıdaki eylem planı da göz önünde bulundurularak bu tezde; araçların tipine göre sınıflandırılması, renk ve hız gibi tanımlanmış araç özelliklerine göre konu tabanlı olarak yayınla/kaydol modeli ile istemcilere/son kullanıcılara dağıtılması işlemini gerçekleyen açık kaynak kodlu yazılımın gerçeklenmesi hedeflenmiştir.

Geliştirilen mimari ile aşağıdaki hedeflere ulaşılmıştır;

- Farklı tipteki araçların kamera görüntüleri üzerinden görüntü işleme yöntemleri ve sınıflandırıcılar kullanılarak sınıflandırılması,
- Araçların renk ve ön tanımlı hız bilgilerine göre ayrıştırılması,
- Yayınla/kaydol modeli ile sınıflandırılan, renk ve hız bilgilerine sahip görüntülerin kaydolmuş kullanıcılara konu tabanlı olarak gönderilmesi,
- Trafik video görüntülerinden tanımlanmış özelliklere göre arama yapabilecek bir ürün geliştirilmesi. Böylelikle insan ve zaman anlamında tasarruf sağlanmıştır. Genellikle güvenlikle ilgili video yorumlama görevleri, bir veya daha fazla monitör aracılığıyla kendilerine büyük miktarda sunulan verileri işlemek zorunda kalan insan

operatörleri tarafından yapılmaktadır. Bu operatörler uzun süre aynı işi yaptıklarından dikkatleri dağılmakta, böylelikle tehlikeli durumları gözden kaçırma olasılıkları artmaktadır. Aynı zamanda tüm video üzerindeki görüntüler üzerinde arama yapmak yerine belirli konulara kaydolan kullanıcılara gelen görüntüler üzerinde daha az arama uzayında arama yapmak zaman tasarrufu sağlayacaktır.

- Yapılabilirlik ve başarım testlerinin literatürde var olan veri setleri ve kendi hazırladığımız veri setleri üzerinde sınanması.
- Geliştirilen mimarinin yurt içi ve yurt dışında benzerlerinin olmamasına rağmen bazı kısımları literatürde mevcuttur; fakat araçların sınıflandırılmasından sonra konu tabanlı olarak kullanıcılara dağıtılması ile ilgili her hangi bir çalışma veya patent/faydalı modele rastlanılmamıştır. Böyle bir mimarinin tamamen açık kaynak olarak herhangi bir ekstra maliyet gerektirmeksizin geliştirilmesi ile ülkemizde başta trafik şube müdürlükleri olmak üzere kent güvenlik sistemlerinden elde edilen trafik video görüntüleri ile uğraşan diğer birimlerin fayda temin etmesi beklenmektedir.

Son yıllarda akıllı ulaşım sistemleri ve akıllı trafik yönetim uygulamaları, çeşitli devlet kuruluşları veya özel şirketler tarafından üzerinde titizlikle durulan konulardandır. Görüntü işleme yöntemleri de bu yazılım ve uygulamaları geliştirirken kullanılan temel algoritmalardandır. Genelde görüntü işleme temelli araç sınıflandırma sistemleri üzerinde literatürde birçok çalışma bulunmasına rağmen yaptığımız araştırmalar neticesinde sınıflandırılan araç görüntülerinin istemcilere konu tabanlı olarak servis edilmesi üzerinde herhangi bir çalışma yapılmamıştır. Geliştirilen mimaride kent güvenlik sistemlerinden elde edilen trafik videolarından elde edilen gerçek zamanlı görüntüler, içeriklerine göre ilgili konulara bölütlenerek o konuya ait kanaldan yayımlanmaktadır. O kanala kaydolan istemciler gerçek zamanlı görüntünün sadece ilgili olan kısımlarını (chunks) almaktadırlar. Bu şekilde ağ trafiği azalmış, görüntü işleme uygulamalarında ölçekleme ve performans problemlerine çözüm sunulmuştur.

Takip eden paragraflarda görüntü işleme temelli araç sınıflandırma çalışmalarının, renk temelli nesne tanıma çalışmalarının, hareket halindeki nesnelerin hızlarının bulunması yaklaşımlarının, mesaj iletim servislerinin ve sürekli (akan-streaming) veriler üzerine yayınla/kaydol uygulamalarının literatürdeki yerlerine ayrı ayrı değinilecektir.

Literatürde nesne tanıma ve sınıflandırma çok geniş bir alanı kapsamaktadır ve bu alanda yapılmış pek çok çalışma bulunmaktadır. Sınıflandırma probleminin bu çalışmada ilgilenilen parçası olan araç sınıflandırması ise araç tespiti ve takibine göre daha az çalışılmış bir konudur. Messelodi ve diğ. [4] ile Buch ve diğ. [5] araba tespiti için sırasıyla Kalman filtre ve Gauss dağılımına dayalı bir arka plan çıkarma yöntemi ve üç boyutlu modele dayalı bir sınıflandırma metodu kullanmışlardır. Önerilen yaklaşımlardaki sistemin çalışabilmesi için kameranın yüksekliği, bakış açısı vb. parametrelerinin bilinmesi gerekmektedir. Bunun yanında önerilen bu yöntemler hesaplama olarak oldukça karmaşık bir yapıya sahiptir. Sınıflandırma başarısı [4]'te %92,5 doğruluk (accuracy), [5]'te %90,4 anısama (recall), %87,9 hassasiyet (precision) olarak belirtilmiştir. Morris ve Trivedi [6-7] Gauss modellemeye dayalı arka plan çıkarımı ile araç tespiti yapmışlar ve tespit edilen imge bölgelerinin imge bölgesi özellikleri (alan, çevre, çevreleyen dikdörtgenin eni ve boyu vb.) çıkarılarak temel bileşen analizi (Principal Component Analysis - PCA) ve doğrusal ayırtaç analizi (Linear Discriminant Analysis - LDA) yardımıyla özellik vektöründe boyut düşürme gerçeklemişlerdir. K-yakın komşu sınıflandırması ile de araçların sınıflarını tespit etmişlerdir. Bu çalışmalardaki sonuçlar, sadece otoyolu yandan gören bir bakış açısında bir video üzerinde test edilerek verilmiştir. Sınıflandırma başarısı [6]'te %82,9, [7]'da %74,4 doğruluk olarak verilmektedir. Chen ve Zang [8] ile Zang ve diğ. [9] öğrenmeye dayalı olmayan video bölütleme metodu SPCPE (Simultaneous Partition and Class Parameter Estimation) algoritması ile araçları tespit etmişler ve onları çevreleyen dikdörtgenleri bulmuşlardır. Elde edilen araç bölgeleri normalize edilerek birbirleri arasında düzgün bir şekilde oranlanmaktadır. Normalize edebilmek için kamera parametrelerinden yararlanmışlardır. Chen ve Zang araç bölgesini çevreleyen dikdörtgenin içindeki piksel değerlerini özellik vektörü olarak kullanarak bağımsız bileşen analizi (Independent Component Analysis - ICA) ile boyut düşürme uygulanmakta, Zang ve diğ. ise PCA metoduna dayalı iki adet sınıflandırma algoritması sunmaktadırlar. İki çalışmada da sınıflandırıcı olarak SVM kullanılmaktadır. Sınıflandırma başarısı [8]'de yaklaşık %65 anısama, %75 hassasiyet, [9]'de yaklaşık %70 anısama, %50 hassasiyet olarak verilmektedir.

Gandhi ve Trivedi [10]'da yönlü eğim histogramı (Histogram of Oriented Gradients –HOG) özelliklerinin kullanıldığı araç içi bir sistem sunmuşlardır. Hareket halindeki

araç içindeki omni kamera ile elde edilen görüntülerde hareket tabanlı araç tespiti yapılmakta ve tespit edilen araçların HOG özellikleri çıkarılarak SVM ile sınıflandırılmaktadır. Sınıflandırma başarısı %64,3 doğruluk olarak verilmektedir. Zhang ve diğ. [11] ise ayarsız kamera kullanarak bir tespit ve sınıflandırma yöntemi sunmuşlardır; fakat kameranın, şerit çizgilerini paralel görecek şekilde şerit çizgileriyle aynı eksende yerleştirilmesi gerekmektedir. Ayrıca sistemin çalışabilmesi için kullanıcı tarafından piksel sayısı cinsinden uzun araç eşik değerinin ve bir gölge örneğinin sisteme girilmesi gerekmektedir. Çalışmada tespit doğruluğu %97 olarak verilirken sınıflandırma başarısına dair direkt bir bilgi verilmemiştir. Demet ve Gözde [12] ise otoyola ait herhangi bir bilgiye ve kamera parametre kalibrasyonuna ihtiyaç duymadan öğrenmeye dayalı bir sınıflandırma yöntemi sunmuşlardır. Önerilen yöntemi iki farklı video üzerinde test etmişler. İlk videoda sadece HOG özellikleri kullandıklarında %79,7 doğruluk, HOG ile beraber imge özellikleri kullandıklarında %80,6 doğruluk elde etmişlerdir. İkinci videoda ise sadece HOG özellikleri kullandıklarında %91,2 doğruluk, HOG ile beraber imge özellikleri kullandıklarında %96,4 doğruluk elde etmişlerdir. Önerilen yöntemde kullanıcıdan sadece görüntüde herhangi bir aracın tamamının görülebileceği bir tespit bölgesi belirlemesi istenmektedir. Ghada ise [13] yapmış olduğu çalışmada araç tipi sınıflandırmak için geometrik ve görünüm özelliklerini beraber kullanmıştır. Katmanlı bir sınıflandırma yapısı düşünülerek ilk katmanda araçlar küçük, orta ve büyük kategorisine ayrılmış, ikinci katmanda ise orta büyüklükteki araçlar kendi içinde tekrar sınıflandırılmıştır. Sınıflandırıcı olarak SVM seçilmiştir. Yapılan testler neticesinde ilk katmandaki sınıflandırma işleminin başarısı genişlik ve yükseklik özelliklerine göre (%96,7) ve sadece SIFT'e göre (%95,8) en yüksektir. İkinci katmandaki başarı ise yalnız SIFT (%89,6) kullanımında en yüksektir. Dong ve diğ. [14] araçların önden görünümlerini yarı denetmenli yapay sinir ağı ile sınıflandırmışlardır. Yapay sinir ağı girdi olarak aldığı her bir görüntü için olasılıksal bir sonuç döndürmektedir. Gündüz çekilmiş görüntüler üzerinde %95,7, gece çekilmiş görüntüler üzerinde ise %88,8 sınıflandırma başarısı yakaladıklarını belirtmişlerdir. Literatürdeki çalışmalar ile karşılaştırıldığında geliştirilen sistemde HOG özellikleri kullanılmış ve konu tabanlı olarak gerçek zamanlı sınıflandırılan görüntülerin/video parçalarının istemcilere servis edilmesi sağlanmıştır.

Hiyerarşi ağacındaki konulardan biri de renk temelli olarak nesnelerin/araçların tespit edilmesidir. Nesne saptama bilgisayar görüsü alanının üzerinde en çok çalışılan konularındandır. Geliştirilen nesne detektörlerinin çoğunda şekil bilgisi daha çok düşük seviye özellik temsili olarak kullanılmakla renk bilgisi ihmal edilmektedir [15-17]. Renk bilgisinin şekil bilgisi ile beraber kullanılması nesne tanımada daha efektif sonuçlar vermektedir [18]. RGB dışında YCbCr ve HSV gibi diğer renk modelleri de nesne tanımada veya segmentasyon amaçlı olarak kullanılabilmektedir. Eken [19] yüksek lisans tezinde uydu görüntülerindeki ağaçlar, yollar, binalar, sulu bölgeler ve kumlu-toprak bölgelere ait elementlerin YCbCr ve HSV renk modellerindeki bileşenlerinin karakteristik özelliklerinden yararlanarak sulu kısımları elde etmiştir. Ayrıca literatürde gürbüz ton tanımlayıcı, karşıt türev tanımlayıcı ve renk isimleri gibi renk tanımlayıcıları mevcuttur [20]. Bu çalışmada ise hareketli nesnelere ait bölgelere renk maskeleme filtresi uygulanarak istemcinin belirlediği konuya (renge) karşı düşen yerler belirlenmiştir. Bir diğer konu ise hareket halindeki nesnelerin hızlarının bulunmasıdır. Literatürde görüntü işleme teknikleri kullanılarak hareket halindeki nesnenin hızını saptamaya veya tahmin etmeye yönelik birçok çalışma mevcuttur. Bu çalışmalarda genellikle ya ekstra bir donanım kullanılmakta veya görüntü işleme teknikleri ile problemi çözmektedirler [21]. Hız saptama için Joel ve diğ. [22], Pelegri ve diğ. [23] ve Ryusuke ve diğ. [24] araç hızını tespit etmek için manyetik sensorlerle beraber bir bilgisayar yazılımından yararlanmışlardır. Harry ve diğ. [25] gerçek zamanda hızı tespit etmek için Laser-tabanlı kesintisiz bir algılama sistemi önermişlerdir. Osman ve diğ. ise [26] mikro dalga sinyalleri kullanmışlardır. Huei-Yung ve Kun-Jhih [27] araç hızını ölçmek için bulanık (blur) görüntülerden yararlanmışlardır. Bram ve Schreiber [28] AdaBoost saptama ve Lucas Kanade şablon eşleştirme tekniklerini kullanmışlardır. Pumrin ve Dailey [29] otomatik hız ölçümü için bir yöntem sunmuşlardır. Bu çalışma kapsamında kapsamında araçların hızlarının bulunması için tespit edilen araçların belirli alanlar içinde (ROI) her çerçevede piksel cinsinden ne kadar mesafe kat ettiğine bakılmaktadır.

Araçların sınıflandırılmasına ilave olarak incelediğimiz diğer bir konu ise, konu tabanlı yayınla/abone (pub/sub) ol sistemi ile sınıflandırılan görüntülerin dağıtılmasıdır. Yayınla/kaydol etkileşimi, dağıtık ortamda mesaj temelli iletişim biçimidir. Bu iletişim türünde üreticiler (producers/publishers) bilgiyi yayınlarlar,

tüketiciler (consumers/subscribers) ise almak istedikleri bilgiye kaydolarak alırlar. Yayınla/kaydol sistemi mesaj kuyruklama paradigması ile yakından ilgilidir ve mesaj temelli orta katman (middleware) sisteminin bir parçasını oluşturur. Çoğu mesajlaşma sistemi de hem pub/sub hem de mesaj kuyruklama paradigmasını API'lerinde desteklerler. Java Mesaj Servisleri de bunlardan biridir [30]. Yayınla/kaydol modelinde üretici tarafından yayınlanan bilgilerden tüketiciler bir kısmını almaktadır. Bu iş filtreleme veya kaydolma yöntemi olarak bilinmektedir. Genel olarak konu tabanlı, içerik tabanlı (content based) ve tip tabanlı (type based) filtreleme olmak üzere üç kısma ayrılmaktadır. Konu tabanlı Yayınla/kaydol modelinde mesajlar konulara (veya mantıksal kanallara) yayınlanır. İlgili konuya kaydolan tüketici o konuya yayınlanan tüm mesajları alır. Aynı mantıkla, aynı konuya kaydolan tüm alıcılar yayınlanan mesajlardan aynısını alırlar [31]. İçerik tabanlı modelde ise yayınlanan mesajlara ait niteliklerin veya içeriğin uyuşması/eşleşmesi halinde ilgili mesajlar tüketiciler tarafından alınır [32]. Tip tabanlı modeli ise nesne yönelimli programlama paradigmasından esinlenerek ortaya atılmıştır. Bu modelde; üretici haberleşme kanalında mesaj nesnelerini üretir, tüketici ise o haberleşme kanalına kaydolması kaydıyla ilgilendiği nesne tipindeki mesajları alır [33]. Literatürde konu tabanlı yayınla/kaydol sistemi ile yapılmış sınırlı sayıda yayın mevcuttur. Banno ve diğ. [34] yerel olarak üretilen mesajların yerel olarak tüketilmesini sağlayan dağıtık olay tabanlı bir sistem geliştirmişlerdir. Gascon-Samson ve diğ. [35] bulutta dinamik, ölçeklenebilir, kanal tabanlı standart bir yayınla/kaydol sisteminin gerçeklenebileceğini göstermişlerdir. Sistemin etkinliğini aynı anda aktif olan istemci sayılarına göre test etmişlerdir. Tüysüz ve diğ. [38] kişisel aktiviteleri yönetmek için iş akışı tabanlı mobil rehber tasarlamışlardır. Kullanıcılar genelde gezgin olduklarından ağ kullanımını minimize etmek için konu tabanlı yayınla/kaydol modeli mesajlaşma alt yapısını kullanmışlardır. Veri akışı (data stream) üzerine yayınla/kaydol uygulamaları da bir diğer araştırma konusudur. Veri akışı uygulamalarında akış, genellikle coğrafik olarak dağıtık olan üretici ve onları sorgulayan tüketicilerden meydana gelmektedir. Gray ve Nutt dağıtık veri akışını yayınlayıp üzerinde sorgu yapılmasına imkân veren yayınla/kaydol mimarisi temelli bir çözüm önerisi sunmuşlardır [36]. Yang ve diğ. [37] kablosuz algılayıcı ağlar ve kablosuz mesh ağlar kullanarak gerçek zamanlı video izleme için dağıtık yayınla/kaydol mimarisi önermişlerdir. Önerilen mimaride kablosuz algılayıcı ağlar

olay saptama için, kablosuz mesh ağlar ise video iletimi için kullanılmıştır. Yaptığımız araştırmalara göre sınıflandırılan araç görüntülerinin konu tabanlı olarak istemcilere/tüketicilere servis edilmesi üzerinde herhangi bir çalışma yapılmamıştır.

Bu tez çalışması, 5 bölümden oluşmaktadır. Bölüm 1, giriş bölümü olup burada çalışmanın konusu, bu teze başlanmasındaki amaçlaraçıklanmış ve konu ile ilgili geniş literatür taraması yapılmıştır. Bölüm 2'de HOG özellik tanımlayıcı, sınıflandırma için kullanılan metotlar (DVM ve KSA), yayınla/kaydol sistemleri ve bu sistemlerden Apache Kafka dağıtık mesajlaşma yapısı, NGINX gerçek zamanlı mesajlaşma protokolü, Node.js ve WebSocket gibi temel kavramlardan bahsedilmiştir. Bölüm 3'te araçların tip, hız ve renklerine göre sınıflandırılması ve dağıtık mesajlaşma yoluyla son kullanılacılara gönderilmesi için önerilen mimari hakkında bilgi verilmiştir ve herbir alt aşama için gerçeklenen deneysel sonuçlar sunulmuştur. Bölüm 4'te geliştirilen sistemin arayüz üzerinden kullanımı anlatılmıştır. Bölüm 5'te genel bir değerlendirme yapılmış ileriye dönük yapılabilecek çalışmalara yer verilmiştir.

1. GENEL BİLGİLER

1.1. HOG Özellik Tanımlayıcısı

Özellik tanımlayıcısı, bir görüntünün nesne algılama ve görüntü tanıma bilgilerini çıkararak, görüntüyü basitleştiren bir tekniktir. Genelde, özellik tanımlayıcısı herhangi görüntüyü uzunluk vektörüne dönüştürür. HOG özellik tanımlayıcısı, gradyanların yönlerinin dağılımını özellik olarak kullanır. Bir görüntünün gradyanları kullanışlıdır; çünkü gradyanların büyüklüğü kenarlar, köşeler etrafında büyüktür ve kenar ve köşelerin etrafında düz bölgelere göre nesne şekli hakkında çok daha fazla bilgi bulunur.

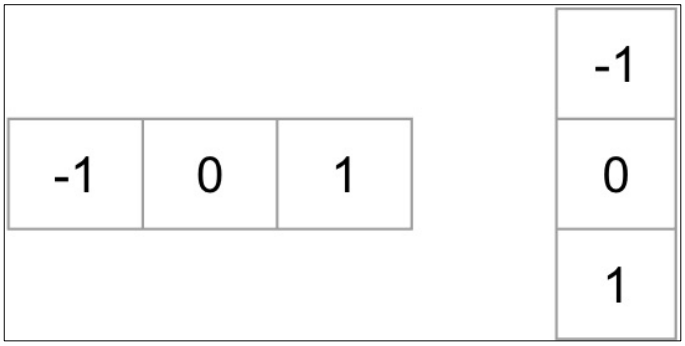
HOG özellik tanımlayıcısı, yaya tespitinde Ölçek Bağımsız Özellikler (SIFT) ve dalgacık (wavelet) yöntemlerine göre daha iyi sonuç göstermiştir [38]. Dalal ve Triggs, yaya tespiti için 64x128 boyutundaki görüntüler kullanılmışlardır ve görüntüler herhangi bir boyutta olabilirler. Yöntemin kısıdı analiz edilmesi gereken görüntülerin sabit bir en ve boy oranına sahip olması gerektiğidir. Örneğin, 90x180, 128x256 veya 100x200 görüntüler olabilir, ancak 80x140 boyutundaki görüntüler 1:2 oranını sağlamadığı için geçerli değildir.

