T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İÇ MEKANLARDA ZEMİN YOL MODELİ ÜZERİNDE DERİN ÖĞRENME İLE OTONOM ARAÇLARIN ROTA TAKİBİ

DOKTORA TEZİ

Mustafa ERGİNLİ

Endüstri Mühendisliği Anabilim Dalı

Endüstri Mühendisliği Bilim Dalı

ŞUBAT 2023

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İÇ MEKANLARDA ZEMİN YOL MODELİ ÜZERİNDE DERİN ÖĞRENME İLE OTONOM ARAÇLARIN ROTA TAKİBİ

DOKTORA TEZİ

Mustafa ERGİNLİ

Endüstri Mühendisliği Anabilim Dalı

Endüstri Mühendisliği Bilim Dalı

Tez Danışmanı: Prof. Dr. İbrahim ÇİL

ŞUBAT 2023

Mustafa ERGİNLİ tarafından hazırlanan "İç Mekanlarda Zemin Yol Modeli Üzerinde Derin Öğrenme İle Otonom Araçların Rota Takibi" adlı tez çalışması 17.02.2023 tarihinde aşağıdaki jüri tarafından oy birliği/oy çokluğu ile Sakarya Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Endüstri Mühendisliği Bilim Dalı'nda DOKTORA TEZİ olarak kabul edilmiştir.

Tez Jürisi

Jüri Başkanı :	**Prof. Dr. İbrahim ÇİL** (Danışman) Sakarya Üniversitesi	.....
Jüri Üyesi :	**Prof. Dr. Cemil ÖZ** Sakarya Üniversitesi	.....
Jüri Üyesi :	**Prof. Dr. Ayhan İSTANBULLU** Balıkesir Üniversitesi	.....
Jüri Üyesi :	**Doç. Dr. Berrin DENİZHAN** Sakarya Üniversitesi	.....
Jüri Üyesi :	**Doç. Dr. İbrahim KÜÇÜKKOÇ** Balıkesir Üniversitesi	.....

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Sakarya Üniversitesi Fen Bilimleri Enstitüsü Lisansüstü Eğitim-Öğretim Yönetmeliğine ve Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Etiği Yönergesine uygun olarak hazırlamış olduğum "İÇ MEKANLARDA ZEMİN YOL MODELİ ÜZERİNDE DERİN ÖĞRENME İLE OTONOM ARAÇLARIN ROTA TAKİBİ" başlıklı tezin bana ait, özgün bir çalışma olduğunu; çalışmamın tüm aşamalarında yukarıda belirtilen yönetmelik ve yönergeye uygun davrandığımı, tezin içerdiği yenilik ve sonuçları başka bir yerden almadığımı, tezde kullandığım eserleri usulüne göre kaynak olarak gösterdiğimi, bu tezi başka bir bilim kuruluna akademik amaç ve unvan almak amacıyla vermediğimi ve 20.04.2016 tarihli Resmi Gazete 'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince Sakarya Üniversitesi'nin abonesi olduğu intihal yazılım programı kullanılarak Enstitü tarafından belirlenmiş ölçütlere uygun rapor alındığını, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun ortaya çıkması halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi beyan ederim.

(17/02/2023).

(imza)

Mustafa ERGİNLİ

Eşime ve çocuklarıma…

TEŞEKKÜR

Doktora eğitimim boyunca bilgi ve tecrübeleriyle bana rehberlik eden ve her zaman yanımda olduğuna inandığım kıymetli danışman hocam Prof. Dr. İbrahim ÇİL'e en içten saygılarımı ve teşekkürlerimi sunarım.

Tez izleme jürimde zaman ayırarak çalışmam hakkında yaptıkları yapıcı eleştiri, öneri ve yorumlarından dolayı Prof. Dr. Cemil ÖZ'e ve Doç. Dr. Berrin DENİZHAN'a çok teşekkür ederim.

Mustafa ERGİNLİ

İÇİNDEKİLER

Sayfa

KISALTMALAR

AGV : Automated Guided Vehicle
AVM : Autonomous Vehicle Model
CNN : Convolutional Neural Network
DC : Direct Current
DRR : Dynamic Resource Reservation
GANS : Generative Adversarial Networks
GPS : Global Positioning System
GPU : Graphics Processing Unit
GRU : Gated Recurrent Unit
IMU : Inertial Measurement Unit
IPM : Inverse Perspective Mapping
K-SSD : Kalman-Based Single Shot Multi-Box Detector
LASSO : Least Absolute Shrinkage and Selection Operator
LiDAR : Light Detection and Ranging
LSTM : Long Short-Term Memory
MNIST : Modified National Institute of Standards and Technology
PCA : Principal Component Analysis
Radar : Radio Detection and Ranging
ReLU : Rectified Linear Unit
RNN : Recurrent Neural Network
ROI : Region of Interest
SGD : Stochastic Gradient Descent
SRNN : Simple Recurrent Neural Networks
SVM : Support Vector Machine

TABLO LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

İÇ MEKANLARDA ZEMİN YOL MODELİ ÜZERİNDE DERİN ÖĞRENME İLE OTONOM ARAÇLARIN ROTA TAKİBİ

ÖZET

Üretim ortamlarında malzemelerin bir yerden başka bir yere insansız olarak taşınması yıllar boyunca önemli olmuştur. Malzemelerin insansız olarak taşınması için otonom araçlar geliştirilmiştir. Bir otonom araç zeminde bulunan sabit çizgileri, bakır kabloları takip ederek ilerleyebildiği gibi dolaşım için mıknatıs, lazer, radyo dalgası ve görüş kamerası da kullanmaktadır. Otonom araçlarda kendi kendine dolaşım temel bir konudur. Zaman içinde otonom araçlar geliştikçe ve kullanımları arttıkça çözülmesi gereken çeşitli sorunlar ortaya çıkmıştır.