HOG özelliklerini çıkartma algoritması aşağıdaki adımlardan oluşur:

i. Ön işlemler/Görüntüyü normalleştirme
ii. Görüntünün yatay ve dikey gradyanlarının hesaplanması
iii. Gradyanların histogramlarının hesaplanması
iv. Bloklar arası normalleştirme
v. Özellik vektörü içine birleştirme

İlk aşamada, aydınlatma efektlerinin etkisini azaltmak için görüntüyü normalleştirme yöntemleri uygulanır. Genel olarak pratikte görüntünün her bir renk kanalının karekökünü veya logaritmasını hesaplayarak gamma sıkıştırılması olarak adlandırılan yöntem kullanılır. Gamma sıkıştırma yöntemi gölgeleme ve aydınlatma varyasyonlarının etkilerini azaltmaya yardımcı olur.

İkinci aşamada, görüntünün yatay ve dikey gradyanları hesaplanır. Gradyanlar Şekil 1.1'deki çekirdek (kernel) büyüklüğü 1 olan filtreler kullanılarak hesaplanabilir.



Şekil 1.1. Görüntünün gradyanlarını hesaplamak için kullanılan filtreler

Üçüncü aşamada, görüntü hücre (cell) olarak adlandırılan küçük bölgelere ayırılır. Her bir hücre için hücredeki tüm piksellerin yatay (g_x) ve dikey (g_y) gradyanların üzerinde (1.1) - (1.2) denklemleri kullanılarak gradyanların büyüklükleri ve yönleri bulunur. Gradyanların histogramlarını hesaplamak için gradyanların büyüklüklerini ve yönlerini kullanarak her bir hücre için önceden belirtilmiş sabit sayıdaki kutular (bins) hücredeki piksellerin gradyan büyüklüklerinin değerleri ile doldurulur. Dalal ve Triggs yaya tespitinde 0, 20, 40, 60, ..., 160 açılarına karşılık gelen uzunluğu dokuz olan histogram vektörünü kullanmışlardır [38].

$$g=\sqrt{g_x^2 - g_y^2} \tag{1.1}$$

$$\theta=\arctan(\frac{g_y}{g_x}) \tag{1.2}$$

Dördüncü aşamada hücre gruplarını olarak bilinen bloklar normalleştirilir. Bloklardaki normalleştirme; aydınlatma, gölgeleme gibi problemlere karşı dayanıklı olması için yapılır. Normalleştirilmiş blok histogramları HOG tanımlayıcıları olarak da adlandılır.

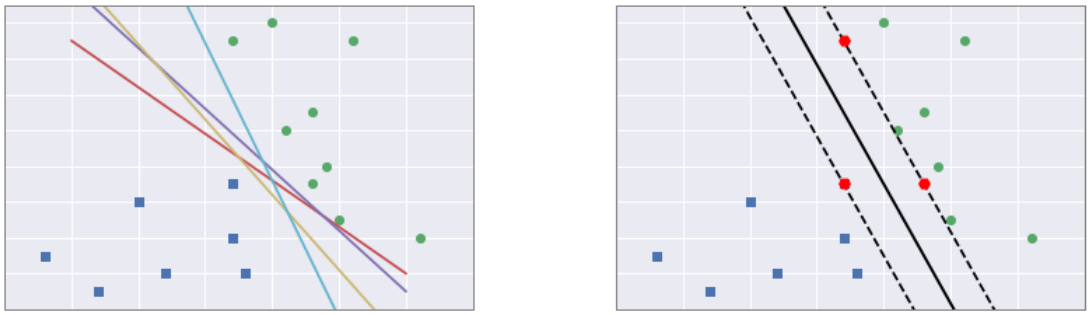
Son adımda tüm bloklardaki HOG tanımlayıcıları birleştirilerek HOG özellik vektörü hesaplanır.

1.2. Destek Vektör Makineleri (DVM)

Destek Vektör Makineleri (DVM) olarak bilinen yöntem, sınıflandırma konusunda kullanılan oldukça yaygın, etkili ve basit gözetimli öğrenme algoritmalarından biridir. DVM yönteminin çalışma mantığı; iki veya daha fazla sınıfı bilbirlerinden ayırabilen en uygun karar fonksiyonun tahmin edilmesi, başka bir deyişle sınıfları birbirlerinden en uygun bir şekilde ayırabilen hiper düzlemin (hyperplane) tanımlanmasına dayanmaktadır [39].

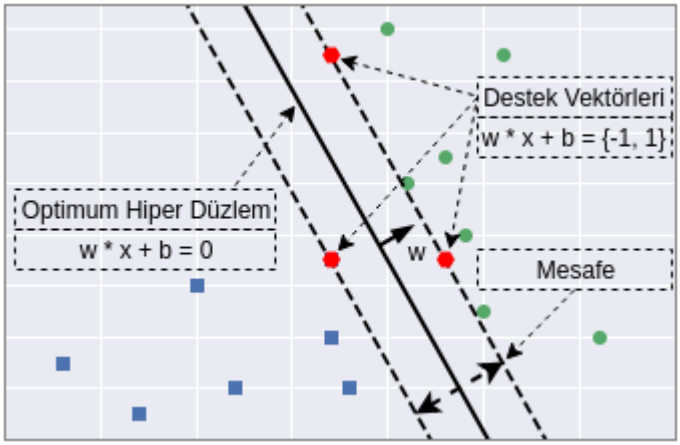
Destek vektör makineleri ile ikili sınıflandırma uygulamalarında genellikle sınıf etiketleri {-1, +1} olarak tanımlanır ve iki sınıfa ait örneklerin, eğitim verisi kullanılarak elde edilen karar fonksiyonu ile birbirinden ayrılması amaçlanır ve karar fonksiyonu kullanılarak eğitim veri setini optimal şekilde ayırabilecek hiper düzlem bulunur.

Şekil 1.2(a)'da gösterildiği üzere iki sınıflı veri setini birbirinden ayırmak için birden fazla hiper düzlem çizilebilir. Ancak, DVM'in hedefi iki sınıf vektörlerinin arasındaki maksimum genişlikteki marjini bulmaktır. Şekil 1.2(b)'de görüldüğü üzere iki sınıflı veri setinin birbirine en yakın noktaların birbirlerinden maksimum genişlikte uzaklaştıran siyah çizgiye optimum hiper düzlem ve hiper düzlemin sınırlarını geçen kırmızı noktalar ise destek vektörleri olarak adlandırılır.



Şekil 1.2. (a) İki sınıflı problem için hiper düzlemler, (b) Optimum hiper düzlem ve destek vektörleri

Şekil 1.3'te gösterildiği üzere doğrusal ayrılabilen sınıflandırma problemlerinde optimum hiper düzlemi belirlemek için destek vektörlerinin arasındaki uzaklık maksimize edilmelidir.



Şekil 1.3. Optimum hiper düzlemin belirlenmesi

Doğrusal olarak ayrılabilen ikili sınıflandırma problemlerinde DVM eğitimi için n sayıda örnekten, m sayıda özellikten ve n sayıda etiketlerden oluşan eğitim veri seti (1.3) – (1.4) matrislerinde gösterildiği üzere kabül edilirse, optimum hiper düzleme ait eşitsizlikler (1.5) – (1.6) gösterildiği şekilde olur:

$$x=\begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1m} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & x_{n3} & \cdots & x_{nm} \end{bmatrix}, \quad x \in R^{nm} \tag{1.3}$$

$$y=\begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{bmatrix}, \quad y \in \{-1, +1\} \tag{1.4}$$

(1.5) – (1.6) eşitsizliklerinde $x \in R^{nm}$ olup, veri seti özelliklerini, $y \in \{-1, +1\}$ ise sınıf etiketlerini, w ağırlık vektörünü ve b eğilim değerini göstermektedir. Optimum hiper düzlemi seçebilmek için bu düzlemi iki hiper düzlem içinde Şekil 1.3'te gösterildiği üzere medyan çizgisi olarak çizilmesi gerekir.

$$wx_i+b \geqslant +1, \quad \text{her } y= +1 \text{ için} \tag{1.5}$$

$$wx_i+b \leqslant -1, \quad \text{her } y= -1 \text{ için} \tag{1.6}$$

(1.5) – (1.6) eşitsizliklerinin iki tarafındaki ifadeleri y ile çarpılarak (1.7) denklemindeki sonuç elde edilir:

$$y_i(wx_i+b) \geqslant 1,\ y \in \{-1, +1\} \tag{1.7}$$

(1.7) eşitsizliğini kullanarak sadece destek vektörleri için doğru olan (1.8) denkleme dönüştürürüz.

$$y_i(wx_i+b)-1=0 \tag{1.8}$$

Doğrusal ayrılabilen ikili sınıflandırma problemlerinde optimum hiper düzlemi bulmak için, optimum hiper düzlemin iki tarafındaki yardımcı hiper düzlemler arasındaki mesafe birbirlerinden maksimum bir şekilde uzaklaştırılmalılar. Böylece iki sınıfın örneklerini optimum bir şekilde farkedebilen sınıflandırıcı ortaya çıkar. İki yardımcı hiper düzlemlerin arasındaki mesafe (1.9) denkleminde gösterildiği gibi bulunur.

$$\text{mesafe}=(x_+-x_-)\frac{w}{\|w\|} \tag{1.9}$$

(1.9) denkleminde (1.10) – (1.11) denklemleri kullanılarak (1.12) formülü ortaya çıkar.

$$wx_+=1-b,\ y=+1 \text{ için} \tag{1.10}$$

$$wx_-=-1-b,\ y=-1 \text{ için} \tag{1.11}$$

$$\text{mesafe}=\frac{2}{\|w\|} \tag{1.12}$$

(1.12) denklemi kullanılarak optimum hiper düzlemin sınırları arasındaki mesafe maksimum değere ulaşması için *w* normal vektörünün büyüklüğü (magnitude) minimum hale getirilmesi gerekir. Optimum hiper düzlemin bulunması için (1.13) denkleminde gösterildiği gibi optimizasyon problemine dönüştürülür.

$$\min\left[\frac{1}{2}\|w\|^2\right] \tag{1.13}$$

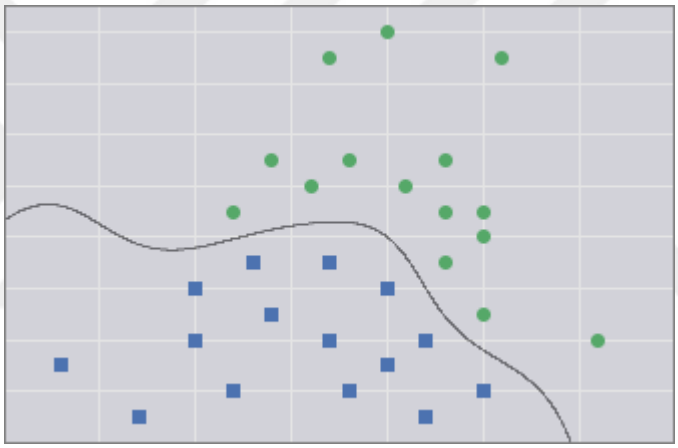
(1.13) optimizasyon problemi, Lagrange denklemleri kullanılarak (1.14) denkleminde gösterildiği gibi denkleme dönüştürülür.

$$L(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum\alpha_i[y_i(wx_i+b)-1] \quad (1.14)$$

En sonunda, doğrusal olarak ayrılabilen iki sınıflı DVM için karar fonksiyonu (1.15) denkleminde gösterildiği gibi yazılır.

$$f(x)=sign(\sum\lambda_i y_i(x.x_i)+b) \quad (1.15)$$

DVM doğrusal olarak ayrılamayan sınıflar için de kullanılır. Pratikte gerçekleştirilen çoğu uygulamalarda sınıfların optimal hiper düzlem ile doğrusal olarak ayrılması her zaman mümkün değildir. Şekil 1.4'te iki sınıflı bir örneğin geometrik olarak doğrusal ayrılamama durumu gösterilmiştir.



Şekil 1.4. Doğrusal ayrılamayan iki sınıf durumu

Doğrusal olarak ayrılamayan sınıflar, girdi uzayının boyutu daha büyük boyuta dönüştürülerek, doğrusal olarak hiper düzlem ile ayrılabilmektedir. Giriş verilerinin girdi uzayının boyutun farklı bir uzay boyutuna dönüştürme işlemlerli çekirdek (kernel) fonksiyonlarının yardımı ile gerçekleştirilmektedir. Çekirdek fonksiyonlar matematiksel olarak (1.16) denkleminde gösterildiği gibi ifade edilmektedir.

$$K(x_i,x_j)=\phi(x_i)\phi(x_j) \quad (1.16)$$

Sonuç olarak, doğrusal olarak ayrılamayan iki sınıflı DVM için karar fonksiyonu (1.17) denkleminde gösterildiği gibi yazılır.

$$f(x)=sign(\sum \lambda_i y_i\phi(x_i)\phi(x_j)+b) \quad (1.17)$$

DVM'de Tablo 1.1'de gösterilmiş olan çekirdek fonksiyonları kullanılır. Doğrusal olarak ayrılamayan veri setleri için genelde radyal tabanlı fonksiyon pratikteki uyugulamalarda diğer fonksiyornlara göre daha çok kullanılır. Polinom çekirdek fonksiyonunun derecesinin artmasıyla sınıflandırma hızı düşer. Bu nedenle küçük derece değerlerini kullanmak daha mantıklıdır.

Tablo 1.1. Destek Vektör Makinelerinde kullanılan çekirdek fonksiyonlar

Çekirdek Fonksiyonlar	Matematiksel İfadeleri
Doğrusal (linear) Çekirdeği	$K(x_i,x_j)=x_i x_j$
Polinom (polynomial) Çekirdeği	$K(x_i,x_j)=(\gamma x_i x_j+1)^d$
Radyal Tabanlı (radial basis) Fonksiyon Çekirdeği	$K(x_i,x_j)=e^{-\gamma\|x_i-x_j\|^2}$
Sigmoid Çekirdeği	$K(x_i,x_j)=\tanh(\gamma x_i x_j+1)$
Eksponansiyel (exponanial) Chi^2 Çekirdeği	$K(x_i,x_j)=e^{-\gamma x^2(x_i x_j)}$
Histogram Kesişimi (intersection) Çekirdeği	$K(x_i,x_j)=\min(x_i,x_j)$

1.3. Konvolüsyonel Sinir Ağları (KSA)

Konvolüsyonel ağlar, ya da Konvolüsyonel Sinir Ağları olarak adlandırılmış olan sinir ağları grid topolojisine sahip olup verileri işlemek için kullanılan sinir ağlarının bir çeşididir. Örnek olarak zamanla akan ya da görüntü verileri söylenilebilir. KSA pratikte son derece başarılı olduklarını kanıtlamışlardır. KSA tabiri ağın "konvolüsyon" denilen matematiksel bir işlemi kullandığını belirtir. "Konvolüsyon" doğrusal işlemlerin özel bir çeşitidir. KSA ağ mimarisinde tam bağlı katmanlar yerine en az bir kere "konvolüsyon" katmanını kullanan basit sinir ağlarının bir türüdür.

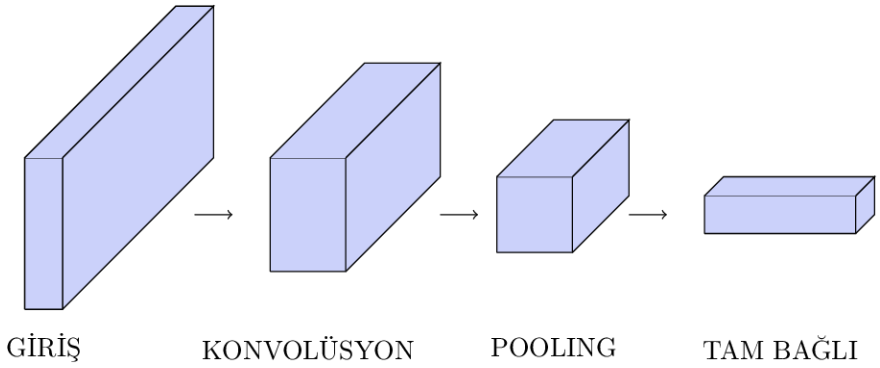
1.3.1. Konvolüsyonel sinir ağlarının mimarisi

Genelde sinir ağları giriş, çıkış ve gizli katmanlardan oluşur. Giriş olarak tek bir vektörü alarak gizli katmanların yardımıyla dönüştürme işlemlerini yapar. Gizli katmanların nöronlarının her biri bir önceki katmandaki nöronlara tamamen bağlıdır. En son katman çıkış katmanı olarak adlandırılır ve çıkış katmanı çözmek istenilen

problemlerin tipi eğer sınıflandırma ise hangi sınıfa ait olduğunun olasılığını eğer regresyon problemi ise gerçek bir sayı üretir. Örneğin KSA ile sınıflandırma denince yaygın olarak kullanılan LeNet-5 mimarisi gelir. Bu mimaride ağ girişi olarak 28x28 boyutlarındaki siyah-beyaz görüntü ile beslenir ve sonuç olarak da görüntünün içindeki ifadenin sıfırdan dokuza kadar olan rakamlardan hangisine karşılık geldiği olasılığını tespit eder [40]. Regresyon için pratikte kullanılan örneklerin biri ise ev fiyatlarını tahmin etmedir [41]. Ağ giriş olarak evlerin oda sayısı, konumunu ve yapım yılı gibi özellikler ile beslenir ve çıkışta evin fiyatını tahmin eder.

Normal sinir ağları büyük boyutlu görüntüler için iyi bir şekilde ölçeklenemez. Örneğin CIFAR-10 veri setindeki resimlerin boyutu 32x32x3 şeklinde tanımlanmıştır [42]. Böylece ilk tam bağlı katmandaki nöronların sayısı 32x32x3 = 3072 olur. Göründüğü gibi 3072 çok büyük bir sayı olmadığından ağ mimarisi için uygun olabilir. Ancak görüntünün boyutu büyüdükçe girişteki nöronların sayısı hızla artar. Mesela boyutu 200x200x3 olan bir görüntü için parametlerin sayısı 200x200x3 = 120000 olur. Bu nedenle tam bağlı katmanların kullanılması zamanı boşa harcamaktadır. Ayrıca parametrelerin sayısı arttıkça sinir ağları hızlı bir şekilde aşırı öğrenmeye maruz kalırlar.

Genelde KSA girişi görüntülerden oluştuğundan, görüntülerin ağ mimarisini daha mantıklı bir şekilde sınırlandırmasından olumlu yönde faydalanır. Özellikle KSA'nın normal sinir ağlarından farkı ise konvolüsyon katmanları üç boyuta sahiptir: genişlik, yükseklik ve derinlik. Derinlik parametresi, bir ağdaki toplam katman sayısını ifade eden tam bir KSA derinliğine değil, bir aktivasyon matrisinin üçüncü boyutunu gösterir. Örneğin CIFAR-10 veri setindeki görüntüler giriş aktivasyonları olarak kullanılabilir ve boyutları 32x32x3 olur. Her katmandaki nöronlar bir önceki katmandaki tüm nöronlara bağlanacağının yerine sadece küçük bir alana bağlanırlar. Üstelik CIFAR-10 veri seti için KSA çıktısı 1x1x10 boyutunda olur. Şekil 1.5'te gösterildiği gibi KSA tam görüntünün üzerinde konvolüsyon, pooling işlevlerini uygulayarak görüntünün boyutunu küçültür ve çıkış olarak 10 tane sınıfın olasılıklarını tutan bir matris üretir. Üretilen matrisin içeriğine bakarak giriş görüntünün hangi sınıfa ait olduğu öğrenilir.



Şekil 1.5. Konvolüsyon Sinir Ağının basit mimarisi

1.3.2. Konvolüsyonel sinir ağlarındaki katmanlar

KSA çeşitli katmanlardan oluşur. Her katmanın kendine ait bir sorumluluğu vardır. Genel olarak KSA mimarisinde aşağıdaki üç ana katman kullanılır. Bunlar konvolüsyon, pooling ve tam bağlı (fully connected) katmanları olarak adlandırılmışlardır. Herhangi bir KSA mimarisi oluşturulurken bu üç katman belli bir düzende toplanır. Bahsedilen katmanları kullanarak basit bir KSA mimarisi geliştirilebilir.