Gerçek zamanlı rota takibi, endüstriyel tesislerde kullanılan otonom araçlar için önemli bir araştırma konusudur. Bir otonom aracın hedefine giden rotayı takip ederken rotası üzerindeki diğer nesne veya araçlarla çarpışmaması, rotasında ilerlerken rotasının dolu olması durumunda dinamik olarak başka bir rota seçebilmesi çözülmesi gereken problemler arasındadır.

Yapay zekanın gelişmesi ve özellikle derin öğrenmenin ortaya çıkışından sonra otonom araçlarda araştırmalar farklı bir boyuta taşınmıştır. Derin öğrenme mimarilerinden olan evrişimli sinir ağları kameradan alınan ve farklı açılardan elde edilmiş görüntüleri başarılı bir şekilde sınıflandırarak aracın rota takibini yapmasına ve konumu belirlemesine yardımcı olmaktadır.

Bu çalışmada, otonom araçların hedeflerine gitmeleri için en kısa rotanın bulunması, bu rotalar üzerinde otonom olarak nasıl ilerleyecekleri, rotalarda ilerlerken diğer bir araçla çarpışma sorunlarının çözümü için derin öğrenme tabanlı bir zemin yolu modeli ve algoritması geliştirilmiştir. Zemin yolu modeli araçların konumlarını da gösteren işaretçilerden (marker) oluşmaktadır. Araçların ilerleyeceği rotalar bu işaretçiler kullanılarak oluşturulmaktadır. Bir rota birbirinden faklı sayıda ve hedefe kadar ilerleyen çok sayıda işaretçi içermektedir. Çakışan kavşaklarda aynı işaretçiler bulunmaktadır. İşaretçilerin üzerindeki değerler artan veya azalan ardışık harf ve sayıdan oluşmaktadır. İşaretçilerden oluşan zemin yolu modeli vektör olarak bir merkezi sunucudan tüm araçlara gönderilmektedir. Görev verilecek araca sunucu tarafından hedef işaretçi adresi gönderilmektedir. Otonom aracın kendisinde yüklü olan algoritma ile hedefe giden tüm rotalar hesaplanarak en kısa olan seçilmektedir. Araç seçilen rotada ilerlemektedir. Araç ilerlerken rotadaki işaretçi görüntüleri aracın kamerası ile algılanmaktadır. Bu görüntülere görüntü işleme yöntemleri uygulanarak araca en yakın işaretçi bulunmaktadır. Elde edilen bu görüntü önceden eğitilmiş bir derin CNN modeliyle sınıflandırılmaktadır. Sınıflandırılan görüntü araca yüklenmiş hedef vektör ile aynı ise araç hedefe doğru ilerlemektedir. Aynı zamanda bu görüntü aracın konumunu belirlemek için sunucuya gönderilmektedir. Sunucudaki bu işaretçi adresi diğer araçlar tarafından bu aracın konumunu belirlemek için kullanılmaktadır. Bu adres araçların çarpışmasını engellemek için de kullanılmaktadır. Eğer bir araç bu

işaretçiye yakın bir işaretçide ise diğer aracın ilerlemesini beklemektedir. Sınıflandırılan görüntü araca yüklenmiş hedef vektör ile farklı ise araç çevresindeki diğer işaretçileri aramakta ve bulunan işaretçileri sınıflandırarak algoritmayı sürdürmektedir.

Otonom araçlar sensörlerden gelen çok fazla veriyi analiz ettiklerinden büyük veri ile uğraşmaktadırlar. Bu nedenle bu tezde kullanılan otonom araçlarda gerçekleştirilen veri işleme uç hesaplama (Edge Computing) ile yapılmaktadır. Sunucuya ise sadece az miktarda veri gönderilip alınmaktadır. Yapmış olduğumuz deneysel çalışmalar sonucunda araçların rota takibi başarı ile sağlanmıştır.

Otonom rota izleme araçları için CNN derin öğrenme modeline dayalı geliştirilen zemin yolu modeli ve algoritması araçların çarpışmasını engelleyerek otonom rota takibini sağlayan bir iç mekân konumlandırma sistemidir. Araçlar, zemin yolu modelinde hedef konuma giden tüm rotaları hesaplayarak en kısa yolu seçmekte ve rotayı yüksek doğrulukla takip etmektedirler.

ROUTE TRACKING OF AUTONOMOUS VEHICLES WITH DEEP LEARNING ON THE FLOOR PATH MODEL IN INDOOR AREAS

SUMMARY

The unmanned transport of materials from one place to another in production environments has been important over the years. Autonomous vehicles have been developed for the unmanned transport of materials. An autonomous vehicle can proceed by following fixed lines and copper cables on the ground, as well as using magnets, lasers, radio waves, and vision cameras for navigation. Self-circulation is a fundamental issue in autonomous vehicles. Over time, as autonomous vehicles have developed and their use has increased, various problems have emerged that need to be solved.

Real-time route tracking is an important research topic for autonomous vehicles used in industrial plants. Problems that need to be solved are that an autonomous vehicle does not collide with other objects or vehicles on its route while following the route to its destination and that it can dynamically choose another route in case its route is not empty. Different researchers have worked to solve these problems. In previous studies, solutions were built on mathematical models. Mathematical models can be used in static environments. Another disadvantage of mathematical models is that they are insufficient for the solution when data increases exponentially. The use of camera data in production environments has limited the use of mathematical models. With the development of machine learning and its use in different environments, researchers have evolved in this direction to solve problems. New machine learning methods have been proposed by researchers, and these methods have been used in production environments. Recently, the increase in the amount of data with the development of technology and different sensors has shown that previous machine learning methods cannot be used efficiently with big data.

After the development of artificial intelligence and especially the emergence of deep learning, research in autonomous vehicles has moved to a different dimension. The volume increase in the sensor data in autonomous vehicles, speed, and various data from different sensors emerge as big data. The internet of things emerging here enables devices and systems to be connected and to communicate data over a network. Thanks to the internet of things, which has started to be used in production environments today, the amount of data collected reaches huge volume values. It is not easy to process such big data with existing machine learning methods and to obtain a learning model using different data. The more different the data, the more time it takes to process it. For these reasons, deep learning is used to obtain big data models. Deep learning architectures produce efficient results with big data. In our studies, convolutional neural networks, one of the deep learning architectures, successfully classify the images taken from the camera and obtained from different angles, helping the vehicle to follow the route and determine the location.

There are different sensors, such as a camera, distance sensor, and LIDAR, on an autonomous vehicle. These sensors produce different and a lot of data over time during the vehicle's movement. The processing of these data becomes important for the autonomous vehicle's movement and reaching its destination. Since the data obtained from different sensors is continuously collected over time, it also increases rapidly in volume. In this case, the concept of big data is encountered in new types of autonomous vehicles. Even in autonomous vehicles moving in an indoor production environment, sending and processing big data to a server and sending this processed information back to the vehicle takes much communication time. In addition, hardware problems, such as wireless network problems, arise in the transmission of big data. Such problems cause issues in the vehicle's movement and even leave the vehicle inactive in time. To avoid these problems, Edge Computing technology is used in this thesis. In short, edge computing uses a distributed structure in the data processing. By using edge computing, the processing of sensor data is done in each autonomous vehicle, improving calculation times. Thus, it communicates with the server using fewer data, and at the same time, bandwidth problems of the network are avoided. The increase in the number of autonomous vehicles reveals major problems in communication with the server. Thanks to edge computing, data processing speed and communication speed increase as each sensor data are processed in the vehicles themselves. This way, only the location information address is sent and received quickly by the server.

In this study, a deep learning-based floor path model and algorithm are developed to find the shortest route for autonomous vehicles to go to their destination, how to proceed autonomously on these routes, and to solve the problems of collision with another vehicle while driving on routes. The floor path model consists of markers that also show the locations of the vehicles. The routes on which the vehicles will travel are created using these markers. A route contains a different number of markers leading to the destination. Overlapping intersections have the same markers. The values above the markers consist of ascending or descending consecutive letters and numbers. The floor path model consisting of markers is sent to all vehicles from a central server as a vector. The target location address is sent by the server to the vehicle to be tasked. With the algorithm installed in the autonomous vehicle itself, all routes to the destination are calculated, and the shortest one is selected. The vehicle continues on the selected route. While the vehicle is moving, the marker images on the route are detected by the vehicle's camera. By applying image processing methods to these images, the closest marker to the vehicle is found. This resulting image is classified with a pre-trained deep CNN model. If the classified image is the same as the target vector loaded on the vehicle, the vehicle moves toward the target location. At the same time, this image is sent to the server to determine the location of the vehicle. This location address on the server is used by other vehicles to determine the location of this vehicle. This address is also used to prevent vehicles from colliding. If a vehicle is at a marker close to this marker, it waits for the other vehicle to advance. Autonomous vehicles are dealing with big data as they analyze a lot of data coming from sensors. For this reason, data processing performed in autonomous vehicles used in this thesis is done with edge computing. Only a small amount of data is sent and received to the server. As a result of the experimental studies we have done, the route tracking of the vehicles has been successfully achieved.

A deep learning-based floor path model is designed for route tracking of autonomous vehicles. This model is a collision avoidance indoor positioning system for

autonomous route-tracking vehicles based on the CNN deep learning model. A new floor path model and algorithm based on indoor positioning has been developed for autonomous vehicles. Vehicles calculate all routes to the target location in the floor path model, choose the shortest route and follow the route with high accuracy.

1. GİRİŞ

İnsan müdahalesi olmadan malzemeleri bir departmandan diğerine taşımak üzere programlanan otonom araçlar, akıllı üretimin bir parçası olarak görülmekte ve iş yükünü azaltmaktadır. Rota takip stratejilerinin belirlenmesi ve gerçek zamanlı rota takibi, endüstriyel üretim alanlarında kullanılan otonom araçlar için önemli bir araştırma konusudur [1, 2]. Çalışma ortamına bağlı olarak değişken ortam yapısına uygun esnek bir rota izlenmesi önemlidir. Çalışma ortamının statik ve ön bilgisi veya dinamik ve bilinmeyen durum, otonom aracın yörünge planlama özelliğinde belirleyici parametrelerdir. Araca sabit bir rota tahsis edildiğinde ve bu rota herhangi bir engelle kesintiye uğramadığında statik yörünge planlayıcıları önerilmektedir. Harita bilgisi olmayan veya sürekli değişen engel yapıları olan ortamlar için dinamik rota planlayıcılar önerilmektedir. Güzergâh takibinde yerde bakır hat takibi, kablosuz yönlendirme sistemleri, lazer sistemleri gibi geleneksel yöntemler kullanılmaktadır.

Güzergâh tasarımı, otonom bir araç sistemi geliştirmenin ilk ve en önemli adımıdır. Bu süreç, üretim alanlarındaki iş yükünün dengelenmesi ve trafik yoğunluğunun azaltılması açısından hayati önem taşımaktadır. Güzergâh, araç yüklerini bir alandan diğerine [3] tek yönlü veya çift yönlü [4] taşımak için kullanılan kılavuz yollardan oluşmaktadır.

Derin öğrenme, veri özelliklerini kullanarak karar verebilen ve uzmanlara bağımlılığı azaltabilen bir yöntemdir [5]. CNN (Evrişimli Sinir Ağı - Convolutional Neural Network), yaygın olarak kullanılan bir derin öğrenme yöntemi türüdür. Evrişimli, doğrultulmuş doğrusal birimler (ReLU'lar), havuzlama ve tam bağlantılı katmanlardan oluşan çok katmanlı bir sinir ağı modelidir [6]. Derin öğrenme yöntemleri kullanılarak kamera görüntüleri hızlı bir şekilde analiz edilebilmektedir. Üretim ortamlarında kullanılan araçların güzergahları esnek bir şekilde tasarlanabilmektedir. Tasarlanan bu rotaların görüntüleri araç kameraları ile kolaylıkla elde edilebilmekte ve derin öğrenme modelleri kullanılarak belirlenebilmektedir. Böylece araçlar, insan müdahalesi olmadan rotalarını analiz edebilmekte ve hedeflerine gidebilmektedirler. Smolyansky ve ark. TrailNet [7] adı verilen derin bir sinir ağına dayalı ormanlar gibi dış ortamlar

için bir mikro hava aracı sistemi sunmuşlardır. Wang ve ark. AGV (Otonom Yönlendirmeli Araçlar - Automated Guided Vehicles)' ye monte edilmiş bir kamera kullanarak yol işaretleri elde etmişlerdir. Bu sinyalleri geliştirdikleri bir sinir ağına girdi olarak uygulamışlar ve AGV' nin görsel tabanlı yol takibini gerçekleştirmişlerdir [8].