1.3.3. Konvolüsyon katmanı

Konvolüsyon katmanı matematiksel hesaplamaların büyük bir kısmını yapan temel katmandır. Konvolüsyon katmanının parametreleri ise öğrenilebilir filtrelerden ya da diğer adıyla çekirdeklerden (kernellerden) oluşmaktadır. Filtrelerin boyutu küçüktür; ancak derinliği girişin derinliği ile eşit olmalıdır. Örneğin, KSA ilk katmanındaki filtrenin boyutu 5x5x3 olabilir. Başka bir deyişle 5 piksel genişlik, 5 piksel yükseklik ve derinliği 3 olarak da adlandırılır. İleri geçiş sırasında (forward), her bir filtreyi girişin genişliği ve yüksekliği boyunca kaydırarak, filtrenin değerleri ile giriş arasındaki her bir pozisyondaki çarpımı bulunur. Filtreyi girişin genişliği ve yüksekliği boyunca kaydırdıkça bu filtrenin her mekansal pozisyonda verdiği yanıtlardan 2-boyutlu bir aktivasyon haritası (activation map) ortaya çıkar. İçgüdüsel olarak ağ eğitim sırasında kenar gibi görsel özellikleri ya da renk lekelerini gördüğünde aktif hale gelen filtreleri öğrenecektir. Konvolüsyon katmanlarındaki filtreler birbirlerinden farklı olarak 2-boyutlu aktivasyon haritalarını üretirler. Üretilmiş aktivasyon haritalarını girişin derinliği boyunca toplanır ve çıkış matrisi üretilir.

Yüksek boyutlu görüntüler ile calışırken bir katmanın nöronlarını bir önceki katmanın her bir nöronu ile bağlamak mantıksız bir yöntemdir. Bunun yerine her bir nöron girişin yerel bir bölgesine bağlanır ve bu yerel bölge algı alanı (receptive field) olarak adlandırılır. Yerel algı alanının ve o alana bağlanan filtrelerin boyutu aynı olması gerekmektedir. Örneğin, giriş görüntünün boyutunun 32x32x3 olduğunu varsayalım. Eğer algı alanının boyutu 5x5 ise konvolüsyon katmanındaki her bir nöron toplamda 5x5x3=75 ağırlık ve bir yanlılık (bias) parametrelerine sahip olacaktır. Dikkat edilecek nokta ise girişin derinliği ne kadar ise filtrelerin derinliği ona eşit olmasıdır.

Konvolüsyon katmanındaki çıkışın nöron sayısı üç tane hiper parametre ile kontrol edilir. Bunlar derinlik (depth), kaydırma adım uzunluğu (stride) ve sıfır-tamponlama (zero-padding).

Derinlik; kullanmak istenilen filtrelerin sayısına karşılık gelir ve her bir filtre giriş görüntüsünde farklı birşeyleri aramayı öğrenir. Örneğin ilk konvolüsyon katmanında kullanılan filtreler giriş görüntüsündeki kenarları ve renk lekeleri görünce aktif hale gelirken ileri katmanlardaki filtreler eğitim sırasında öğrenilmiş olan şekilleri görünce aktif hale gelirler.

Filtrelerin kaydırılacağı adım uzunluğu belirtilmelidir. Adım uzunluğu bir olduğunda filtreleri sağ tarafa doğru bir adım kaydırarak giriş genişliğinin sonuna ulaştığında aşağı tarafa doğru bir adım kaydırarak filtrenin sağ alt tarafı girişin sağ alt tarafına ulaşana kadar bu adımlar tekrarlanacaktır. Kaydırma adım uzunluğu büyüdükçe çıkışta elde edilecek çıktının boyutu küçülür. Kaydırma adım uzunluğu ne kadar küçük ise, görüntüden daha fazla bilgi elde edilir. Kaydırma adım uzunluğu büyüyünce görüntüdeki bazı önemli bilgiler kaybedilebilir [43].

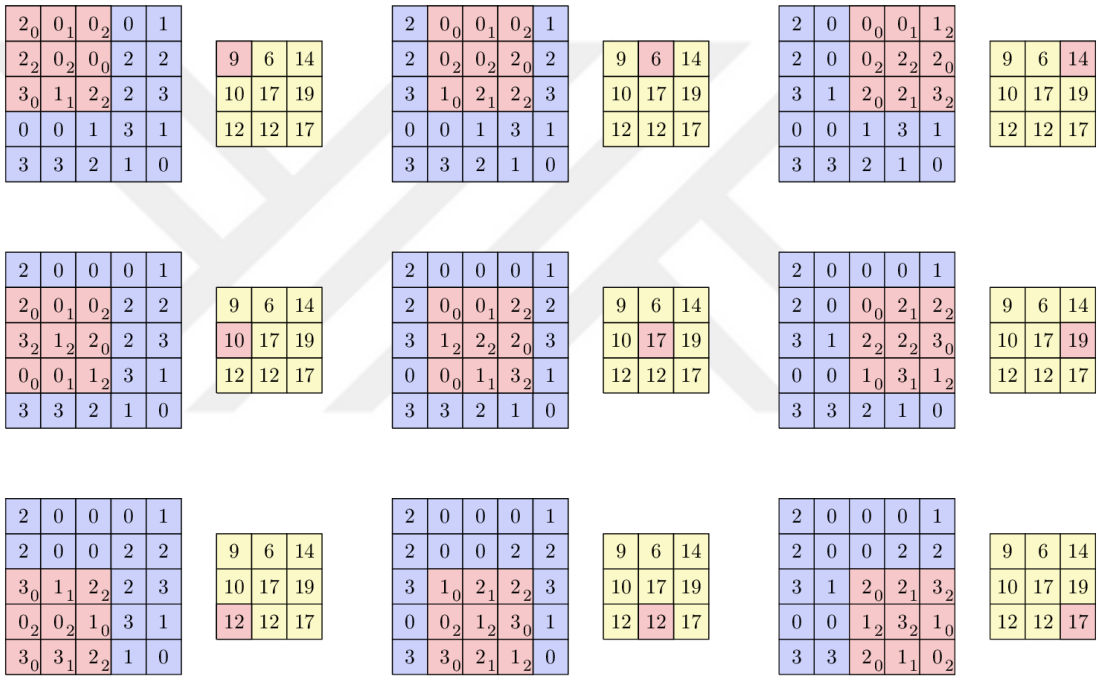
Bazı durumlarda girişin kenarlarını sıfırlar ile doldurmak uygun görülür. Böylece çıkışın boyutunu kontrol edebilecek bir tane daha hiper parametre sıfır-tamponlama yardıma gelir.

Girişin boyutunu kare olarak varsayarak konvolüsyonel katmanın çıkış boyutu (o) giriş boyutuna (i), algı alanının boyutuna (k), kaydırılacağı adım uzunluğuna (s) ve

sıfır-tamponlama sayısına (p) bağlı olarak denklem 1.18'de gösterildiği üzere bulunabilir.

$$o=\lfloor \frac{i-2p+k}{s} \rfloor+1 \tag{1.18}$$

Şekil 1.6'da giriş boyutu 5x5, kullanılacak filtrelerin boyutu 3x3, kaydırılacak filtrelerin adım uzunluğu 1 ve sıfır-tamponlama sayısı sıfır kullanıldığında konvolüsyon katmanının çıkış boyutu 3x3 olmaktadır. Eğer kaydırılacak adım uzunluğu 2 ise çıkış boyutu 2x2 olmaktadır.

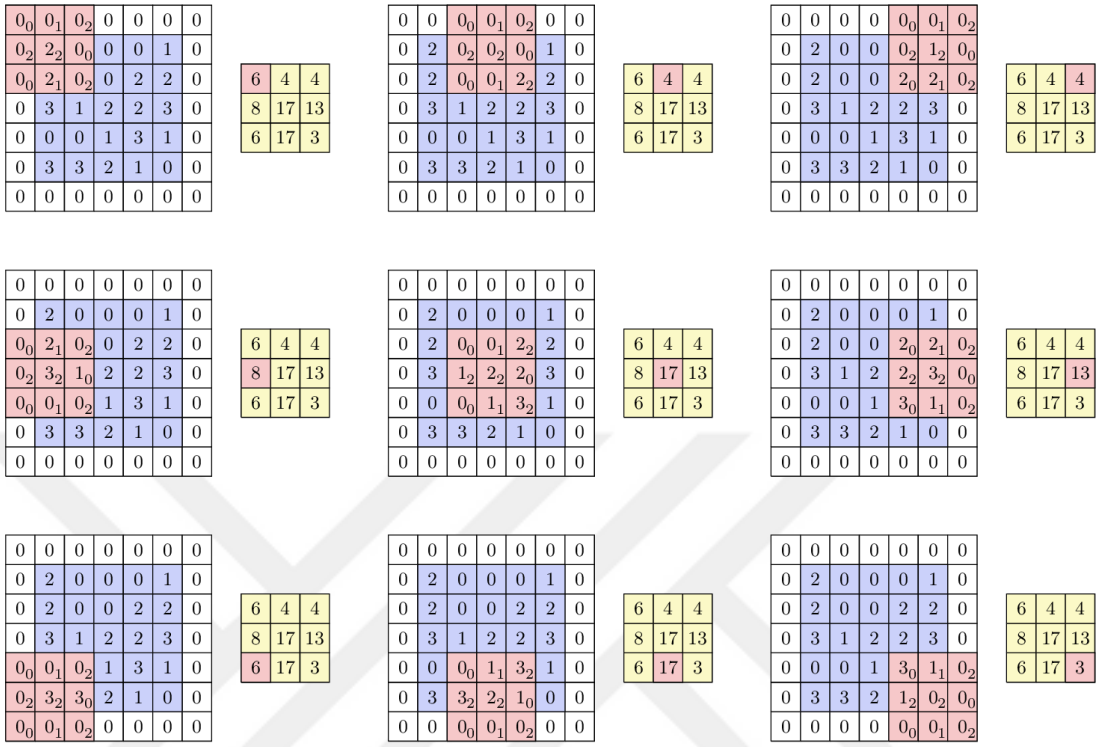


Şekil 1.6. Çıkışın kaydırma adım uzunluğuna ve sıfır-tamponlamaya olan bağımlılığı

Kaydırma adım uzunlukları ayarlanırken önem verilmesi gereken koşullar vardır. Örneğin, giriş boyutu i=10, sıfır-tamponlama p=0 ve filter boyutu k=3 ise kaydırma adım uzunluğu s=2 kullanılamaz. Eğer bu parametler kullanılarak çıkış boyutu (1.19) formülü ile hesaplanırsa çıkış boyutunun 4.5 olduğu ve tam sayı olmadığından dolayı kaydırma adım uzunluğu geçersiz olduğunu belirtir. KSA mimarisini oluştururken bu tür problemleri çözmek için sıfır-tamponlama gibi yöntemler kullanılır.

$$o=\lfloor \frac{i-2p+k}{s} \rfloor+1=\lfloor \frac{10-2x0+3}{2} \rfloor+1=4.5 \tag{1.19}$$

Bazı durumlarda Şekil 1.7‘de gösterildiği üzere sıfır-tamponlama hiper parametresi çıkış boyutunu ayarlamak için kullanılır.



Şekil 1.7. Sıfır-tamponlamanın çıkış boyutuna etkisi

Krizhevsky ve arkadaşları tarafından geliştirilmiş olan Konvolüsyon Sinir Ağları mimarisi 2012 yılında ImageNet yarışmasında boyutu 227x227x3 olan görüntüleri giriş olarak alacak şekilde tasarlanmıştr. İlk konvolüsyon katmanının algı alanı boyutu k = 11, kaydırma adım uzunluğu s = 4, sıfır tamponlama p = 0 ve 96 tane filtre kullanılarak denklem 1.20’de gösterildiği gibi çıkış boyutu 55x55x96 boyutlu çıkış matrisi üretilmiştir. Çıkış matrisin her bir nöronu 11x11x3 olan bölgeye bağlanmıştır [44].

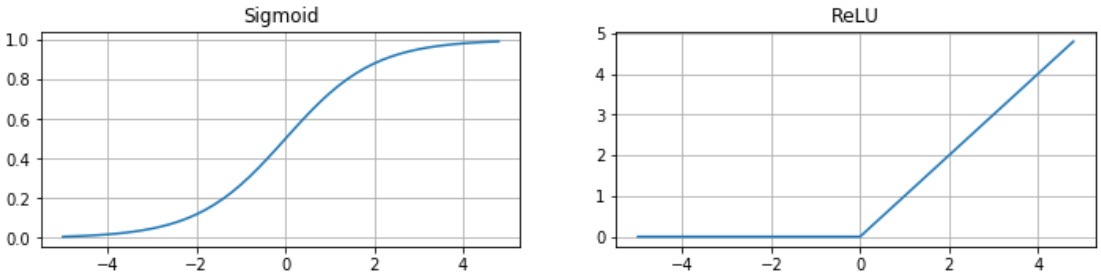
$$o=\lfloor \frac{i - 2p + k}{s} \rfloor+1=\lfloor \frac{227 - 2x0 + 11}{4} \rfloor+1=55 \tag{1.20}$$

1.3.4. Aktivasyon katmanı

Derin öğrenme ağlarında aktivasyon fonksiyonlarının seçimi, modelin eğitim zamanını ve performansını önemli bir şekilde etkilemektedir. Şu anda, en başarılı ve yaygın kullanılan aktivasyon fonksiyonu Düzleştirilmiş Doğrusal Birim Katmanıdır

(ReLU) [45]. ReLU aktivasyonu fonksiyonu diğer fonksiyonlara göre eğitim sürecüne haracanan zamanı azaltır. Hiperbolik tanjant aktivasyon fonksiyonu, CIFAR-10 veri seti için 4 katmanlı KSA modeli 25% eğitim hatasına 37 dönemde (epoch) ulaşırken, ReLU fonksiyonu 6 dönemde ulaşmaktadır [44].

Konvolüsyon katmanlarından sonra genellikle doğrusal olmayan (non-linear) aktivasyon fonksiyonları modele doğrusalsızlık katmak için kullanılır. Doğrusal olmayan aktivasyon fonksiyonlarının kullanılmasının temel amacı ise konvolüsyonel katmanları tarafından üretilen çıktılardaki doğrusallığı gidermektir. Çünkü tüm katmanlardaki aktivasyon fonksiyonları doğrusal olabildiğinden dolayı modelin ürettiği sonuç, çıktıların linear kombinasyonu olarak hesaplanabilir. Şekil 1.8'de sigmoid ve ReLU aktivasyon fonksiyonların grafikleri gösterilmiştir. Pratikte en çok kullanılanı ReLU'dur. ReLU'da sigmoid aktivasyon fonksiyonuna göre parametreler daha hızlı bir şekilde öğrenilmektedir. Bu iki fonksiyondan başka da hiperbolik tanjant, Leaky ReLU, ELU, softmax gibi aktivasyon fonksiyonların çeşitleri bulunmaktadır [46-48]. Son zamanlarda Google tarafından geliştirilen Swish aktivasyon fonksiyonu ImageNet veri setini kullanarak eğitilmiş olan Mobile NASNet-A'nın performansını 0,9% ve Inception-ResNet-v2'nin performansını 0,6% arttırmaktadır [49].

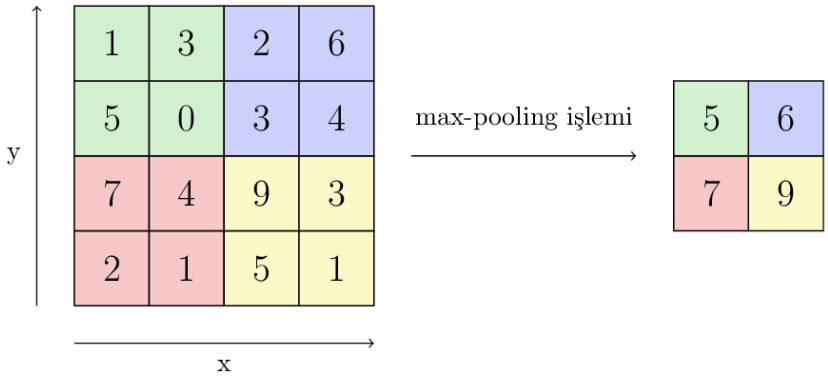


Şekil 1.8. Sigmoid ve ReLU aktivasyon fonksiyonları

1.3.5. Pooling katmanı

Pratikte ardışık konvolüsyonel katmanların arasına periodik olarak bir pooling katmanı eklemek yaygın bir yöntemdir. Pooling katmanı; ağ mimarisindeki parametrelerin ve hesaplamaların sayısını azaltan ve aşırı öğrenmeye engel olan bir mekanizmadır. KSA'da en yaygın kullanılan pooling çeşidi ise max pooling'dir. Pooling katmanı bağımsız olarak haraket ederek ve "max" işlemini kullanarak girişi

yeniden boyutlandırır. Şekil 1.9'da boyutu 2x2 olan filtrelerin her biri genişlik ve yükseklik boyunca uygulandığında aktivasyonların 75%'i atılmaktadır.



Şekil 2.9. Max-pooling işlemi

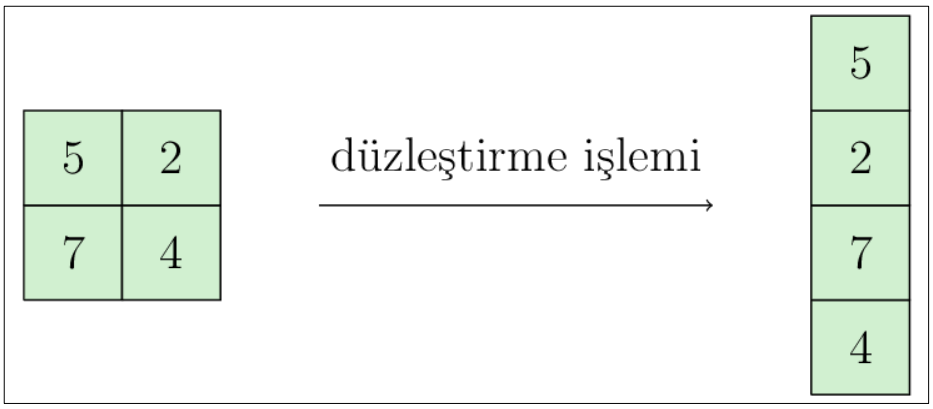
Genelde pooling katmani üç boyutlu girişi alır ve çıkış olarak üç boyutlu bir matris üretir. Çıkışın boyutunu ayarlamak için genelde iki tane hiper parametre kullanılır. Onlar kaydırma adım uzunluğu (stride) s ve pencere büyüklüğü k. Çıkış matrisinin boyutları denklem 1.21 ile bulunur.

$$o=\lfloor \frac{i - k}{s} \rfloor+1 \tag{1.21}$$

Pratikte pooling katmanının yaygın olarak görülen iki çeşiti f=3, s=2 hiper parametrelerini kullanan, overlapping pooling olarak adrandırılan ve f=2, s=2 hiper parametrelerini kullanan pooling katmaları kullanılmaktadir. Büyük boyuttaki pooling katmanları pratikte iyi sonuçlara ulaşmamışlardır. Pooling katmanında max poolingden başka ortalama pooling (average pooling) ve L2-norm pooling fonksiyonları kullanılmaktadır [50].

1.3.6. Düzleştirme katmanı (Flattening)

KSA ile sınıflandırma uygulamaları geliştirilirken modelin son katmanı olarak tam bağlı katman kullanılır. Düzleştirme katmanının amacı, konvolüsyon katmanı ve tam bağlı katmanları birleştirmektir. Şekil 1.10'da gösterildiği gibi 2x2 olan iki boyutlu matrisi 4x1 boyutlu matrise dönüştülür.



Şekil 1.10. Düzleştirme işlemine örnek

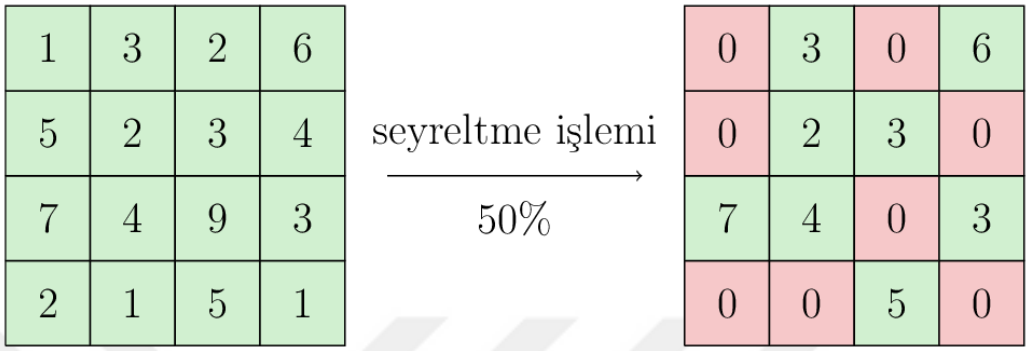
1.3.7. Tam bağlı katmanı

Tam bağlı katman, çıktı katmanında bir softmax aktivasyon fonksiyonu kullanan geleneksel çok katmanlı perceptrondur. Tam bağlı terimi, önceki katmandaki her nöronun sonraki katmandaki her nörona bağlı olduğunu ima eder. Konvolüsyon ve pooling katmanlarındaki çıktı giriş görüntüsünün yüksek düzeydeki özelliklerini temsil eder. Tam bağlı katmanın amacı, yüksek düzeydeki özellikleri kullanarak giriş görüntüsünün önceden belirlenmiş sınıflara göre olasılıklarını tespit etmektir. Softmax aktivasyon fonksiyonunun çıkış olasılıklarının toplamı her zaman bire eşittir.