1.1. Tez Çalışmasının Amacı

Günümüzün endüstriyel ortamları için insan gücü ile çevreci olmayan, yakıt ve maliyetli enerji kaynakları tüketen taşıma araçları etkin malzeme taşıma araçları olarak kabul görmemektedir. Bu konu araştırmacıların üzerinde çalıştığı yaygın genel problemlerden biridir.

Hipotez 1. Üretim ortamının zeminine, otonom araçların kullanacağı ve hedeflerine ulaştıracağı işaretçilerden oluşan farklı rotalar oluşturulabilir mi?

Hipotez 2. Otonom araçlar derin öğrenme ve bilgisayar görmesini kullanarak üretim ortamının zemine yerleştirilen işaretçileri görerek hedefe giden farklı rotaları takip edebilirler mi?

Hipotez 3. Otonom araçların derin öğrenmeyi kullanarak çok sayıda otonom aracın bulunduğu ortamlarda diğer araçları ve zemindeki engelleri bilgisayar görmesi ile görerek çarpışmalar engellenebilir mi?

Hipotez 4. Ortamda çok fazla otonom araç olduğunda zemin yolu modeli ürerinde oluşturulan rotalar kullanılarak trafik yoğunluğu dengelenebilir mi?

Bu çalışmada, iç mekân üretim ortamlarında kullanılan otonom araçların gerçek zamanlı rota takibi ve çarpışma problemlerini çözmek için görüntü işleme ve derin öğrenme tabanlı bir iç mekân konumlandırma sistemi geliştirilmiştir [9, 10]. Önerilen sistemde zemin için işaretleyicilerden oluşan bir zemin yolu modeli geliştirilmiştir. Zemin yolu modelinde kullanılan işaretleyiciler, harf ve sayıların birleşiminden oluşmaktadır. Rotalar bu işaretleyiciler kullanılarak oluşturulmaktadır. İlk olarak, zemin yolu modeli bir sunucu tarafından otonom araçlara vektör olarak gönderilmektedir. Aracın hareket etmesi için hedef işaretleyici adresi gerekmektedir. Hedef adrese giden rotalar ve bu rotaların mesafeleri araç tarafından hesaplanmaktadır.

Hesaplanan rotalar arasından hedefe en yakın rota seçilmektedir. Araçtan dört işaretçi rota mesafesi içinde seçilen rota üzerinde başka bir araç yoksa araç modeli etkinleştirilmektedir. Seçilen rota üzerinde başka bir araç varsa hedefe daha yakın olan diğer rota için de aynı kontroller yapılmaktadır. Boş rota bulunamazsa model araç bekletilmektedir. Otonom araç rota üzerinde hareket ederken otonom araç modeli (AVM) üzerine monte edilen kameradan elde edilen görüntüler ön işleme görüntü yöntemleri ile işlenmektedir. Bu görüntüler, transfer öğrenme yöntemi kullanılarak eğitilen CNN modeline girdi olarak verilmektedir. Ağ çıktısı, otonom araç modelinin rotayı takip etmesini sağlamak için hedef rota vektörünün karşılık gelen öğesiyle karşılaştırılmaktadır. Ağ çıktısı ile rota vektör bilgisi eşit ise diğer araçların konum bilgileri sunucu ile iletişime kurularak kontrol edilmektedir. Adres bilgisi alınan araç, ilgili araçtan dört işaretçi uzaklıkta ise ilgili araç beklemekte aksi halde yoluna devam etmektedir. Bu işlemler araç hedefine ulaşana kadar tekrar etmektedir. Zemin yolu modeli ve algoritması, oluşturulan rotaları kullanarak, otonom aracın başka bir araçla çarpışmadan en uygun rotayı takip ederek hedefine ulaşmasını sağlamıştır.

1.2. Tezin Katkıları

Çalışmamızın temel katkıları şu şekilde özetlenebilir:

• Otonom araçların rota takibi için derin öğrenme tabanlı bir zemin yolu modeli geliştirilmiştir.

• Geliştirilen sistem, CNN derin öğrenme modeline dayalı otonom rota izleme araçları için çarpışmayı engelleyen bir iç mekân konumlandırma sistemidir.

• Otonom araçlar için iç mekân konumlandırmaya dayalı yeni bir zemin yol modeli ve algoritması geliştirilmiştir.

• Araçlar, zemin yolu modelinde hedef konuma giden tüm rotaları hesaplayarak en kısa yolu seçmekte ve rotayı yüksek doğrulukla takip etmektedirler.

• Yaptığımız deneyler modelimizin geçerliliğini doğrulamıştır.

Bu çalışmanın geri kalanı aşağıdaki gibi yapılandırılmıştır. Bölüm 2, ilgili çalışmaları özetlemektedir. Bölüm 3' de, makine öğrenmesi, derin öğrenme, evrişimli sinir ağları, transfer öğrenme, otonom araçlar, veri setinin hazırlanması, görüntü ilgi alanının tespit

edilmesi, geliştirilen rota takibi için derin öğrenme tabanlı zemin yolu modeli sunulmuştur. Bölüm 4’ de, deneysel sonuçlar yer almaktadır. Bölüm 5 sonuçları içermekte ve gelecekteki potansiyel çalışmaları önermektedir.