1.3.8. Seyreltme katmanı (Dropout)

Seyreltme, Geoff Hinton ve Toronto Üniversitesindeki öğrencileri tarafından geliştirilen sinir ağları için en etkili ve en yaygın olarak kullanılan regülarizasyon tekniğidir [51]. Seyreltme tekniği ile rastgele nöron çıkışlarının sıfırlanması ile modelin ezberleme probleminin önüne geçilmektedir [44]. Seyreltme katmanı tam bağlı katmanlardan sonra kullanılmaktadır. Çünkü tam bağlı katmanların parametre sayısı büyük olabildiğinden dolayı modelin ezberleme olasılığını arttırır. Seyreltme katmanı için eşik değeri 0 ve 1 arasındaki tanımlanabilir ancak, pratikte 0,2 ve 0,5 arasındaki değerli kullanılmaktadır. Örneğin, seyreltme eşik değeri olarak 0,2 seçilirse tam bağlı katmanların aktivasyon değerlerinin 20% etkisiz hale getirilir, başka bir deyişle sıfırlanır.

Şekil 1.11‘de gösterildiği üzere 4x4 matrise 0,5 eşik değeri ile seyreltme işlemi uygulandığında matrisin değerlerinin 50% sıfırlanır. Seyreltme işlemi aktivasyon matrislerine sadece eğitim sırasında uygulanır. Test sırasında seyreltme işlemi aktivasyon matrisine uygulanmaz.



Şekil 1.11. Eşik değeri 0.5 ile seyreltme işleminin yapılması

Neden seyreltme modelin ezberlemesine engel olur sorusuna Geoff Hinton, diğer şeylerin yanı sıra bankalar tarafından kullanılan dolandırıcılığı önleme mekanizmasından ilham aldığını söylüyor. Kendi sözleri ile: “Bankaya gittim. Kasiyerler sürekli değişmeye devam etti ve ben onlardan nedenini sordum. Onlar da bilmediklerini söylediler. Bankanın başarılı bir şekilde dolandırılması için çalışanlar arasında işbirliği yapılabileceği için böyle bir önlemin alındığını düşündüm. Bu olay, her bir örnekte nöronların farklı bir altkümesinin rasgele bir şekilde sıfırlanması komploları önleyeceğini ve böylece modelin ezberlemesini azaltacağını anlamamı sağladı.” Buradaki temel fikir, zayıf bilgilerin unutulması modelin ezberlemesine engel olmasıdır [52].

1.3.9. Konvolüsyon sinir ağlarının eğitimi

KSA diğer normal sinir ağları gibi geriye yayılım (backpropagation) algoritmasının yardımıyla eğitilir. Genelde eğitim süreci basit olarak altı adımdan oluşmaktadır;

- İlk aşamada, KSA katmanları, giriş görüntüsünün boyutları, konvolüsyon katmanındaki filtrelerin sayısı, hata fonksiyonu, optimizasyon algoritması ve modelin performansını ölçmek için metrikler seçilerek mimari oluşturulur.
- İkinci aşamada, filtreler ve tam bağlı katmanların ağırlıkları rasgele değerler ile başlatılır.

- Üçüncü adımda, KSA eğitim görüntüsü ile beslenir ve ileri yayılma algoritmasının yardımıyla konvolüsyon, ReLU, pooling ve tam bağlı katmanlardan geçerek giriş görüntüsünün önceden belirlenmiş sınıflara göre olasıklarını bulur.
- Dördüncü adımda, en son katmandaki olasılıkların gerçek cevaplar arasındaki hata hesaplanır.
- Beşinci adımda, KSA'da tüm ağırlıklara göre dördüncü adımda hesaplanan hatanın gradyanları geriye yayılım algoritmasının yardımıyla hesaplanır ve hatayı en aza indirmek için tüm filtre değerlerini ve tam bağlı katmanların ağırlıkları gradient descent algoritmasının yardımıyla güncellenir.
- Son adımda, üçüncü ve beşinci adımlar tüm eğitim veri seti için tekrarlanır. Tüm eğitim veri setinin KSA'dan bir kere ileri yayılım ve bir kere geriye yayılım yapılması bir dönem (epoch) olarak adlandırılır.

Görüntü sınıflandırma uygulamalarında eğitilmiş modellerinin performansı eğitim veri setinin büyüklüğüne, eğitim veri setinin sınıflara göre dağılımına, modelin tasarımındaki katmanların, filtrelerin sayısına ve seyreltme gibi regülarizasyon tekniklerine bağlıdır. Katmanların, filtrelerin ve tam bağlı katmanlardaki nöron sayısı arttıkça eğitim süresi de uzar. Eğitim süresini hızlandırmak için çeşitli CUDA ve cuDNN gibi yazılımsal, Jetson TX2 gibi donanımsal çözümler bulunmaktadır [53–55]

Küçük görüntü veri setlerinde derin öğrenmeye yönelik yaygın ve oldukça etkili bir yaklaşım, önceden eğitilmiş modelleri (pretrained model) kullanmaktır. Önceden eğitilmiş model, önceden büyük ölçekli bir görüntü sınıflandırma probleminde büyük veri seti kullanılarak eğitilmiş modeldir.

Günümüzde makine öğrenmesi ve derin öğrenme kütüphaneleri ve donanımsal geliştirme platformları bilişim firmaları, üniversiteler ve laboratuvarlar tarafından açık kaynak kodlu olrak geliştirilmektedir. Örneğin, Tensorflow, Theano, PyTorch, Caffe ve Keras gibi kütüphanelerin yardımıyla MNIST, CIFAR-10 veri setlerini kullanarak kısa zaman aralığında makine öğrenmesi modeli geliştirebilir [56–60].

Büyük ölçekli derin öğrenme modelleri eğitirken hesaplama kabiliyeti büyük ekran kartlarına ihtiyaç duyulmaktadır. Ekran kartları pahalı olduğundan dolayı derin

öğrenme modellerini buluttarda eğitmek ya da dağıtık eğitim framework'leri geliştirilmektedir. Keras, Tensorflow veya PyTorch kütüphanelerini birden fazla ekran kartları üzerinde çalıştırabilen "horovod" gibi dağıtık hesaplama framework'leri ortya çıkmaktadır [61].

1.4. Yayınla ve Kaydol Sistemleri

Yayınla/Kaydol (Publish/Subscribe) etkileşimi, dağıtık ortamda mesaj temelli iletişim birimidir. Bu iletişim türünde üreticiler (producers/publishers) bilgiyi mesaj olarak aracı (broker) gönderirler ve tüketiciler (consumers/subscribers) ise almak istediği bilgiyi aracılara kaydolarak alırlar. Yayınla/kaydol sistemi mesaj kuyruklama paradigması ile yakından ilgilidir ve mesaj temeli orta katman sisteminin bir parçasını oluşturur. Yayınla/kaydol modelinde mesajlar konulara yayınlanır. İlgili konuya kaydolan tüketici o konuya yayınlanan tüm mesajları alır. Çoğu mesajlaşma sistemleri hem pub/sub hem de kuyruklama paradigmasını API'lerinde desteklerler. Apache Kafka bunlardan birisidir [62].

1.4.1. Apache Kafka dağıtık mesajlaşma sistemi

Günümüzde sosyal ağların, akıllı telefonların ve bilişim teknolojilerin gelişmesi ile anlık veriler devasa boyutlara ulaştı. Bu verileri hızlı, sorunsuz ve ölçeklenebilir bir şekilde gerçek zamana yakın bir vakitte işlemek günümüzün problemlerinin bir tanesidir. Apache Kafka bunun gibi sorunları çözmek için Linkedin tarafından geliştirildi ve daha sonra açık kaynak kodlu olarak herkesin hizmetine sunuldu.

Apache Kafka temelde log kaydına benzer bir yapıda kayıtları tutan ve bu kayıtları diğer sistemlere mesajlaşma kuyruğu şeklinde sunan dağıtık bir sistemdir. Apache Kafka tüm kayıtlarını diske kaydediyor ve RAM'de hiçbir şey saklamıyor. Diske verileri doğrusal zamanda yazması ve okuma hızlı olduğundan dolayı Apache Kafka diğer sistemlere göre avantaj sağlamaktadır.

Apache Kafka'nın en önemli özelliklerinden bir tanesi onun dağıtık olmasıdır. Linkedin firması kullanıcılar tarafından anlık üretilen verileri gerçek zamana yakın vakitte işlemek için 1500'den fazla bilgisayarlar üzerinde Apache Kafka'nın kurulu

olduğunu ve kullandığını söylüyor [63]. Apache Kafka aşağıdaki özellikleri yerine getirmektedir.

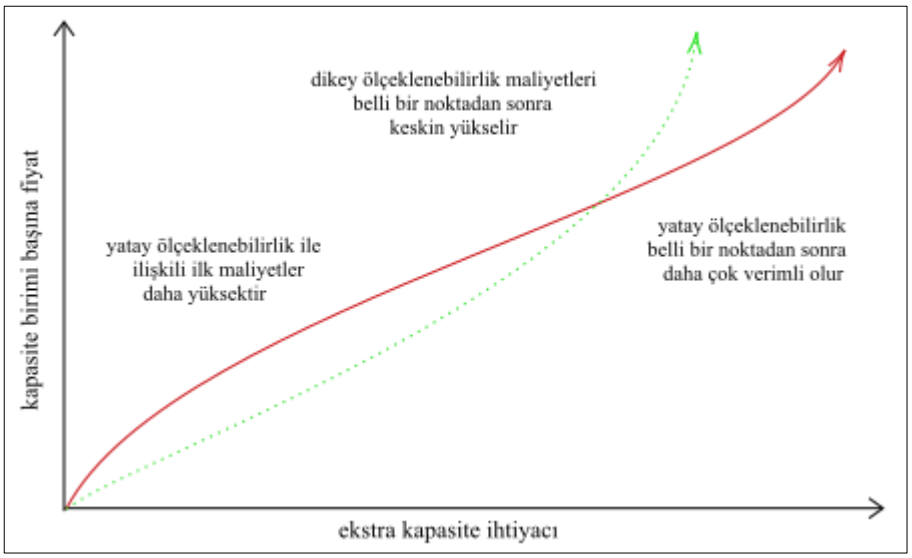
1.4.1.1. Dağıtık olma

Dağıtık sistem, birden fazla makineye bölünen ve makinelerin hepsi bir kümede birlikte çalışarak son kullanıcıya tek bir makine olarak görünen bir sistemdir. Apache Kafka farklı makinelere (broker) mesajları depoladığı, aldığı ve gönderdiği için dağıtık bir sistemdir. Dağıtık sistemdeki makineler ortak duruma (shared state) sahiptir, eş zamanlı çalışırlar ve onların birinin devre dışı kalması sistemin çalışmasını (uptime) kesintiye ugratamaz.

Geleneksel veritabanları tek bir makinenin dosya sisteminde saklanır ve her hangi bir bilgi istendiğinde makineyle doğrudan bağlantı kurulur. Geleneksel veritabanlarının dağıtık veritabanına çevirilmesi için bu veritabanlarının aynı anda birden çok makinelerde çalışması gerekir. Kullanıcı dağıtık veritabanındaki istediği bilgisayar ile bağlantı kurabilmeli ve x makinesine eklediği kayıtları y makinesinden çekebilmeli.

1.4.1.2. Ölçeklenebilirlik

Kullanıcıların sayısı arttıkça sistemin çalışması ve cevap vermesi yavaşlar. Bunun gibi durumlarda sistemin ölçeklenebilir olması önemli noktalardan bir tanesi. Ölçeklenebilirliğin yatay ve dikey olmak üzere iki tane çeşidi vardır. Sistemin yatayda ölçeklenebilir olması, bir tane çok güçlü aynı zamanda pahalı bir donanım kullanmasıdır. Dikey ölçeklenebilir sistemlerde donanım kısıtlarıyla karşı karşıya kalmak ve donanımları güncellerken sistemin çalışmasının kesintiye uğraması en yaygın problemlerdendir. Örneğin mevcut sistemin işlemcisinin frekansını 6GHz veya RAM boyutunu 1TB yapmak mümkün değildir. Sistemin yatayda ölçeklenebilir olması, ucuz ve çok sayıda makinelerin aynı anda kullanılması anlamına gelir. Yatay ölçeklenebilirlikte yüzlerce, binlerce makinelik sunucu ağı kurulabilir. Şekil 1.12'de gösterildiği gibi belli bir noktadan sonra yatay ölçeklenebilirlik daha verimli olur. Yatay ölçeklenebilirliğin yönetimi dikey ölçeklenebilirliğe göre daha zordur.



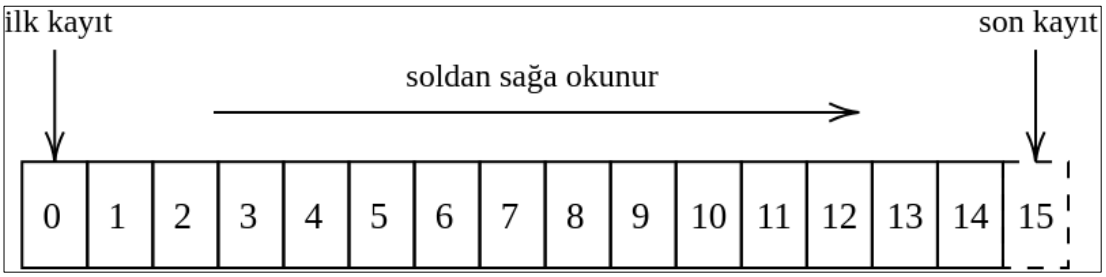
Şekil 1.12. Yatay ve dikey ölçeklenebilirliklerin farkı

1.4.1.3. Hata toleransı (Fault tolerance)

Hata toleransı, bir sistemin bileşenlerinden birisinin tamamen arızalanarak işlemez hale gelmesi ya da bileşenin içinde bir ya da daha fazla arıza oluşması durumunda, sistemin düzgün bir şekilde işlemeye devam edebilme özelliğidir. Dağıtık sistemler, her hangi bir arıza ortaya çıktığında onları düzeltebilcek şekilde tasarlanırlar. Örneğin, altı düğümlü (node) Apache Kafka kümesinde (cluster), düğümlerden iki tanesi arızalandığı durumunda bile çalışmaya devam edebilir. Hata toleransı sistemin performansına doğrudan ilişkili oluğuna dikkat edilmelidir, sistemdeki hata toleransı arttıkça, sistemin performansı düşer.

1.4.1.4. Commit log

Commit log, sadece ve sadece ekleme işlemini destekleyen kayıtlı sıralı veri yapısıdır. Kayıtlar üzerinde silme veya yerini değiştirme gibi işlemler uygulanamaz. Şekil 1.13'te gösterildiği gibi soldan sağa doğru okunur ve kayıtların sırasını garanti altına alır. Her bir kayıt için eşi benzeri olmayan ardışık kayıt numarası atanır. Kayıtların sıralanması, soldaki kayıtlar daha eski ve sağdaki kayıtlar daha yeni olduğundan dolayı burda zaman söz konusu olduğunu gösterir ve kaydın numarası tarih bilgisi/damgası (timestamp) olarak da düşünülebilir.



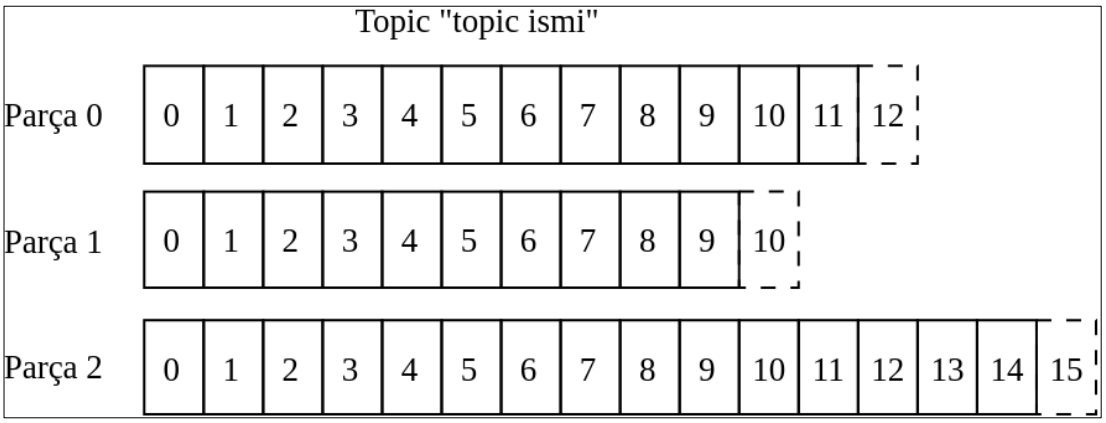
Şekil 1.13. Commit Log veri yapısının örnek gösterimi

Kafka, tüm mesajları (kayıtları) sırasıyla diske kaydeder ve onları okurken de sıralı bir şekilde okuma avantajını sağlar. Kafka mesajları okuma ve yazma işlemlerini sabit O(1) zamanda yaparken diğer veri yapıları diskten okuma ve diske yazma işlemlerini O(logN) zamanda yapmaktadırlar. Okuma ve yazma işlemleri bir birlerinden bağımsız olarak gerçekleştirilir. Yazma işlemi dengeli ağaç veri yapılarındaki gibi okuma işlemini engellemez. Burda önemle vurgulanacak nokta Kafka mimamirisinde veri boyutu performanstan tamamen ayrılmıştır ve Kafka sunucusunda 100KB veya 100TB veriye sahip olmasına rağmen aynı performansı yakalar.

1.4.1.5. Konu (topic) yapısı

Konu, kayıtların veya mesajların yayınlandığı bir kategoridir. Apache Kafka'da konular her zaman çok aboneli olup, kendisine yazılan verilere sıfır, bir ve birden çok tüketicilere (consumer) sahip olabilirler.

Konular oldukça büyük olabileceğinden daha iyi performans ve ölçeklenebilirlik yakalaması için konular Şekil 1.14'te gösterimiş gibi küçük parçalara (partitions) ayırırlar ve parçalara hangi kayıtların tutulacağını üretici yönetebilir. Örneğin, kullanıcıların her hangi bir sisteme giriş bilgilerini kullanıcının ismindeki ilk harfe göre farklı parçalarda tutulabilir. Kafka, bir parça içindeki tüm mesajların, geldikleri sırada sıralanmasını garanti eder; ancak parçalar arasında global sıralama yoktur.



Şekil 1.14. Üç parçaya ayırılmış topic

Kafka'nın diğer bir özelliği ise konu parçalarının replikalar üzerinde kopyalanmasıdır. Herhanigi bir zamanda, replikalardan bir tanesi lider olarak hareket edecek ve liderde bir sorun çıkarsa replikalardan biri lider olarak devralınır.

Kafka, akılsız broker (dumb broker) ve akıllı tüketici (smart consumer) ilkesini takip ediyor. Bu Kafka'nın tüketiciler tarafından hangi mesajların okunduğunu takip etmediği ve onları silmediği; ancak belirli bir süre veya belirli bir eşik değerine ulaşana kadar sakladığı anlamına gelir. Tuketiciler Kafka'yı yeni mesaj için sorgularlar ve hangi kayıtları okumak istediklerini söylerler. Bu, onların istedikleri offset'i arttırmasına veya azaltmasına ve mesajları konudan yeniden okumasına olanak tanır.

1.4.1.6. Veri kopyalama (Data replication)

Kafka kümesindeki broker'lerin bir tanesi çöktüğü anda verileri korumak için konu parçalarının (partition) verileri birden fazla broker'ler arasında paylaşılır. Her zaman bir parçaya bir broker sahiplik eder ve bu broker sayesinde diğer uygulamalar parçadan okuma ve yazma işlemlerini yürütürler. Bu broker'e parça lideri denir ve parçadaki verileri diğer takipçi broker'ler ile paylaşır. Takipçi broker'ler ise lider broker'der aldığı verileri depolarlar ve lider broker'in çökmesi durumunda lider olarak seçilebilirler. Böyle bir mekanizmanın olması yayınlanmış mesajın kaybolmayacağı garantisini verir.

1.5. OpenCV Kütüphanesi

OpenCV (Open Source Computer Vision Library) açık kaynak kodlu, bilgisayarla görüde kullanılan görüntü işleme kütüphanesidir. Java, C++ ve Python yazılım dillerini destekler. OpenCV ayrıca NVIDIA CUDA ve OpenCL gibi donanımsal ivmelendirme kütüphanelerini de kullanabilmektedir. OpenCV'yi oluşturan önemli bileşenlerin bazıları aşagıdaki gibidir [64].

- Çekirdek (Core): boyut, nokta, dörtgen, matris gibi basit yapıları ve temel fonksiyonları içerir. Ayrıca XML/YML uzantısındaki dosyalar için işlem yapabilen gerekli bileşenleri ve kümeleme algortimalarını barındırır.
- DNA (DNN): Derin Sinir Ağları için geliştirilmiş bir birimdir. Caffe, Darknet, Tensorflow ve Torch gibi makine öğrenmesi kütüphaneleri ile eğitilmiş modelleri test etmek için kullanılır.
- Video G/Ç: Kameralara, video cihazlarına erişmek, görüntüleri okumak ve yazmak için gerekli metotları barındırır.
- HighGUI: Resimleri görüntüleme ve pencereleri yönetme için lazım olan metotları içerir.
- Makine Öğrenmesi: Karar Ağaçları, Gradient Boosting Ağaçları, En Yakın K Komşu, Naive Bayes, Yapay Sinir Ağları ve DVM algoritmalarını içermektedir.
- Takip Algoritmaları: Videolardan nesneleri takip eden MedianFlow, TLD ve KCF gibi algoritmaları içermektedir.

1.6. FFmpeg

FFmpeg, gerçek zamanlı ses ve video kaynaklarından veri okuyabilen çok hızlı bir video ve ses dönüştürücüsüdür. Ayrıca isteğe bağlı örnek değerler arasında dönüşüm yapar ve yüksek kaliteli çok fazlı filtreler yardımıyla video boyutunu anında değiştirebilir [65].

FFmpeg, herhangi bir videonun formatını ve kodeklerini (codec) hızlı değiştirebilen açık kaynak kodlu yazılımdır. FFmpeg, ses ve görüntü kodeklerini birçoğunu desteklemekle beraber HTTP (Hypertext Transfer Protocol), RTMP (Real-Time Messaging Protocol), RTSP (Real-Time Streaming Protocol), HLS (HTTP Live Streaming) gibi yayınlama protokollerini de desteklemektedir.

FFmpeg; videoları kesme, videolardan görüntü çıkarmak ve videoları protokollerin yardımı ile sunuculara yayınlamak için pratikte sıkça kullanılan bir programdır.

1.7. NGINX RTMP Birimi

NGINX, ücretsiz açık kaynaklı, yüksek performanslı bir HTTP ve ters vekil sunucusudur (reverse proxy). NGINX yüksek performansı, istikrarlılığı, geniş özellikleri, basit konfigürasyonu ve düşük kaynak tüketimi ile bilinir. Geleneksel sunuculardan farklı olarak, NGINX istekleri işlemek için iş parçalarına dayanmaz. Bunun yerine çok daha ölçeklenebilir, olaya odaklı, asenkron bir mimari kullanır. NGINX bir tane makine üzerinden yüzlerce makine kümesine (cluster) kadar her yönde ölçeklenebilir bir sunucudur [66].

NGINX, Netflix, Hulu, Pinterest, CloudFlare, Airbnb, WordPress.com, GitHub, SoundCloud, Zynga, Eventbrite, Zappos, Media Temple, Heroku, RightScale, Engine Yard ve MaxCDN gibi yüksek performanslı sitelerde kullanılmaktadır [66].

NGINX RTMP birimi, NGINX sunucusuna ek birim olarak dahil edilebilir. NGINX RTMP biriminin özellikleri arasında RTMP (Real-Time Messaging Protocol) /HLS (HTTP Live Streaming)/MPEG-DASH (Dynamic Adaptive Streaming over HTTP) protokollerini destekleyen canlı yayın, talebe bağlı video yayını (video on demand), yerel dosya sisteminden video oynatma, hızlı yayınlama ve düşük bellek alanı kullanmak ve bellek ayırmalarını minimum düzeyde tutmak için gelişmiş ara bellek bulunur [67].

1.8. Node.js Teknolojisi

Node.js, açık kaynaklı, sunucu tarafında çalışan ve ağ bağlantılı uygulamalar için geliştirilmiş bir çalıştırma ortamıdır. Node.js uygulamaları genelde istemci tarafı betik (client-side scripting) olan JavaScript programlama dili kullanılarak geliştirilir [68]. Node.js Google tarafından geliştirilmiş yüksek performanslı V8 JavaScript kodunu makine koduna dönüştüren yüksek performanslı JavaScript motoru üzerinde çalışan bir platformdur.

Node.js'in önemli özellikleri bloklamayan giriş/çıkış (non-blocking IO) ve asenkron yapısı yüksek performanslı ve ölçeklenebilir uygulamalar geliştirmeye yardımcı

olurlar. Veri tabanından veri okuma, dış servislere veri aktarma veya dosyadan bilgi okuma sırasında bekleme yapmaz. İşlem sonuçlarını geri dönüş (callback) metodlarının yardımıyla elde eder.

Node.js'in gittikçe popüleritesi artmakla beraber yeni özellikler eklenerek geliştirilmektedir. Linkedin ve Paypal gibi büyük firmalar uygulamalarını Node.js'i kullanarak yeniden geliştirdiklerini, uygulamaların performanslarının arttığını ve sunuculara olan maliyetin azaldıklarını pratikte göstermişlerdir [69-70].

1.9. WebSocket

Sunucularda herhangi bir değişiklik olduğunda sunucu bu değişiklikleri istemciye bildiremez. Değişiklikleri istemciye iletebilmek için iki farklı yöntem kullanılır.

Birincisi, istemci sunucuya istek göndererek sunucudaki yeni değişiklikleri alır. Bu istek-yanıt ilişksi iki şekilde gerçekleşir. Bu istek-yanıt ilişkilerinden biri polling olarak adlandırılır ve istemcinin belli zaman aralıklarıyla sunucuya istek yapmasıdır. Sunucu her istek için istemciye ayrı ayrı yanıt verir. Diğeri ise long polling, istemci sunucuya istek gönderdiğinde sunucu o istekle alakalı herhangi bir bilgi içermiyorsa, boş yanıt göndermek yerine yeni bir bilgi gelene kadar sunucunun istemciyi bekletmesidir.

İkincisi, istemciden sunucuya ve sunucudan istemciye iki taraflı (full-duplex) bağlantı yardımıyla bilgilerin aktarılması. Diğer adıyla websocket olarak bilinir ve HTTP protokolüne uygun olmayan gerçek zamanlı web uygulamalarında kullanılır.

Websocket bağlantı oluşturmak için istemci sunucuya istek gönderir, sonra sunucu kabul edip, websocket üzerinden istemci için özel bir bağlantı açarak istemci ve sunucu arasında iki taraflı bir bağlantı oluşturur. Sonrasında, istemci sürekli olarak sunucuya istek göndermez ve sunucu gerektiği zaman istemciye bilgileri gönderebilir.

1.9.1. Socket.IO gerçek zamanlı motoru

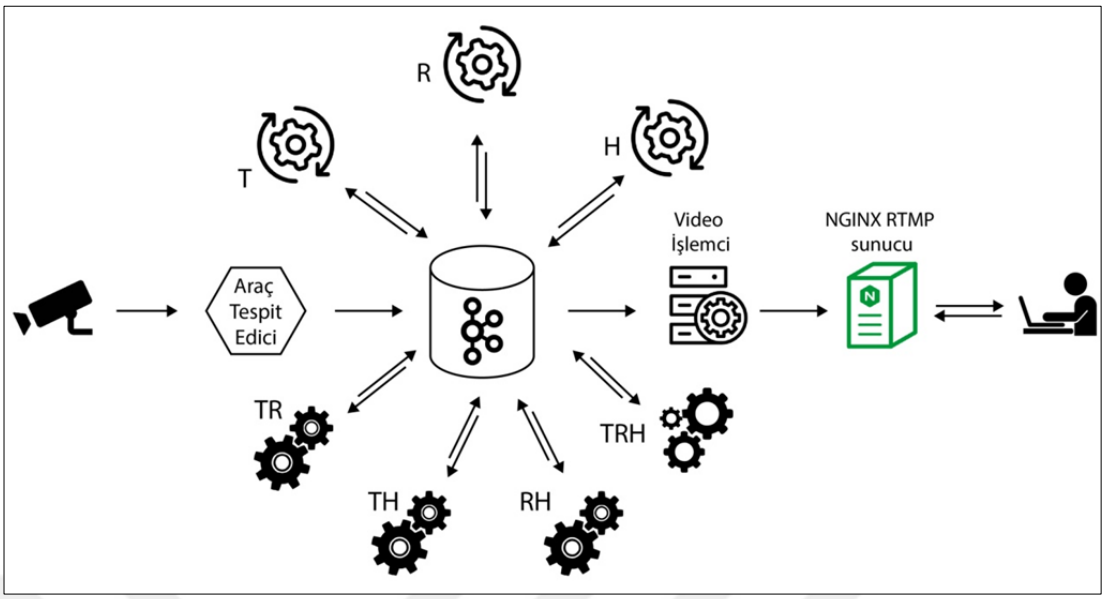
Socket.IO, gerçek zamanlı web uygulamaları için bir Node.js birimidir. Web istemcileri ve sunucuları arasında gerçek zamanlı, iki taraflı iletişim sağlar [71]. Socket.IO WebSsocket iletişimi kurmak için aşağıdaki iki parçaya ihtiya duyar:

- Birincisi, WebSocket desteği olan bir istemci. Örnek olarak Google Chrome tarayıcısı söylenebilir.

- İkincisi, WebSocket desteği olan bir sunucu. Örnek olarak Node.js sunucusu söylenebilir.

2. ÖNERİLEN YÖNTEM

Geliştirilen mimari, Şekil 2.1'de gösterildiği gibi dört ana kısımdan oluşmaktadır: (i) araçların tiplerine göre sınıflandırılması (oto, otobüs, minibüs, minivan ve kamyon), (ii) araçların renklerine göre sınıflandırılması (beyaz, yeşil, mavi, kırmızı, siyah, gri ve sarı), (iii) araçların hızlarının tespit edilmesi (30 km/saatten küçük, 30-40, 40-50, 50-60, 60-70, 70-80, 80-90 ve 90 km/saatten fazla) ve (iv) bu üç kategoriye göre sınıflandırılmış görüntüleri yayınla/kaydol (publish/subscribe) teknolojilerini kullanarak sisteme abone olan son kullanıcılara video şeklinde gerçek zamana yakın bir vakitte aktarılması. İlk adımda, araçların koordinatlarını tespit eden programlar kendilerine atanmış kameralardan veya önceden çekilmiş video dosyalarından videoları çerçeveler şeklinde (frame olarak) okuyup ve çerçeve içindeki araçların koordinatlarını tespit edip Apache Kafka yardımıyla frame0 konusuna araçların koordinat bilgilerini yollarlar. İkinci adımda, araçları tipine göre sınıflandıran program (T) frame0 konusundan araçların koordinat bilgilerini okuyup araç tiplerine göre sınıflandırır ve sonuçları oto, otobüs, minibüs, minivan ve kamyon olmak üzere farklı farklı konulara yayınlar. Araçları rengine göre sınıflandıran program (R) ise frame0 konusundan bilgileri okuyup araçları rengine göre sınıflandırır ve beyaz, yeşil, mavi, kırmızı, siyah, gri ve sarı olmak üzere yedi tane konuya işlem sonuçlarını yayınlar. Araç hızını tespit eden program ise araçların hızlarını (H) tespit eder ve işlem sonuçlarını sekiz tane konuya yayınlar. Üçüncü adımda, son kullanıcılar araçları tipine ve rengine (TR), tipine ve hızına (TH), rengine ve hızına (RH) veya tipine, rengine ve hızına (TRH) göre sınıflandırabilmeleri için birleştirme işlemleri gerçekleştirir. Son adımda, son kullanıcıların sorgularına göre sınıflandırma ve birleştirme sonuçlarında ortaya çıkan video parçaları son kullanıcılara RTMP (Real-Time Messaging Protocol) protokolü video akışı olarak gönderilir.



Şekil 2.1. Sistemin genel mimarisi

2.1. HOG Algoritması ile Öznitelik Çıkarımı

Veri seti hazırlama sırasında Şekil 2.2'de gösterildiği gibi görüntüden araç kare boyutunda (128x128) çıkartılıp ve ondan sonra görüntünün HOG öznitelik vektörü hesaplanmıştır. Veri setindeki görüntülerin boyutları 128x128 olarak seçilmiştir. Kare olmasının nedeni ise araçların çoğu zaman ön taraftan görünümü kareye benzer olmasındandır. Hücre boyutu 8x8, blok boyutu 16x16 ve blokların kaydırma uzunluğu 8 olarak seçilmiştir. Görüntülerin boyutları arttıkça ve HOG öznitelik vektörünün boyutu da artar ve hesaplama performansını kötü yönde etkileyebilir.

Şekil 2.2. HOG öznitelik vektörünün çıkartılması

2.2. Araç Tespit Etme Yöntemi

Görüntüden nesne algılama gibi bilgisayarla görü (computer vision) uygulamalarında, aranan nesnenin mevcutluk durumunun yanında lokasyon bilgileri de önemlidir. Kayan pencere (sliding window) yaklaşımı kullanılarak, aranan

nesneye ait lokasyon bilgisi kolaylıkla edilebilir. Şekil 2.3'te gösterildiği gibi kayan pencere yöntemindeki temel mantık, giriş görüntüsünü sabit boyutlu bir dörtgen pencere kullanarak görüntünün sol üst tarafından başlayarak sağ alt tarafına ulaşana kadar sağa kaydırmak ve kullanılan pencerenin aranan nesneyi içerip içermediğine karar vermektir. Genelde karar verme işlemi makine öğrenme modeli kullanılarak aranan nesne için eğitilmiş bir sınıflandırıcı uygulayarak gerçekleştirilir.

Araç algılama, verilen görüntüde araç olup olmadığına karar verme ve eğer araç mevcut ise, aracın kestirilen lokasyonunu sunma işlemidir. Sabit boyutlu pencereyi görüntü üzerinde bütün olası lokasyonlara kaydırarak ve kaydırılan bu pencerenin içerdiği görüntüyü bir ikili (binary) sınıflandırıcıdan geçirerek aranan nesnenin konum bilgisini elde etmek mümkündür. Bu çalışma çerçevesinde DVM yöntemini HOG öznitelik vektörleri ile eğiterek ikili sınıflandırıcı modeli tasarlanmıştır. İkili sınıflandırıcı için eğitim aşamasında araç içeren ve içermeyen görüntülerin HOG öznitelikleri çıkartılarak ikili sınıflandırıcı model eğitilmiştir.

Şekil 2.3. Kayan pencere tekniği uygulanarak araç algılama

Kayan pencere yöntemiyle ilgili olan sorunlardan bir tanesi bu yöntemin çok hızlı olmamasıdır. Ayrıca bu teknik kullanılarak tespit edilen bölgeler aranan nesnenin yanında büyük ölçüde arka plan görüntüsü de içerebilmektedir ve bu durum tespit işleminden sonra gerçekleştirebilecek nesne tanıma gibi işlemlerin performansını kötü yönde etkileyebilir. Hız problemin üstesinden gelebilmek için etkin bir şekilde GPU üzerinde çalışan HOG tanımlayıcıları geliştirilmiştir. Bilgisayarla görü uygulamalarında en çok kullanılan OpenCV kütüphanesinde HOG tanımlayıcılarını hızlı bir şekilde hesaplamak amacıyla GPU üzerinde çalışan cuda::HOG veri yapısı geliştirilmiştir [72].

2.3. Araçları Tiplerine göre Sınıflandırma

Araçları tipine göre sınıflandırma, tespit edilmiş aracın önceden tanımlanmış sınıflara göre sınıflandırılmasıdır. Tasarlanmış sistemde oto, kamyon, otobüs, minibüs ve minivan olmak üzere önceden tanımlanmış beş araç tipi mevcuttur ve amaç tespit algoritması tarafından tespit edilen aracın bu beş sınıfa göre sınıflandırılmasıdır. Bu çalışmada iki farklı yaklaşım ile sınıflandırma işlemi gerçekleştirilmiştir: DVM ve KSA.

Karmaşıklık matrisi, gerçek ve tahmin edilen değerleri görüntüleyerek bir sınıflandırma modelinin performansını görselleştiren tablo düzenidir. Şekil 2.4'te ikili sınıflandırıcı için karmaşıklık matrisi gösterilmiştir.

		Kestirilen	
		Pozitif	Negatif
Gerçek	Pozitif	TP	FN
	Negatif	FP	TN

Şekil 2.4. Karmaşıklık matrisi

Çok sınıflı sınıflandırıcılarda model performansını ölçerken doğruluk (accuracy) yetersizdir. Örneklerin sınıflar arasında doğru dağılmadığı sürece doğruluk metriği küçük sınıflar için duyarsız kaldığı için modelin performans ölçüleri hassaslık (recall/sensitivity), kesinlik (precision) metrikleri Şekil 2.4'te gösterilmiş karmaşıklık matrisini kullanılarak değerlendirilebilir. Formüller aşağıda açıklanmıştır.

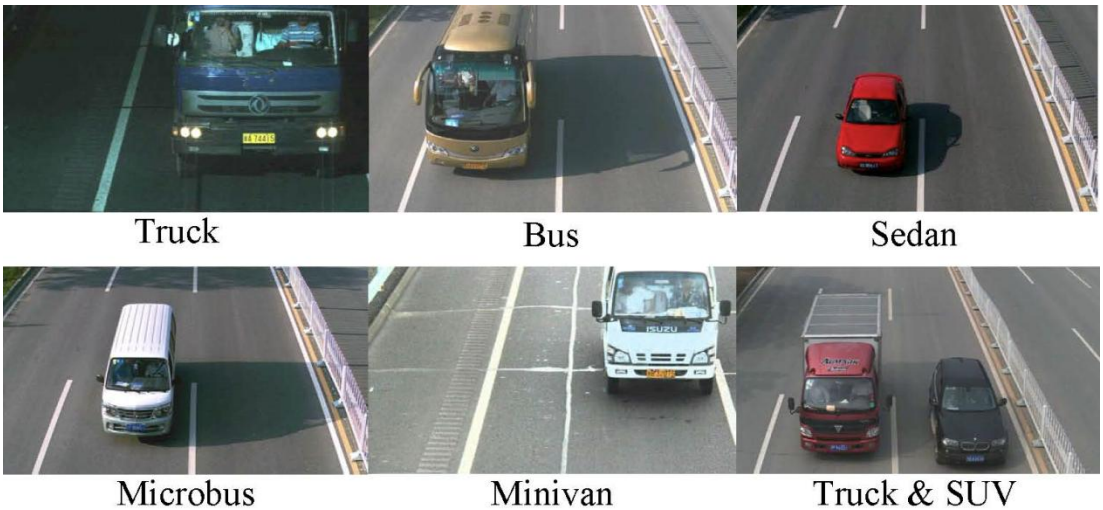
$$\text{Hassaslık} = \frac{TP}{TP+FN} \tag{2.1}$$

$$\text{Kesinlik} = \frac{TP}{TP+FP} \tag{2.2}$$

Denklem (2.1) - (2.2) TP (true positive) gerçek pozitiftir, TN (true negative) gerçek negatiftir, FN (fasle negative) yanlış negatiftir ve FP (fasle positive) yanlış pozitiftir. TP araç tipinin doğru sınıflandırıldığı durumlar, FP ve FN araç tiplerinin yanlış sınıflandırıldığı durumlardır. Hassaslık, test veri setindeki pozitif örneklerin doğru

olarak sınıflandırılmış oranıdır. Kesinlik, pozitif olarak sınıflandırılan veri setindeki örneklerin doğruluk oranıdır. Tablo 3.3‘te Tiny-YOLOv2 modelinin BIT-Vehicle veri setindeki performans sonuçları gösterilmektedir. Tablo 2.4‘te Tiny-YOLOv2 modelinin TPS veri setindeki performans sonuçları gösterilmektedir [73].

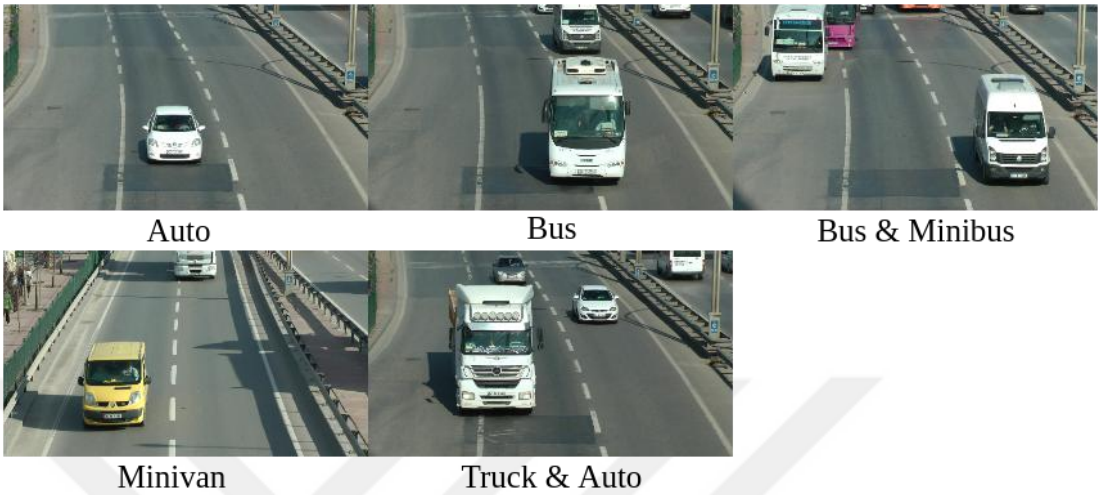
Deneylerimizde Beijing Institute of Technology Üniversitesi Media Computing and Intelligent Systems (Medya Hesaplama ve Akıllı Sistemler) Laboratuvarı çalışanları tarafından oluşturulan ve akademik çalışmalar için paylaşılan “BIT-Vehicle Dataset” veri seti de kullanılmıştır. “BIT-Vehicle Dataset” veri seti toplamda 9850 araç görüntüsü içermektedir. Görüntüler farklı zaman aralıklarında, 1600x1200 ve 1920x1080 olmak üzere iki farklı çözünürlükteler. Araç görüntüleri; farklı aydınlanma koşullarında, farklı yüzey renklerinde ve farklı bakış açılarında elde edilmiştir. Bazı araçların üst veya alt kısımları, yakalama gecikmesi ve aracın boyutu nedeniyle görüntülere dahil değildir. Bir resimde bir veya iki araç olabilir, bu nedenle her aracın görüntüdeki lokasayon bilgileri .mat uzantılı dosyanın içinde saklanmaktadırlar. Veri setindeki tüm araçlar otobüs (558), microbus (883), minivan (476), sedan (5921), SUV(1392) ve kamyon (823) olmak üzere altı kategoriye ayrılmıştır ve toplam araç sayısı 10053’tür. Şekil 2.5‘te BIT-Vehicle veri setine ait örnek görüntüler içermektedir.