2. LİTERATÜR ARAŞTIRMASI

Son yıllarda otonom araçların geliştirilmesine yönelik çalışmalar yapılmaktadır. Üretim ortamlarında kullanılan otonom araçlarda farklı sorunlarla karşılaşılmaktadır. Bu sorunlardan biri gerçek zamanlı rota takibidir. Bir diğer sorun da rota üzerinde hareket eden aracın başka bir araçla çarpışmasıdır. Otonom araçlarla ilgili çalışmaların bir kısmı rota izleme stratejilerine odaklanmıştır. Bu stratejiler, üretim alanlarında kullanılan AGV' ler için önemlidir. Çarpışmadan kaçınma, AGV' lerin hareket planlaması ve kontrolünde kritik bir sorundur [11, 12]. AGV' ler aynı yolda aynı yönde ve aynı yolda zıt yönde hareket ettiğinde bir çarpışma meydana gelebilmektedir [13]. Güney ve ark. AGV' ler arasındaki hareket çakışmalarını çözmeyi amaçlayan bir kontrolör tasarlamışlardır. Kontrolör tasarımında AGV' lerin hareket özelliklerine dayanan geleneksel sağ elle iki yönlü trafik sistemini kullanmışlardır. Bir kavşakta karşılaşacak olan AGV' lerin geçiş hakkını belirleyerek çarpışmaları engellemişlerdir [14]. Zhao ve ark. AGV' ler için çarpışmadan kaçınmayı destekleyen dinamik kaynak rezervasyonu (DRR) tabanlı bir yöntem önermişlerdir. Bu yöntemde yerleşim düzeni aynı boyutta kare bloklara bölünmektedir. Her karede hareket eden AGV' ler puanları paylaşmışlardır. Bu kaynak noktalar, algoritmada gerçek zamanlı olarak kılavuz yollarından çıkarılmıştır. Yazarlar daha sonra dinamik kaynak rezervasyonları için kararlar almak için çarpışma ve kilitlenme önlemeden yararlanmışlardır. Bu iki parametre, zaman verimliliğini artırırken AGV' lerin çarpışma olmadan seyahat etmesine izin vermiştir. Çalışmadaki simülasyon sonuçları, seyahat süresinin %11 ile %28 arasında kısaldığını göstermiştir [15]. Guo ve ark. AGV' ler için çarpışmaları önlemek için bir yol planlama modeli geliştirmişlerdir. Yazarlar, minimum engelleme oranına göre AGV' nin seyahat hızını, çalışma süresini ve çarpışma mesafesi parametrelerini hesaplamışlardır. Farklı AGV' lerin ortalama engelleme oranını, bekleme süresini ve tamamlanma süresini, farklı etkileşimli protokoller altında çalışma süresini ve farklı ölçek haritalarının ortalama engelleme oranını karşılaştırmak için birkaç deney yapmışlardır. Deneyler sonucunda hızlanma kontrol yönteminin çarpışmayı daha hızlı çözebileceğini ve ikincil çarpışma olasılığını azaltabileceğini göstermişlerdir [16].

AGV' ler, ortamdaki yönlendirme bilgilerini içeren bir yönlendirme sistemini takip etmektedirler. AGV' ler tarafından kullanılan kılavuz teknolojileri, tel/endüktif kılavuzu, optik hat kılavuzu, manyetik bant kılavuzu, lazer navigasyon kılavuzu, kablosuz sensör ağı kılavuzu, görüş kılavuzu, barkod kılavuzu, ultrasonik kılavuzu ve GPS kılavuzudur [17-19].

Murakami, K. AGV yönlendirmesini karma tamsayılı bir doğrusal programlama problemi olarak formüle etmek için bir zaman-uzay ağı kullanmıştır. Bu ağda AGV' lerin ve malzemelerin yönlendirilmesini ayrı ayrı ele almıştır. Sistemini deneysel olarak test etmiştir [20]. Lu ve ark. AGV' lerin yönlendirme sorununun çözümünü derin bir pekiştirme-öğrenme algoritması kullanarak ele almışlardır. AGV' lerin yönlendirme problemini bir Markov karar süreci olarak modellemişler ve gerçek zamanlı yönlendirmeyi sağlamışlardır [21]. Holovatenko ve ark. belirli bir harita üzerinde iki nokta arasında istenen rotayı oluşturan ve izleyen, enerji verimliliğini de içeren bir matematiksel algoritma ve izleme sistemi önermişlerdir [22]. Literatürde AGV' lerin otonom sürüş yöntemlerine dayalı farklı çalışmalar rapor edilmiştir. Otonom sürüş yönteminin temel işlevi makine görmesidir. Makine görmesine dayalı sorunlardan bazıları, zemin yolu algılama ve şerit işaretleme algılamasını içermektedir [23-25]. Bayona ve ark. AGV modeline monte edilmiş bir kamera ile hat takibi için bilgisayar görmesi kullanmışlardır. Araç, yapılan testlerde zeminde düz bir çizgiyi takip etmede başarılı olmuştur [26]. Zhu ve ark. kameradan alınan görüntüyü işleyerek rotayı takip eden bir araç yolu izleme yöntemi önermişlerdir. İlk önce görüntünün gürültüsünü azaltarak ve hesaplama karmaşıklığını azaltmak için görüntüyü ikili hale getirerek yol durumunu netleştirmek için görüntüye bir medyan filtresi uygulamışlardır. Görüntü bozulmasını ve düşük yol izleme doğruluğu sorunlarını azaltmışlardır [27]. Lee ve ark. zeminde farklı noktalara yerleştirilmiş üçgen işaretleri tespit etmek için bir kamera kullanan bilgisayar görme tabanlı bir AGV navigasyon sistemi önermişlerdir. Bir görüntüdeki belirteçleri belirlemek için görüntünün renk tonu ve doygunluk değerlerini kullanarak ve belirli bir boyuttaki belirteçleri seçerek %98.8 tanıma oranı elde etmişlerdir [28]. Revilloud ve ark. yeni bir şerit algılama ve tahmin algoritması geliştirmişlerdir. Yoldaki şerit işaretçilerinin tahminini geliştirmek için gürültü filtreleri kullanmışlardır. Görüntü işleme ve şerit işaretleme algoritmalarını kullanarak görüntüdeki yolu %99.5 başarı ile tespit etmişlerdir [29]. Liu, Y. ve ark. karınca kolonisi algoritması ve yapay görme kullanarak bir robot