Şekil 2.5. BIT-Vehicle veri setindeki araç tipleri

DVM deneylerimizde diğer bir veri seti TPSdataset2 de kullanılmaktadır. Veri seti toplamda 5196 araç görüntüsü içermektedir. Görüntüler farklı zamanlarda, 1920x1080 çözünürlükte çekilerek toplanmıştır. Veri setindeki tüm araçlar

otobüs(722), oto (568), minibüs (297), minivan (540), ve kamyon (3069) olmak üzere altı kategoriye ayrılmıştır ve toplam araç sayısı 5196'dır. Şekil 2.6'da TPSdatset1 veri setine ait örnek görüntüler içermektedir.



Şekil 2.6 TPSdataset1 veri setindeki araç tipleri

2.3.1. Destek vektör makineleri ile araç tipi sınıflandırma

Araçları tipine göre sınıflandırma uygulamasında DVM algoritmasını girdi ile beslememiz gerekir. Üzerinde durduğumuz uygulamada girdi olarak görüntülerin HOG öznitelik vektörleri kullanılmıştır.

DVM iki sınıftan oluşan veri setlerini birbirinden ayırması için tasarlanmış bir yöntemdir. Pratikte ikiden fazla etiketleri olan veri kümeleri için de kullanılır ve iki çeşit yaklaşımı vardır: bir karşı diğerleri (one-vs-rest) ve bir karşı bir (one-vs-one). Bir karşı diğerleri yaklaşımında, N tane sınıf var ise hepsine birer toplamda N tane sınıflandırıcı eğitilir. Sınıf i için i sınıfındaki tüm örnekler pozitif ve diğer kalan sınıfların örnekleri negatif olarak etiketlenir. Bu yöntem sınıfların arasındaki dengesiz örneklerin dağılımına yol açabilir. Diğer bir karşı bir yaklaşımında, her bir sınıf çiftleri için ayrı bir sınıflandırıcı eğitilir. Bu yöntem, N(N–1)/2 tane sınıflandırıcı eğitmenize yol açar ve sınıfların arasındaki örneklerin dengesizliğine daha az duyarlıdır; ancak daha çok zaman alır.

DVM yöntemi ile geliştirilen araç tipi sınıflandırıcı modellerin eğitim ve test sırasında BIT-Vehicle ve TPS veri setleri kullanılmıştır [14,73]. Test aşamasındaki moderllerin performans sonuçları Tablo 2.1 ve Tablo 2.2'de gösterilmiştir.

Tablo 2.1. BIT-Vehicle Veri Setinde DVM performansı

Araç Tipi	Eğitim/Test	Kesinlik, (%)	Hassaslık, (%)
Bus	553 (415/138)	98.5507	98.5507
Microbus	878 (659/219)	91.7431	91.3242
Minivan	474 (354/118)	87.7358	78.8136
SUV	1381 (1032/343)	89.7898	87.172
Sedan	5796 (4310/1436)	97.5762	98.1198
Truck	821 (615/205)	90	96.5854
Ortalama	9905 (7385/2459)	92.5659	91.7609

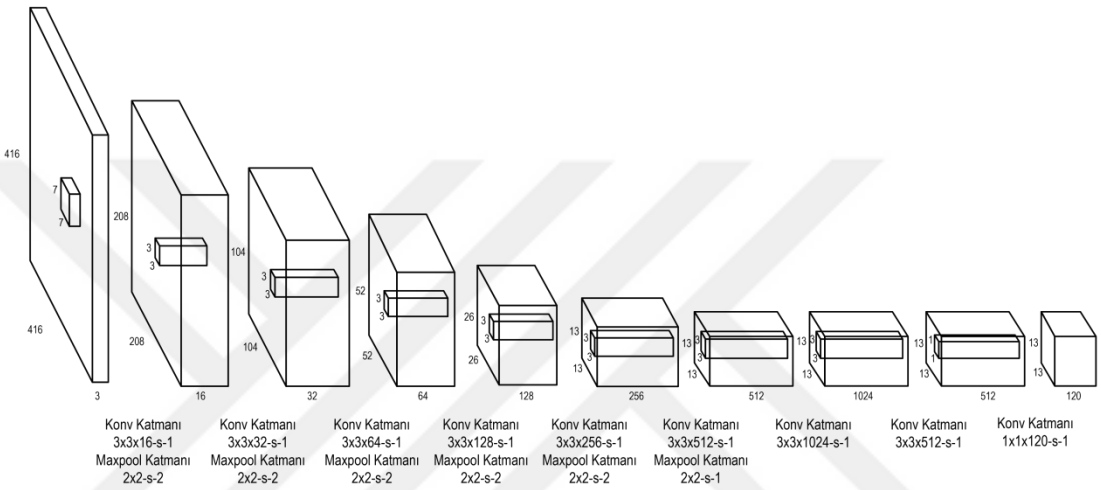
Tablo 2.2. TPSdataset2 Veri Setinde DVM performansı

Araç Tipi	Eğitim/Test	Kesinlik, (%)	Hassaslık, (%)
Auto	147 (111/36)	100	100
Bus	187 (141/46)	91.8367	97.8261
Minibus	72 (54/18)	94.4444	94.4444
Minivan	266 (200/66)	100	98.4848
Truck	719 (540/179)	99.435	98.324
Ortalama	1391 (1046/345)	97.1432	97.8159

2.3.2. Konvolüsyonel Sinir Ağlarıyla Araç Tipi Sınıflandırma

Günümüzde Konvolüsyonel Sinir Ağları, sınıflandırma ve algılama uygulamalarının ayrılmaz parçası haline gelmişlerdir. Konvolüsyonel Sinir Ağlarının pratikte kullanılmamasının temel nedenlerinden biri, onların güçlü hesaplamaları yapabilen işlemcilere ihtiyaç duymalarıydı. Ekran kartlarının gelişmesi ile Krizhevsky ve arkadaşları 2012 yılında gerçekleştirilmiş ILSVRC-2012 yarışmasını KSA yardımı ile sınıflandırıcı bir model geliştirerek kazanmışlardır. İlerleyen zamanlarda R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN ve YOLO olmak üzere kendinden önceki modelin performansını iyileştiren, çeşitli algılama ve sınıflandırma modelleri geliştirilmiştir [74-78].

Tez kapsamında geliştirilmiş uygulamada, gerçek zamanda sınıflandırma ve algılama modeli olan Tiny-YOLOv2 kullanılmıştır. Şekil 2.7‘de gösterildiği gibi, Tiny-YOLOv2 9 konvolüsyonel ve 6 maxpooling katmanlarından oluşmaktadır. Az sayıdaki katmanlar sayesinde gerçek zamanda sınıflandırma ve algılama işlemlerini yapabilmektedir. Giriş olarak 416x416 boyutundaki görüntüler kullanılır. Çıkış olarak görüntünün hangi sınıfa ait olduğu bilgisiyle beraber lokasyon bilgilerini de kestirebilir.



Şekil 2.7. Tiny-YOLOv2 Konvolüsyonel Sinir Ağının mimarisi

Deneylerimizde, Tiny-YOLOv2 modeli Şekil 2.5 ve Şekil 2.6‘da gösterildiği üzere iki çeşitli veri seti üzerinde NVIDIA GeForce 940MX ekran kartı kullanılarak eğitilmiştir. Eğitim sonuçları da Tablo 2.3 ve Tablo 2.4‘te gösterilmiştir.

Tablo 2.3. BIT-Vehicle Veri Setinde Tiny-YOLOv2 performansı

Araç Tipi	Eğitim/Test	Kesinlik, (%)	Hassaslık, (%)
Bus	558 (446/112)	100	100
Microbus	883 (706/177)	96,67	98,31
Minivan	476 (381/95)	98,95	98,95
SUV	1392 (1114/278)	97,19	99,64
Sedan	5921 (4737/1184)	97,20	99,75
Truck	823 (658/165)	87,41	100
Ortalama	10053 (8042/2011)	97,90	99,60

Tablo 2.4. TPSdataset2 Veri Setinde Tiny-YOLOv2 performansı

Araç Tipi	Eğitim/Test	Kesinlik, (%)	Hassaslık, (%)
Auto	568 (483/85)	40,00	75,29
Bus	722 (639/83)	48,98	86,75
Minibus	297 (237/60)	36,27	61,67
Minivan	540 (451/89)	44,74	76,40
Truck	3069 (2588/481)	52,53	92,93
Ortalama	5196 (4398/798)	62,83	86,22

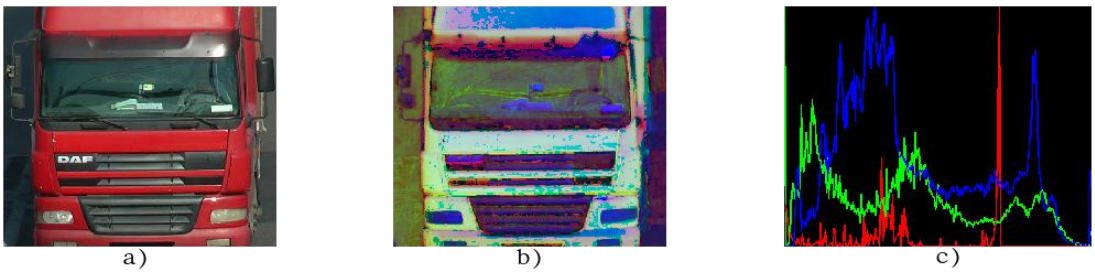
2.4. Araçları Rengine göre Sınıflandırma

Aracın rengine göre sınıflandırılması, aracın önceden belirlenmiş siyah, mavi, gri, yeşil, kırmızı, beyaz ve sarı olmak üzere yedi farklı renge göre sınıflandırmayı içermektedir. Renk bazlı sınıflandırma için DVM yöntemi kullanılmıştır. DVM giriş olarak araç görüntüsünün histogramını kullanmaktadır ve çıkış olarak 7 sınıf renginin bir tanesini kestirmektedir.

Çalışmamızda Şekil 2.8‘de gösterildiği üzere 7 farklı renge ait araç görüntüleri toplanarak TPSdataset3 veri seti oluşturulmuştur. TPSdataset3 veri seti, siyah (90), mavi (35), gri (80), yeşil (38), kırmızı (63), beyaz (119) ve sarı (24) olmak üzere toplamda 449 görüntüden oluşmaktadır.

Şekil 2.8. TPSdataset3 veri setindeki renk örnekleri

Çalışmamızda TPSdataset3 veri setindeki görüntülerin histogram vektörleri oluşturulmuştur. Görüntü histogramı, görüntüdeki piksel yoğunluklarının dağılımını temsil eden bir grafik veya çizimdir. Başka bir deyişle, X ekseninde piksel yoğunluğu ve Y ekseninde ise frekans bilgisine karşılık gelen çizimdir. İlk adımda, görüntü BGR renk uzayından Şekil 2.9(b)'de gösterildiği gibi HSV uzayına dönüştürülmüştür. İkinci adımda, veri setindeki her görüntü için Şekil 2.9(c)'de gösterildiği gibi histogramları hesaplanmıştır. Üçüncü adımda görüntülerin histogramları standart skorlama normalizasyon yönteminden yararlanarak normalleştirilerek görüntülerin öznitelik vektörleri oluşturulmuştur.



Şekil 2.9. (a) Orijinal görüntü, (b) HSV formatın dönüştürülmüş görüntü, (c) Görüntünün histogramı

Araçları rengine göre sınıflandırmak için, araç görüntülerinin histogram öznitelik vektörleri ile DVM polinom çekirdek fonksiyonu kullanılarak sınıflandırma modeli

eğitilmiştir. Tablo 2.5‘te DVM ile geliştirilen sınıflandırıcının performans sonuçları gösterilmiştir.

Tablo 2.5. Araç rengi sınıflandırıcı modelin performans sonuçları

Renk	Eğitim/Test	Hassaslık, (%)	Kesinlik, (%)
Siyah	63/27	96	93
Mavi	25/10	100	100
Gri	56/24	100	100
Yeşil	27/11	100	92
Kırmızı	45/18	100	100
Beyaz	84/35	97	97
Sarı	17/7	100	87
Ortalama	317/132	99	96

2.5. Araç Hızının Tespit Edilmesi

Gerçek zamanlı araçların hızlarının tespit edilmesi genelde donanımsal radar teknolojileri ile gerçekleştirilmektedir. Bu teknolojilerin eksiklerinden bir tanesi onların maliyetlidir. Bu nedenle son yıllarda gerçek zamanlı kameralardan alınan videolar üzerinde araç hızlarını yazılımsal olarak tespit eden algoritmalar geliştirilmiş [79-80].

Trafikte hareket eden araçların hızlarının tespit edilmesi araçların takip edilmesinden sonra yapılacak işlemlerdin bir tanesidir. 50 FPS değerine sahip kameranın yardımıyla toplam yer değiştirme üzerinden hız verisi hesaplanmıştır. Ortalama hız bilgisi toplam yer değiştirmenin toplam zamana bölünmesi denklem (2.3) ile gösterildiği gibi elde edilir.

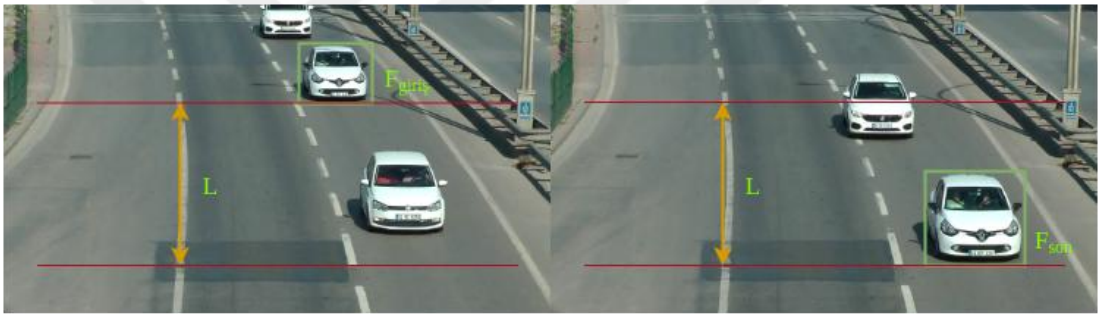
$$\upsilon_{ortalama} = \frac{s_{son} - s_{başlangıç}}{t_{son} - t_{başlangıç}} \tag{2.3}$$

Geliştirdiğimiz hız tespiti algoritmasında ilk aşamada, Şekil 2.10‘da gösterildiği üzere sabit mesafe (L) şerit çizgilerinin yardımıyla belirlenir. Türkiye’de şerit

çizgilerinin uzunluğu durak önlerinde 3 metre ve trafikte 5 metre olarak belirlenmiştir. Bu nedenle Şekil 3.10'daki sabit mesafe (L) uzunluğu 45 metre olur. İkinci aşamada, takip te olan aracın ilgi bölgesine (Şekil 2.10'daki kırmızı çizgilerin arasındaki alan) girdiği ($F_{giriş}$) ve bölğeyi terk ettiği ($F_{çıkış}$) noktalar bulunur. Üçüncü adımda, kamera FPS değeri kullanılarak aracın ilgi bölgesini geçen zaman (2.4) denklemindeki gibi bulunur. Son adımda, sabit mesafe ve zaman sayesinde (2.5) denklemindeki gibi araç hızı bulunur.

$$t=\frac{F_{çıkış}-F_{giriş}}{FPS} \tag{2.4}$$

$$\upsilon_{ortalama}=\frac{L}{t} \tag{2.5}$$



a) b)

Şekil 2.10 (a) İlgi bölgesine aracın girme, (b) ilgi bölgesinden çıkma örnekleri

Geliştirilen uygulamanın performansını test etmek amacıyla videolar Panasonic HDC-HS900 kamerası yardımıyla çekilmiştir. Uygulama yardımıyla gerçek zamanlı hız ölçümleri yapılmıştır. Alınan değerler ile gerçek değerlerin arasındaki hata payı Tablo 2.6'da gösterilmiştir. Alınan değerler ve yapılan ölçümlerin arasındaki ortalama hata payı 18% olduğu gösterilmiştir.

Tablo 2.6. Gerçek ve tespit edilen hızlar arasındaki hata payı

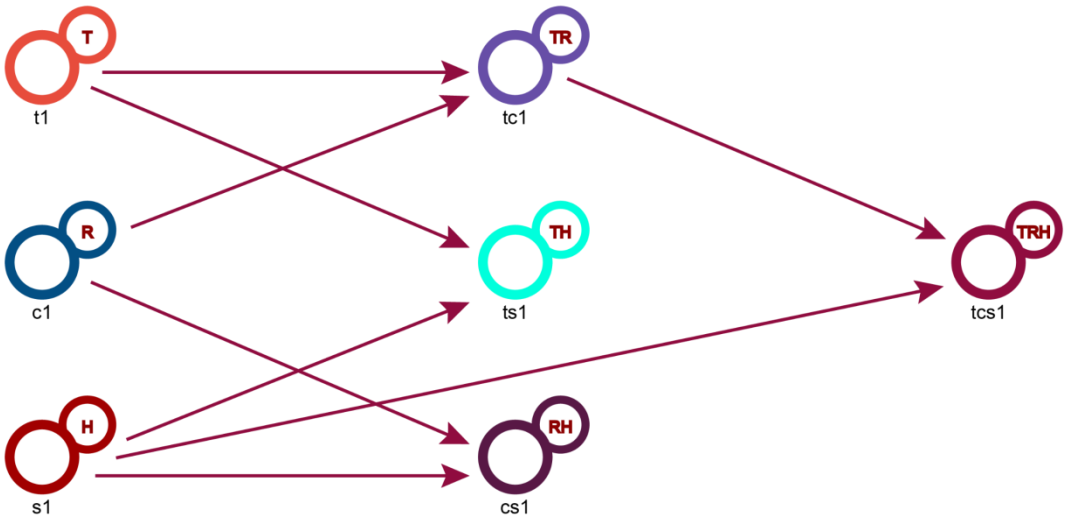
Gerçek Hız, (km/saat)	Tespit Edilen Hız, (km/s)	Hata, (%)
60	66,66 (67)	11
90	109,756 (110)	22
100	118,421 (118)	18

2.6. Araç Tipine, Rengine ve Hızına göre Görüntülerin Sınıflandırılması

Araçları tipine (T), rengine (R) ve hızına (H) göre ayıran sınıflandırıcılar elde ettikleri bilgileri Apache Kafka aracılığıyla oluşturulan ilgili konulara yayınlar. Bu konular temel konulardır. Bu konular aracın tipi için beş adet (oto, otobüs, kamyon, minivan, minibüs), rengi için yedi adet (kırmızı, yeşil, gri, siyah, beyaz, sarı, mavi), hızı için sekiz adet (<30 km/s, 30-40, 40-50, 50-60, 60-70, 70-80, 80-90, 90<) olmak üzere toplamda 20 adettir. Eğer istemci kırmızı araçları izlemek istiyorsa kırmızı konusuna, otoları izlemek istiyorsa oto konusuna abone olabilir.

2.7. Sınıflandırma Sonucundaki Görüntülerin Birleştirilmesi

Trafikteki kameralardan gelen videolar ilk olarak araçları tipine (T), rengine (R) ve hızına (H) göre sınıflandırılmaktadır. Daha sonra bu sınıflandırmadan çıkan veriler tip-renk (TR), tip-hız (TH), renk-hız (RH) olarak birleştirilmektedir. Son olarak tip-renk (TR) göre birleştiriciden gelen veriler ile hız sınıflandırıcısından (H) gelen veriler birleştirilerek tip-renk-hız (TRH) bilgileri elde edilir. Sınıflandırıcıların (T, R, H) ve birleştiricilerin (TR, TH, RH, TRH) arasındaki ilişkiler Şekil 2.11'de görsel olarak izah edilmiştir.