hareket yörünge kontrolü önermişlerdir. Deneysel araştırmalarla geliştirilen sistemin karmaşık ortamlarda makul yörünge planlaması yapabildiğini göstermişlerdir [30]. Lina ve ark. otomatik navigasyon sırasında konum doğruluğunu ve mobil robot hızını iyileştirmek için bir CNN kullanan Kalman tabanlı Tek Atışlı Çok Kutulu Dedektör (K-SSD) hedef konum stratejisi önermişlerdir. Deneysel sonuçlar, önerilen stratejilerinin, ışık farklılıkları ve sahne değişiklikleri gibi durumlarda aracın rota tespitini ve konum doğruluğunu arttırdığını göstermiştir [31]. Güneş ve ark. bir depoda otonom bir aracın yolunu bulmak için araçtaki kameradan alınan görüntüleri kullanmışlardır. Bu görüntülerle Daha Hızlı Bölge tabanlı CNN modelini eğitmişlerdir. Görüntüdeki etiket ve raf ayakları, eğitimli CNN, ters perspektif haritalama (IPM) algoritması ve destek vektör makinesi (SVM) algoritması kullanılarak tanımlanmıştır. Belirli alanlarda raf ayakları ve etiketler kullanarak kamera görüntülerinden sanal bir kılavuz elde etmişlerdir. Çalışmalarında gerçek zamanlı görüntü işleme veya navigasyon kullanmamışlardır [32]. Ryota ve ark. derin takviyeli öğrenmeyi kullanarak bir AGV taşıma sistemi geliştirmişlerdir. Ulaşım sistemlerinde rota planlaması için harita bilgilerini kullanmışlardır. Bir simülatör yardımıyla yöntemin geçerliliğini test etmişlerdir [33].

Şekil 2.1. WOS kategorileri ile elde edilen AVA analiz sonuçları [34].

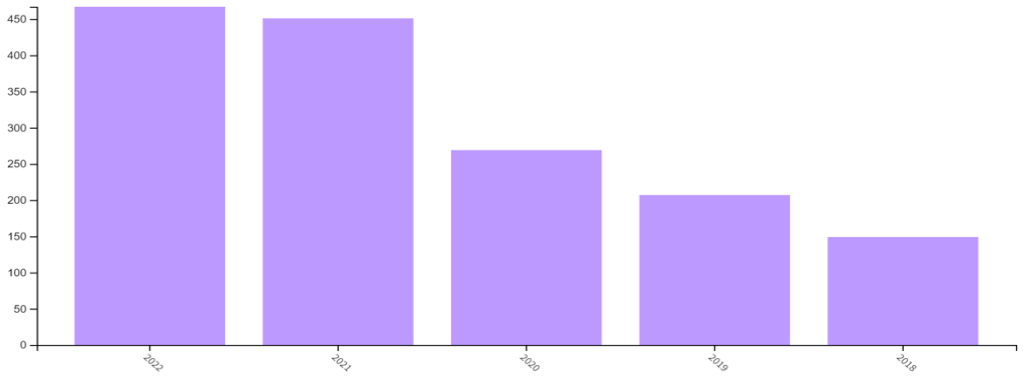
Literatür taramaları Web Of Science (WOS), Scopus, ScienceDirect ve Google Akademik kullanılarak ve son olarak 2022 yılı aralık ayı sonu itibariyle yapılmıştır. Literatür araştırmasıyla ilgili yapılan çalışmaların sonuçları aşağıda özetlenmektedir.

Literatür taraması “Autonomous vehicle OR AGV” (AVA) kelimeleriyle her bir ifade çift tırnak arasına alınarak WOS arama motoruyla yapıldığında ve WOS analiz

sayfasına geçilerek WOS kategorileri seçeneği seçildiğinde şekil 2.1' deki sonuçlar elde edilmiştir. AVA için bulunan toplam sonuç sayısı 2564 adettir. AVA ile ilgili çalışmaların %28.59' unun Ulaştırma Bilimi Teknolojisi, %23.91' inin Elektrik Elektronik Mühendisliği, %18.14' ünün Robotik alanlarında yapıldığı görülmektedir. Arama kriteri araştırma alanları olarak değiştirildiğinde şekil 2.2' deki sonuçlar elde edilmiştir. Bu sonuçlar incelendiğinde %58.44 ile Mühendislik, %29.17 ile Taşımacılık, %18.14 ile Robotik alanlarının öne çıktığı görülmektedir.

Şekil 2.2. WOS araştırma alanları kriteri ile elde edilen AVA analiz sonuçları [34].

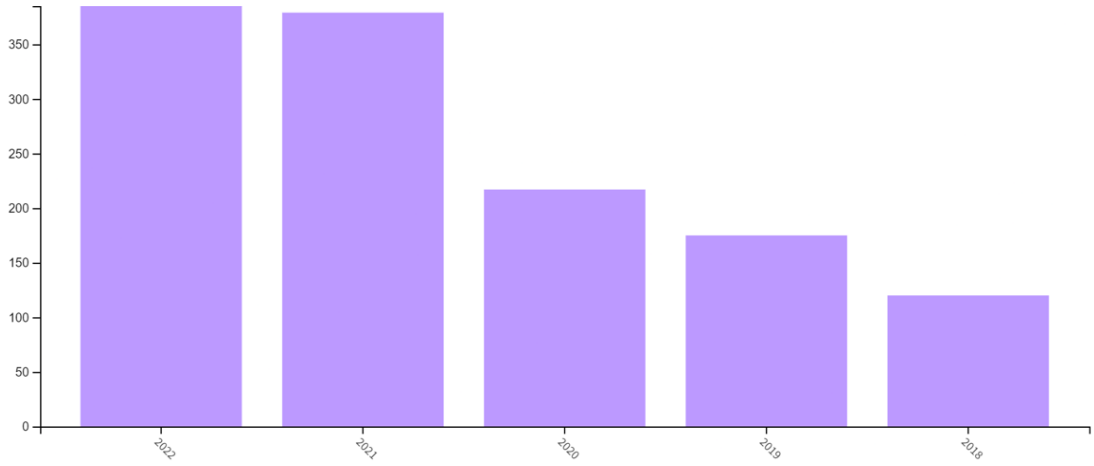
Yıllar bazında ise yapılan çalışmalar şekil 2.3' de verilmiştir. Grafikten görüldüğü gibi 2018 yılında AVA ile ilgili 149 çalışma bulunurken 2022 yılında çalışmalar 467 yayın sayına ulaşmıştır.



Şekil 2.3. WOS yıllar alanı kategorisi ile elde edilen AVA analiz sonuçları [34].

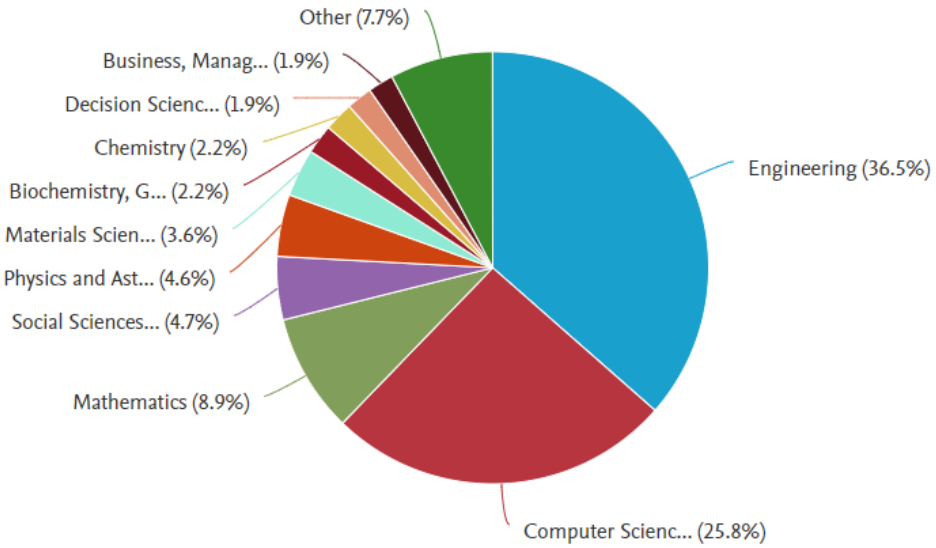
WOS arama motorunda her bir ifade çift tırnak arasına alınarak ve arama kriteri olarak "Autonomous vehicle OR AGV AND deep learning" (AVADL) yazılarak arama yapıldığında 1804 adet sonuç elde edilmiştir. Analiz sonuçlarından yayın yılları kriteri

seçildiğinde 2018 yılındaki yayın sayısı 120 iken 2022 yılındaki yayın sayısının 385 olduğu şekil 2.4' den görülmektedir.



Şekil 2.4. WOS yıllar alanı kriteri ile elde edilen AVADL analiz sonuçları [34].

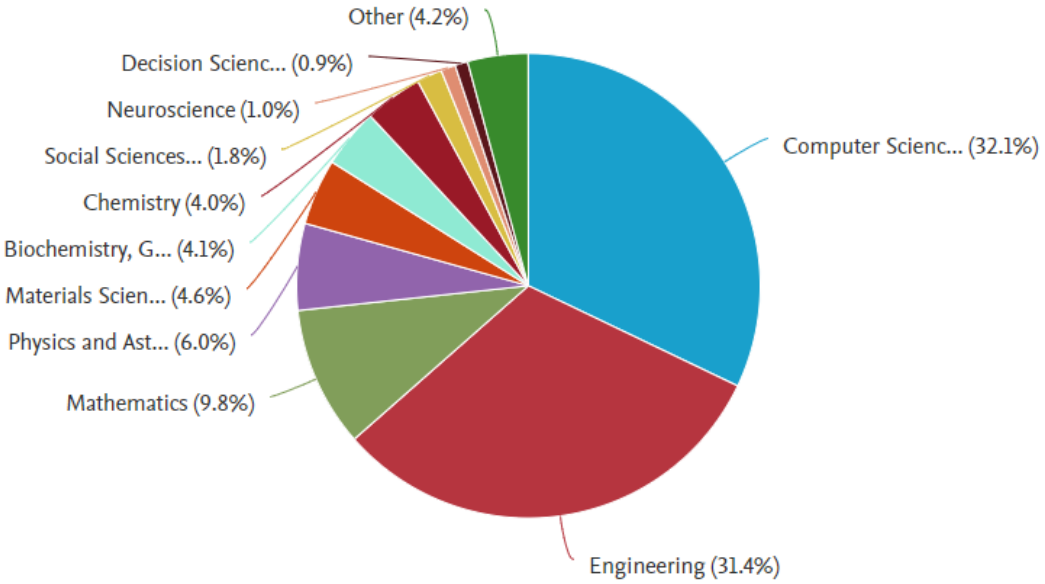
Scopus kullanılarak AVA araması yapıldığında 12272 adet sonuç elde edilmiştir. Bu sonuçlar incelendiğinde Mühendislik alanının %36.5 değeri ile birinci sırada, Bilgisayar Bilimleri alanının ise %25.8 ile ikinci sırada olduğu şekil 2.5' den görülmektedir.



Şekil 2.5. Scopus alanlara göre AVA analiz sonuçları [35].