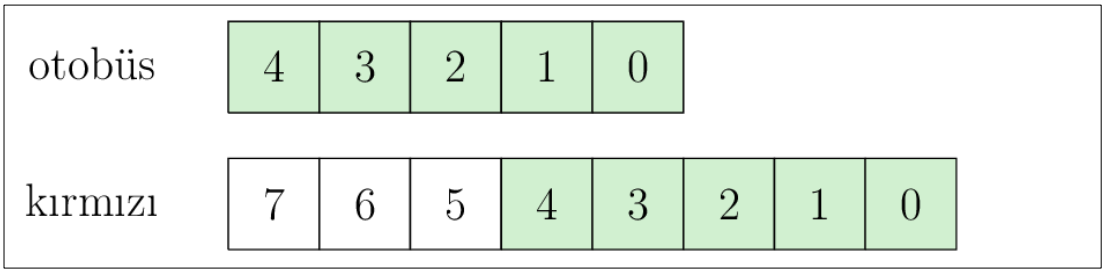


Şekil 2.11 Sınıflandırıcılar ile birleştiricilerin ilişkisi

2.7.1. Araç bilgilerinin tipine ve rengine göre birleştirilmesi (TR)

Araç tipine ve rengine göre birleştiren alt programın çalışma prensibi aşağıdaki gibidir. Araç tipinin ve renginin birleştirilmesi için öncelikle sınıflandırma işlemi ile

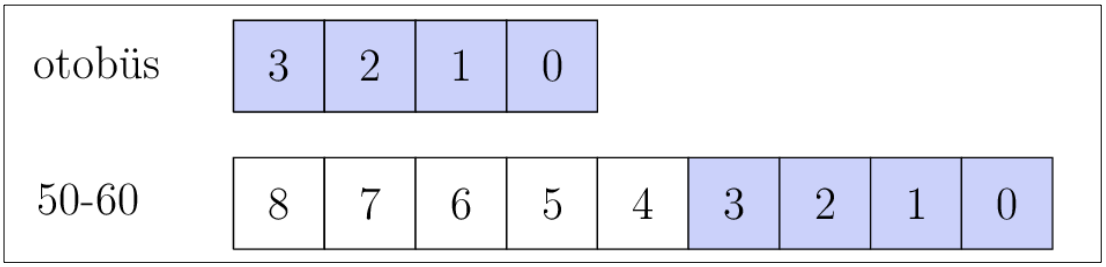
açılan beş tane araç tipi ve yedi tane renk Apache Kafka konularından gelen bilgileri Şekil 2.12'deki gibi benzersiz numaralarına göre birleştirir, tip ve renk kombınasyonlarından oluşan (oto-kırmızı, oto-yeşil, vb.) toplamda 5 x 7 = 35 konuya yayınlar.



Şekil 2.12. Sınıflandırıcılar ile birleştiricilerin ilişkisi

2.7.2. Araç bilgilerinin tipine ve hızına göre birleştirilmesi (TH)

Araç tipine ve hızına göre birleştiren alt programın çalışma prensibi aşağıdaki gibidir. Araç tipinin ve hızının birleştirilmesi için öncelikle sınıflandırma işlemi ile açılan beş tane araç tipi ve sekiz tane hız Apache Kafka konularından gelen bilgileri Şekil 2.13'teki gibi benzersiz numaralarına göre birleştirir, tip ve hız kombınasyonlarından oluşan (oto50-60, oto60-70, vb.) toplamda 5 x 8 = 40 konuya yayınlar.

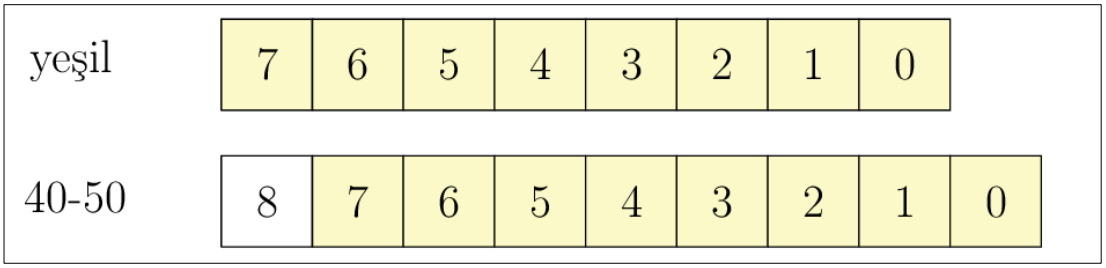


Şekil 2.13. Tip ve hız bilgilerinin birleştirilmesi

2.7.3. Araç bilgilerinin rengine ve hızına göre birleştirilmesi (RH)

Araç rengine ve hızına göre birleştiren alt programın çalışma prensibi aşağıdaki gibidir. Araç renginin ve hızının birleştirilmesi için öncelikle sınıflandırma işlemi ile açılan yedi tane araç rengi ve sekiz tane hız Apache Kafka konularından gelen bilgileri Şekil 2.14'te gösterildiği gibi benzersiz numaralarına göre birleştirir, renk ve

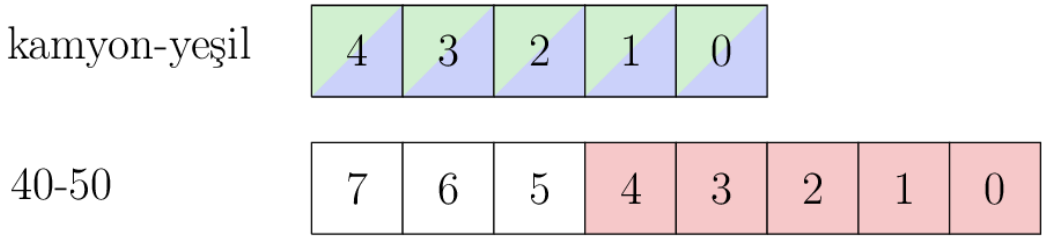
hız kombınasyonlarından oluşan (yeşil30-40, kırmızı60-70, vb.) toplamda 7 x 8 = 56 konuya yayınlar.



Şekil 2.14. Renk ve hız bilgilerinin birleştirilmesi

2.7.4. Araç bilgilerinin tipine, rengine ve hızına göre birleştirilmesi (TRH)

Araç tip-renk ve hız bilgisine göre birleştiren alt programın çalışma prensibi aşağıdaki gibidir. Araç tip-renk ve hız birleştirilmesi için öncelikle tip ve renk birleştiricisi ile açılan 35 tane tip-renk ve sekiz tane hız Apache Kafka konularından gelen bilgileri Şekil 2.15'te gösterildiği üzere benzersiz numaralarına göre birleştirir, tip-renk ve hız kombınasyonlarından oluşan (kamyonsarı50-60, otobüsbeyaz80-90, vb.) toplamda 35 x 7 = 245 konuya yayınlar.



Şekil 2.15. Tip, renk ve hız bilgilerinin birleştirilmesi

2.8. Sınıflandırma ve Birleştirme Sonuçlarının RTMP Sunucuya Gönderilmesi

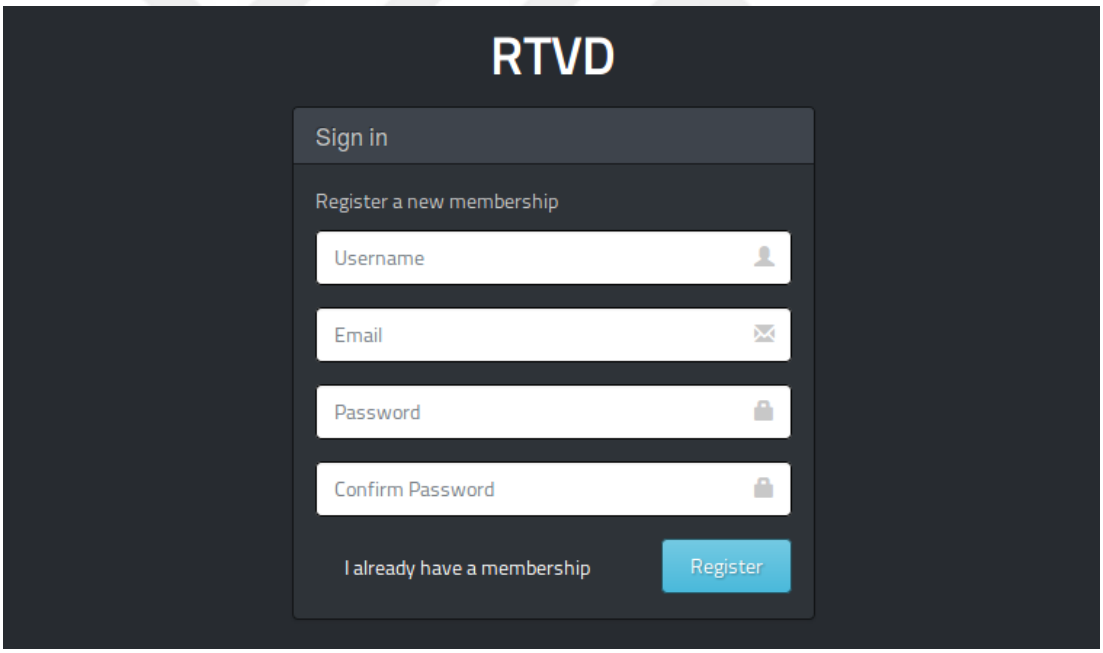
Video işleyen alt program, sınıflandırma ve birleştirme işlemlerinin sonucunda ortaya çıkan verileri konulardan okuyup, videolara çevirerek FFmpeg programının aracılığıyla NGINX RTMP video sunucusuna canlı yayın olarak gönderir. Kullanıcılar sorgularına göre kameralardan gelen görüntüleri filtreleyerek kendilerine lazım olan videolara arayüzden ulaşabilirler.

3. ARAYÜZ

Kullanıcıların programa erişimini ve programın bütününü veya herhangi bir parçasını kontrol etmesini sağlayan çeşitli resimlerin, grafiklerin, yazıların ve komutların yer aldığı sayfalardır.

3.1. Sisteme Kayıt Olma Sayfası

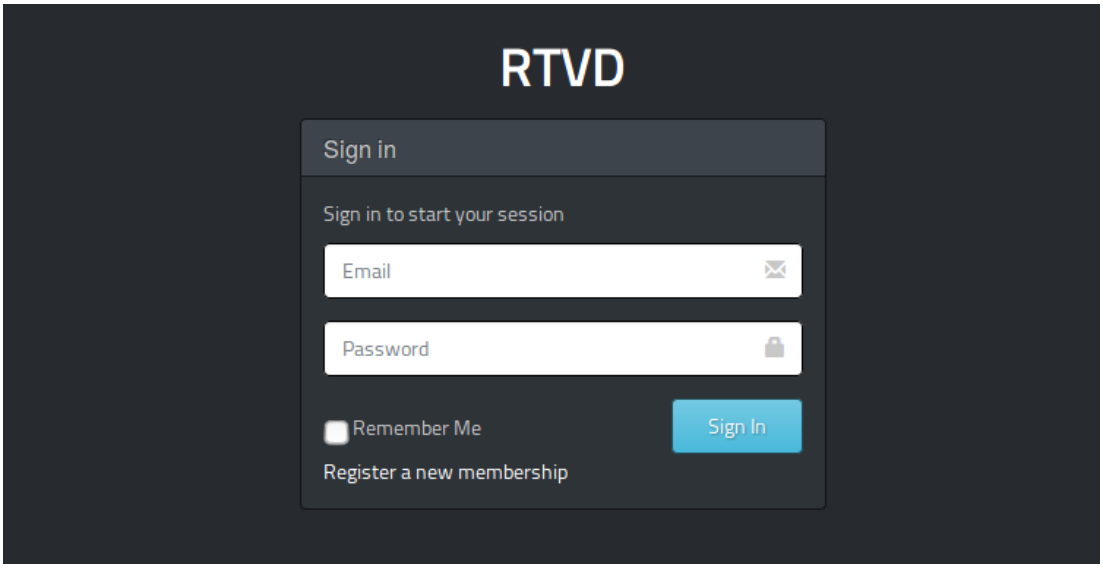
Sisteme kullanıcıları ve yöneticileri dahil etmek amacına hizmet eden arayüz sayfasıdır. Şekil 3.1'de gösterildiği üzere kayıt olmak isteyen kullanıcıdan istenen girişler kullanıcı adı, email ve kişisel şifresidir. Kayıt yapan kullanıcının sisteme erişebilmesi için başvurusunun sistemin yöneticisi tarafından onaylanması gerekir.



Şekil 3.1. Sisteme kayıt sayfası

3.2. Sisteme Giriş Yapma Sayfası

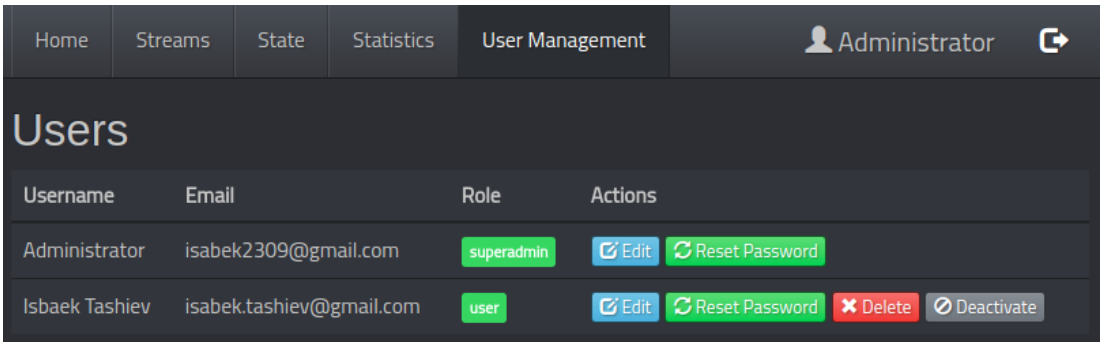
Kaydı sistemin yöneticisi tarafından onaylanmış kullanıcıların sisteme giriş yapması için kullandıkları sayfadır. Şekil 3.2'de gösterildiği gibi sisteme giriş yapmak için kullanıcının email adresi ve kayıt olurken belirlediği şifresi talep edilmektedir.



Şekil 3.2. Sisteme giriş sayfası

3.3. Sistemdeki Kullanıcıları Yönetme Sayfası

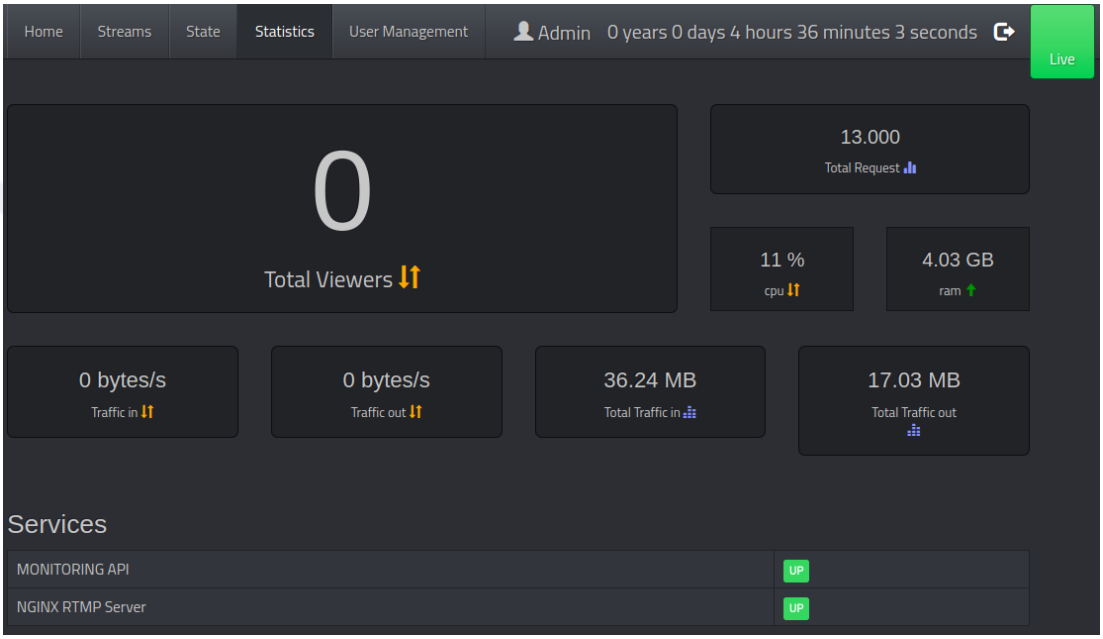
Bu sayfa kullanıcıların başvurularının değerlendirildiği ve bilgilerinin güncellendiği sayfadır. Sistemde üç çeşit kullanıcı vardır. Bunların ilki ve en yetkilisi sistem ayağa kalkarken program tasarımcısı tarafından oluşturulan "Süper Yönetici"dir. Bu yöneticinin bilgileri güncellenebilir fakat kimse sistemden silemez. Diğer kullanıcı çeşitleri ise "Yönetici" ve "Salt Kullanıcı"dır. "Süper Yönetici" diğer kullanıcıların "Yönetici" ve "Salt Kullanıcı" atamasını yaptığı gibi bilgilerini de güncelleyebilir. Kullanıcıları yönetme sayfasına sadece "Süper Yönetici" erişebilir. Şekil 3.3'te gösterildiği gibi "Süper Yönetici" bu sayfadan kullanıcıların bilgilerini güncelleyebilir, şifrelerini değiştirebilir, kullanıcıları silebilir ve devre dışı bırakabilir.



Şekil 3.3. Sistemdeki kullanıcıları yönetme sayfası

3.4. Sistem İstatistikleri Sayfası

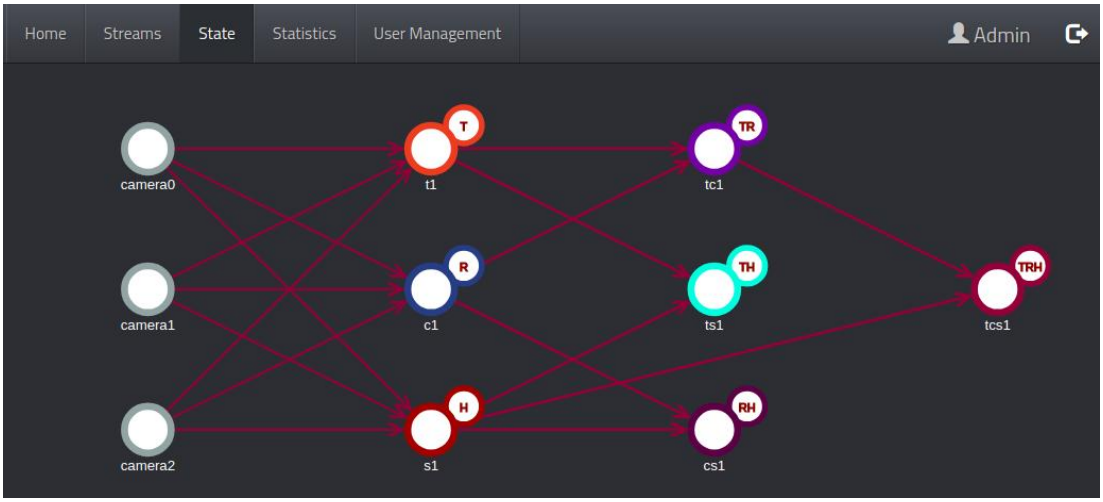
Bu sayfa NGINX RTMP biriminin istatistiklerinin tutulduğu ve sistemin sağlığı hakkında bilgi edinmek için geliştirilmiş sayfadır. Bu sayfaya süper yönetici ve yöneticiler erişebilir. Şekil 3.4'te görüldüğü gibi bu sayfada sağlanan bilgiler izleyici sayısı, donanım kullanım değerleri ve sistemin sağlık/çalışabilirliği bilgileri yer almaktadır.



Şekil 3.4. Sistemin istatistik sayfası

3.5. Sistem Durum Sayfası

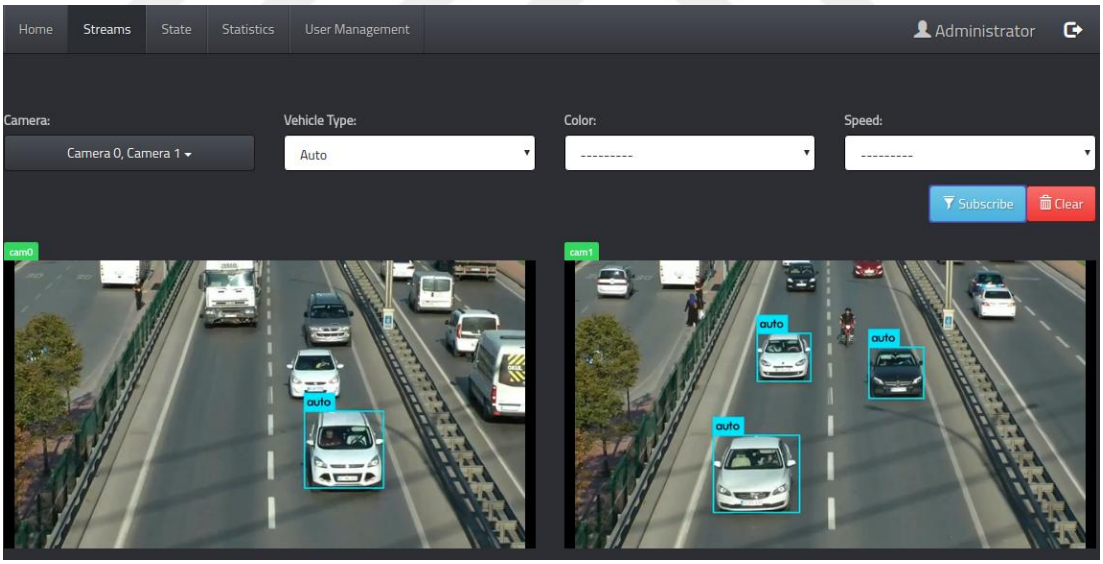
Bu sayfada kameraların, sınıflandırıcıların, birleştiricilerin ve onların arasındaki ilişkilerin durumlarının gösterildiği sayfadır. Kameradan alınan görüntüleri tespit eden, sınıflandırıcı ve birleştiricilerden gelen bildirimleri anlık olarak göstern bir graf yapısıdır. Bu sayfaya süper-yönetici ve yöneticiler erişebilmektedir.



Şekil 3.5. Sistemin durum sayfası

3.6. Araç Görüntülerini İsteğe Göre Filtreleme Sayfası

Bu sayfada kullanıcılar NGINX RTMP video sunucusundan aldıkları görüntüleri istedikleri gibi filtreleyerek izleyebilmektedirler. Bu sayfaya bütün aktif kullanıcılar erişebilir. Şekil 3.6'daki örnekte kullanıcı oto filtresi kullanmıştır.



Şekil 3.6. Araç görüntülerini isteğe göre filtreleme sayfası

4. SONUÇLAR VE ÖNERİLER

Bilindiği üzere görüntü işleme yöntemleri; askeri endüstriden güvenliğe, tıptan robotiğe, astronomiden havacılığa, biyomedikalden uzaktan algılamaya kadar birçok alanda kullanılmaktadır. Trafik ve yönetimi de bu alanlar içinde en önemlilerindendir. Bu tez kapsamında da; yazılımsal ve donanımsal yatırım yapmadan, kamera parametrelerinden (kameranın yüksekliği, yolu görüş açısı vs.), kalibrasyonundan ve otoyola ait herhangi bir bilgiden (arabaların hareket yönü, şerit çizgileri vs.) bağımsız olarak otoyolunu yukarıdan görecek şekilde yerleştirilmiş herhangi sabit bir kamera ile elde edilen araç görüntülerinin tip ve rengine göre sınıflandırılması, sınıflandırılan ve hız bilgilerine sahip görüntülerin kaydolmuş kullanıcılara konu tabanlı olarak gönderilmesi sağlanmıştır. Tezin katkıları aşağıdaki gibi sıralanabilir:

- İyi bir sınıflandırma işleminin yapılabilmesi için veri setinin; farklı aydınlanma koşullarında (gece-gunduz), farklı ölçeklerde (çözünürlüklerde), farklı yüzey renklerinde ve farklı bakış açılarında görüntülere sahip olması gerekmektedir. Beijing Institute of Technology Üniversitesi Media Computing and Intelligent Systems (Medya Hesaplama ve Akıllı Sistemler) Laboratuvarı çalışanları tarafından oluşturulan ve akademik çalışmalar için paylaşılan “BIT-Vehicle Dataset” veri seti [14] ve kendi oluşturduğumuz veri seti kullanılmıştır. Böylece bu tez kapsamında ulusal araç tipi veri seti oluşturulması sağlanmıştır (Bölüm 2.3’e bakınız).
- Hareket halindeki imge bölgelerinin tespit edilmesi ve kısımların araç olanlarının dikkate alınması sağlanmıştır (Bölüm 2.2’ye bakınız).
- Tespit edilen hareketli imge bölgelerinden sınıflandırma aşamasındaki özellik vektöründe kullanılmak üzere elde edilen imge bölgeleri için özellik vektörünün oluşturulması gerçeklenmiştir (Bölüm 2.1’e bakınız).
- Elde edilen özellik vektörleri kullanılarak farklı sınıflandırma algoritmaları (DVM ve KSA) ile hareketli araç imgeleri farklı kategorilere sınıflandırılacak ve sınıflandırma işlemlerinin başarımlarının test edilmesi sağlanmıştır (Bölüm 2.3.1 ve Bölüm 2.3.2’e bakınız).

- Araç renk ve hızlarına göre sınıflandırma işlemleri gerçeklenmiştir (Bölüm 2.4 ve Bölüm 2.5'e bakınız).
- Konu tabanlı olarak yayınla/kaydol modeli ile sınıflandırılan, renk ve hız bilgilerine sahip görüntülerin tüketicilere/istemcilere gönderilmesi sağlanmıştır. Bu kısım tezin en özgün yanını oluşturmaktadır. Belirli konulara kaydolan kullanıcılara görüntüler dağıtılmıştır. Normalde görüntü işleme uygulamaları çok kaynak (CPU, bellek vs.) ve zaman gerektirirken bu görüntüler kamera-video (streaming) olunca durum daha da içinden çıkılmaz bir durum almaktadır. Bu anlamda geliştirilen sistemle gerçek zamanda yapılması zor olan (hard-realtime) uygulamaları gerçek zamana yakına (soft-real time) dönüştürerek konu-tabanlı olarak video görüntülerine erişim sağlanmıştır. Bu şekilde, gerçek zamanlı kamera görüntüleri üzerindeki çalışmalarda ölçekleme ve performans problemlerine çözüm sunulmuştur. (Bölüm 2.7 - Bölüm 2.8'e bakınız).

Gelecek çalışmalarda araç hızının tespitinin performansı iyileştirilebilir veya hız tespit algoritması başka bir algoritmaya değiştirilebilir. Ayrıca araçların plakasını tanıma, şerit ihlalı yapan, kırmız ışıkta geçen arabaları tespit etme ve bu işlem sonucunda elde edilen bilgileri kaydederek daha sonra analiz işlemleri yapılabilir. Böylelikle daha kapsamlı akıllı trafik sistemleri geliştirilebilir, akıllı ve güvenli şehirler projelerine katkı sağlanabilir.

KAYNAKLAR

[1] Roess R.R., Prassas E.S., McShane W.R., *Traffic Engineering*, 4th ed., Prentice Hall, Michigan, 2010.

[2] Akbaş A., Avcı C., Delibaşoğlu İ., Karakullukçu B., Ulusal AUS Sistem Mimarisinin Tasarımı Üzerine, *1. Karayolu Akıllı Ulaşım Sistemleri Kongre ve Sergisi (KAUS 2014)*, İstanbul, Türkiye, 26-28 Mayıs 2014.

[3] http://www.resmigazete.gov.tr/eskiler/2014/10/20141025-21-1.pdf (Ziyaret Tarihi: 6 May 2018).

[4] Messelodi S., Modena C.M., Zanin M. A., Computer Vision System For The Detection And Classification Of Vehicles At Urban Road Intersections, *Pattern Anal Appl*, 2005, **8**(1-2), 17–31.

[5] Buch N., Orwell J., Velastin S.A., Detection And Classification Of Vehicles For Urban Traffic Scenes., *In: Proceedings of 5th international conference on visual information engineering*, Xian, China, 29 July–1 August 2008.

[6] Morris B., Trivedi M., Robust Classification And Tracking Of Vehicles In Traffic Video Streams. *In: Proceedings of IEEE Intelligent Transportation Systems Conference*, Toronto, Canada, 17–20 September 2006.

[7] Morris B., Trivedi M, Improved Vehicle Classification In Long Traffic Video By Cooperating Tracker And Classifier Modules, *In AVSS '06: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*, Sydney, Australia, 22-24 November 2006.

[8] Chen X., Zhang C.C., Vehicle Classification From Traffic Surveillance Videos At A Finer Granularity, *In Advances In Multimedia Modeling, Lecture Notes in Computer Science*, 2007, **4351**, 772-781.

[9] Zhang C., Chen X., Chen W., A PCA-based Vehicle Classification Framework, *IEEE International Workshop on Multimedia Databases and Data Management, in conjunction with IEEE International Conference on Data Engineering (ICDE2006)*, Atlanta, Georgia, USA, 3-7 April 2006.

[10] Gandhi T., Trivedi M.M., Video Based Surround Vehicle Detection, Classification And Logging From Moving Platforms: Issues And Approaches, *In Intelligent Vehicles Symposium*, Istanbul, Turkey, 13-15 June 2007.

[11] Zhang G., Avery R.P., Wang Y., Video-Based Vehicle Detection And Classification System For Real-Time Traffic Data Collection Using

Uncalibrated Video Cameras, *Transportation Research Record*, 1993, 138-147.

[12] Duman D., Akar G. B., Moving vehicle classification, *21st Signal Processing and Communications Applications Conference (SIU)*, Haspolat, Turkey, 24-26 April 2013.

[13] Ghada S.M., Vehicle Type Classification with Geometric and Appearance Attributes, *World Academy of Science, Engineering and Technology International Journal of Civil, Structural, Construction and Architectural Engineering*, 2008, **8**(3), 273-278.

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

[15] Khan F. S., Weijer J., M. Vanrell, Modulating Shape Features by Color Attention for Object Recognition, *Int J Comput Vis*, 2011, **98**(1), 49-64.

[16] Felzenszwalb P. F., Girshick R. B., McAllester D., Ramanan D., Object Detection with Discriminatively Trained Part-Based Models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010, **32**(9), 1627-1645.

[17] Zhang J., Huang K., Yu Y., Tan T., Boosted local structured HOG-LBP for object localization, *IEEE Computer Vision and Pattern Recognition (CVPR) 2011*, Colorado Springs, CO, USA, 20-25 June 2011.

[18] Sande K., Gevers T., Snoek C., Evaluating Color Descriptors for Object and Scene Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011, **32**(9), 1582-1596.

[19] Eken S., *Dağıtık Sistemler İçin Ada Uydu Görüntülerinin Vektörel Modellenmesi ve Zaman-Mekansal Analizi*, Kocaeli Üniversitesi, Kocaeli, 2012.

[20] Weijer J., Schmid C., Verbeek J., Larlus D., Learning Color Names for Real-World Applications, *IEEE Transactions on Image Processing*, 2012, **18**(7), 1512-1523.

[21] Pornpanomchai C. ve Kongkittisan K., Vehicle speed detection system, *2009 IEEE International Conference on Signal and Image Processing Applications*, Kuala Lumpur, Malaysia, 18-19 November 2009.

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

[23] Pelegri J., Alberola J., Llario V., Vehicle detection and car speed monitoring system using GMR magnetic sensors, *IEEE 2002 28th Annual Conference*

of the Industrial Electronics Society. IECON 02, Sevilla, Spain, 5-8 Nov. 2002.

[24] Koide R., Kitamura S., Nagase T., Araki T., Araki M., Ono H., A Punctilious Detection Method for Measuring Vehicles' Speeds, *2006 International Symposium on Intelligent Signal Processing and Communications*, Tottori, Japan, 12-15 December 2006.

[25] Cheng H. H., Shaw B. D., Palen J., Lin B., Chen B., Wang Z., Development and field test of a laser-based nonintrusive detection system for identification of vehicles on the highway, *IEEE Transactions on Intelligent Transportation Systems*, 2013, **6**(2), 147-155.

[26] Osman Z., Chahine S. A., Novel speed detection scheme, *The 14th International Conference on Microelectronics*, Beirut, Lebanon, 13 December 2002.

[27] Lin H. Y., Li K. J., Motion blur removal and its application to vehicle speed detection, *2004 International Conference on Image Processing, 2004. ICIP '04*, Singapore, Singapore, 24-27 October 2004.

[28] Alefs B., Schreiber D., Accurate Speed Measurement from Vehicle Trajectories using AdaBoost Detection and Robust Template Tracking, içinde *2007 IEEE Intelligent Transportation Systems Conference*, Seattle, WA, USA, 30 September-3 October 2007.

[29] Pumrin S., Dailey D. J., Roadside camera motion detection for automated speed measurement, *Proceedings. The IEEE 5th International Conference on Intelligent Transportation Systems*, Singapore, Singapore, 6 September 2002.

[30] https://en.wikipedia.org/wiki/Publish%E2%80%93subscribe_pattern (Ziyaret Tarihi: 16 May 2018).

[31] https://www.omg.org/spec/EVNT/1.1. (Ziyaret Tarihi: 16 May 2018).

[32] Fabret F., Jacobsen H. A., Llirbat F., Pereira J., Ross K. A., Shasha D., *Filtering Algorithms and Implementation for Very Fast Publish/Subscribe*, Santa Barbara, California, USA, 21-24 May 2001.

[33] Eugster P., Type-based publish/subscribe: Concepts and experiences, *ACM Transactions on Programming Languages and Systems*, 2007, **29**(1), 6.

[34] Banno R., Takeuchi S., Takemoto M., Kawano T., Kambayashi T., Matsuo M., A Distributed Topic-Based Pub/Sub Method for Exhaust Data Streams towards Scalable Event-Driven Systems, *2014 IEEE 38th Annual Computer Software and Applications Conference*, Vasteras, Sweden, 21-25 July 2014.

[35] Gascon-Samson J., Garcia F. P., Kemme B., Kienzle J., Dynamoth: A Scalable Pub/Sub Middleware for Latency-Constrained Applications in the

Cloud, *2015 IEEE 35th International Conference on Distributed Computing Systems*, Columbus, OH, USA, 29 June-2 July 2015.

[36] Gray A. J. G., Nutt W., A Data Stream Publish/Subscribe Architecture with Self-adapting Queries, *The 13th International Conference on Cooperative Information Systems*, Agia Napa, Cyprus, 31 October - 4 November 2005.

[37] Zou Y., Cao J., Wu H., TraffiCast: Real-time Pub/Sub based video surveillance system over interconnected WMNs and WSNs, *2010 6th IEEE International Conference on Distributed Computing in Sensor Systems Workshops (DCOSSW)*, Santa Barbara, CA, USA, 21-23 June 2010.

[38] Dalal N., Triggs B., Histograms of oriented gradients for human detection, içinde *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, 20-25 June 2005.

[39] Cortes C., Vapnik V., Support-Vector Networks, *Machine Learning*, 1995, **20**(3), 273-297.

[40] Lecun Y., Bottou L., Bengio Y., Haffner P., Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, **86**(11), 2278-2324.

[41] https://www.kaggle.com/vikrishnan/boston-house-price-prediction-regression. (Ziyaret Tarihi: 08 Haziran 2018).

[42] https://www.cs.toronto.edu/~kriz/cifar.html. (Ziyaret Tarihi: 11 Haziran 2018).

[43] https://www.quora.com/How-does-one-determine-stride-size-in-CNN-filters (Ziyaret Tarihi: 12 Haziran 2018).

[44] Krizhevsky A., Sutskever I., Hinton G. E., Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems*, Lake Tahoe, Nevada, 03-06 December 2012.

[45] Hinton G. E., Nair V., Rectified linear units improve restricted boltzmann machines, *ICML'10 Proceedings of the 27th International Conference on International Conference on Machine Learning*, Haifa, Israel, 21-24 June 2010.

[46] https://en.wikipedia.org/wiki/Activation_function. (Ziyaret Tarihi: 06 Haziran 2018).

[47] https://en.wikipedia.org/wiki/Rectifier_(neural_networks). (Ziyaret Tarihi: 26 Mayıs 2018).

[48] https://en.wikipedia.org/wiki/Softmax_function. (Ziyaret Tarihi: 31 Mayıs 2018).

[49] https://arxiv.org/abs/1710.05941. (Ziyaret Tarihi: 01 Haziran 2018)

[50] https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/. (Ziyaret Tarihi: 12 Haziran 2018).

[51] https://arxiv.org/abs/1207.0580. (Ziyaret Tarihi: 31 Mayıs 2018).

[52] https://www.reddit.com/r/MachineLearning/comments/4w6tsv/ama_we_are_the_google_brain_team_wed_love_to. (Ziyaret Tarihi: 05 Haziran 2018).

[53] https://en.wikipedia.org/wiki/CUDA. (Ziyaret Tarihi: 04 Haziran 2018).

[54] https://developer.nvidia.com/cudnn. (Ziyaret Tarihi: 06 Haziran 2018).

[55] https://developer.nvidia.com/embedded/buy/jetson-tx2. (Ziyaret Tarihi: 06 Haziran 2018).

[56] https://www.tensorflow.org/. (Ziyaret Tarihi: 06 Haziran 2018).

[57] http://www.deeplearning.net/software/theano/. (Ziyaret Tarihi: 06 Haziran 2018).

[58] https://pytorch.org/. (Ziyaret Tarihi: 06 Haziran 2018).

[59] http://caffe.berkeleyvision.org/. (Ziyaret Tarihi: 06 Haziran 2018).

[60] https://keras.io/. (Ziyaret Tarihi: 06 Haziran 2018).

[61] https://github.com/uber/horovod. (Ziyaret Tarihi: 06 Haziran 2018).

[62] https://kafka.apache.org/. (Ziyaret Tarihi: 06 Haziran 2018).

[63] https://engineering.linkedin.com/apache-kafka/how-we_re-improving-and-advancing-kafka-linkedin. (Ziyaret Tarihi: 10 Haziran 2018).

[64] https://docs.opencv.org/3.4.1/modules.html. (Ziyaret Tarihi: 11 Haziran 2018).

[65] https://www.ffmpeg.org/ffmpeg.html. (Ziyaret Tarihi: 11 Haziran 2018).

[66] https://www.nginx.com/resources/wiki/. (Ziyaret Tarihi: 11 Haziran 2018).

[67] https://github.com/arut/nginx-rtmp-module. (Ziyaret Tarihi: 07 Haziran 2018).

[68] https://en.wikipedia.org/wiki/Node.js. (Ziyaret Tarihi: 11 Şubat 2018).

[69] http://highscalability.com/blog/2012/10/4/linkedin-moved-from-rails-to-node-27-servers-cut-and-up-to-2.html. (Ziyaret Tarihi: 07 Haziran 2018).

[70] https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal. (Ziyaret Tarihi: 30 Mayıs 2018).

[71] https://socket.io. (Ziyaret Tarihi: 11 Haziran 2018).

[72] https://docs.opencv.org/3.3.0/de/da6/classcv_1_1cuda_1_1HOG.html. (Ziyaret Tarihi: 29 May 2018).

[73] Şentaş A. ve diğ., 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 and Web Technologies*, Tirana, Albania, 15-17 March, 2018.

[74] Girshick R., Donahue J., Darrell T., Malik J., Rich feature hierarchies for accurate object detection and semantic segmentation, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 23-28 June 2014.

[75] Girshick R., Fast R-CNN, *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 7-13 December 2015.

[76] Ren S., He K., Girshick R., Sun J., Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015, **39**(6), 1137-1149.

[77] He K., Gkioxari G., Dollár P., Girshick R., Mask R-CNN, *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 22-29 October 2017.

[78] Redmon J., Farhadi A., YOLO9000: Better, Faster, Stronger, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21-26 July 2017.

[79] Thadagoppula P. K., V. Upadhyaya, Speed detection using image processing, *2016 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, Tangerang, Indonesia, 3-5 October 2016.

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

KİŞİSEL YAYIN VE ESERLER

[1] **Tashiev İ.**, Kul S., Şentaş A., Küçükayvaz F., Eken S., Sayar A., Becerikli Y. Konvolüsyonel Sinir Ağı Kullanarak Gerçek Zamanlı Araç Tipi Sınıflandırması, *1.Ulusal Bulut Bilişim ve Büyük Veri Sempozyumu B3S'17,* Antalya, 19-20 Ekim 2017.

[2] Şentaş A., **Tashiev İ.**, Küçükayvaz F., Kul S., Eken S., Sayar A., Becerikli Y., Performance Evaluation of Support Vector Machine and Convolutional Neural Network Algorithms in Real-Time Vehicle Type Classification. In *International Conference on Emerging Internetworking, Data & Web Technologies,* Tirana, Albania, 15-17 March, 2018.

ÖZGEÇMİŞ

Isabek TASHIEV 1990'da Kırgızistan'da doğdu. Lise öğrenimini Gapar Aytiev Lise'sinde tamamladı. 2008 yılında girdiği Kırgızistan-Türkiye Manas Üniversitesi Bilgisayar Mühendisliği Bölümü'nden 2013 yılında ikincilik ile mezun oldu. 2015 yılında Kocaeli Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı'nda yüksek lisans eğitimine başladı. Yüksek lisans eğitiminde makine öğrenmesi, görüntü işleme ve yapay zeka konusunda çalışmaları bulunmaktadır.