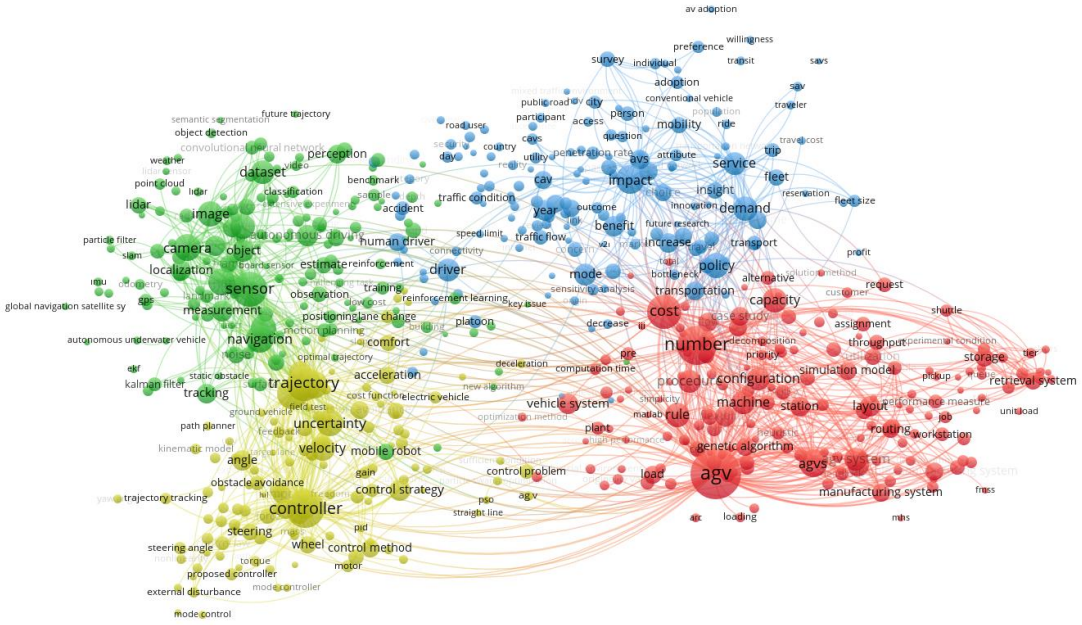
Yıllara göre Scopus AVA yayın sayısı 2014 yılında 186 iken hızlı bir yükselişe geçerek 2022 yılının sonunda 3271 değerine ulaşmıştır. Özellikle bu artış dikkat çekici bir yükselişe sahiptir. Arama AVADL için yapıldığında 1009 adet sonuç elde edilmiştir. Bu sonuç şekil 2.6' de yer aldığı gibi alanlara dağıldığında Bilgisayar Bilimleri alanındaki yayınların %32.10 ile birinci sırada Mühendislik alanındaki

yayınların ise %31.40 ile ikinci sırada bulunduğu görülmektedir. Yıllara göre yayın sayıları incelendiğinde 2015 yılında Scopus üzerinde 1 adet yayın bulunmasına rağmen 2022 yılındaki yayın sayısının 479 adete çıktığı görülmektedir.

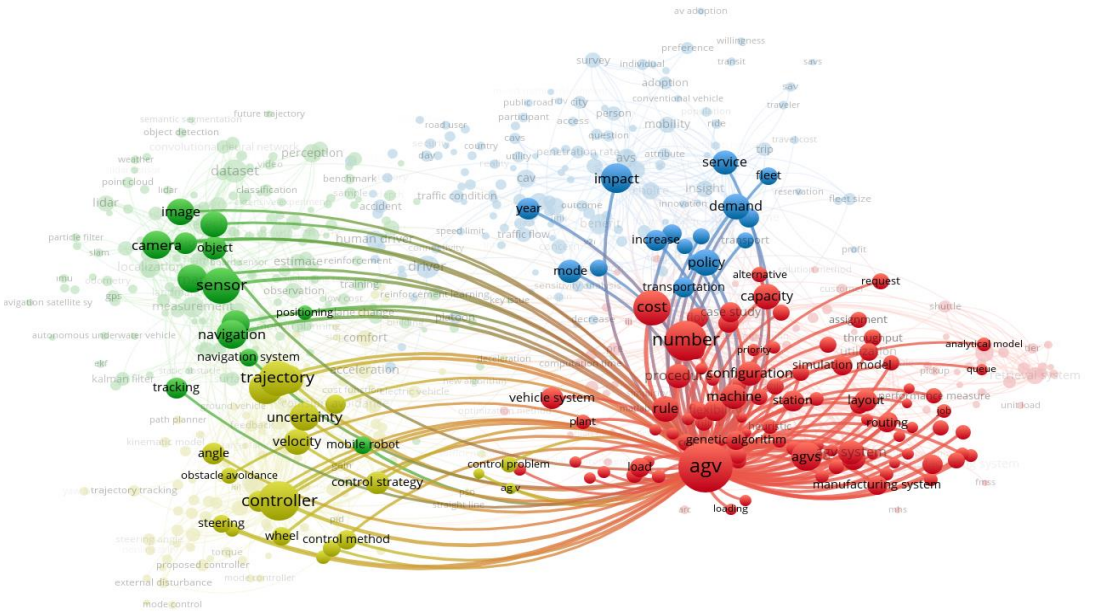


Şekil 2.6. Scopus alanlara göre AVADL analiz sonuçları [35].

WOS ve Scopus arama motorlarından AVA sonuçları indirilerek VOSviewer [36] yazılımında görsel analizler yapılmıştır. İlk olarak WOS' dan elde edilen AVA verileri programa yüklenmiş ve şekil 2.7' deki ekran görüntüsü elde edilmiştir. Şekil incelendiğinde öne çıkan kelimelerin diğerlerinden daha büyük daire şeklinde gösterildiği, renklerine göre dört farklı gruba ayrıldığı, ilk grupta agv (AGV), ikinci grupta yörünge (trajectory), üçüncü grupta servis (service) ve son grupta kamera (camera) gibi kelimelerin yer aldığı görülmektedir. Agv kelimesi ile çalışılmış alanlara bakıldığında şekil 2.8' de görüldüğü gibi dolaşım (navigasyon), izleme (tracking), sensör (sensor), taşıma (transportation), araç sistemi (vehicle system), kamera, kılavuz yol (guide path), yol planlayıcı (path planner) ve engel (obstacle) gibi alanlar ortaya çıkmaktadır.

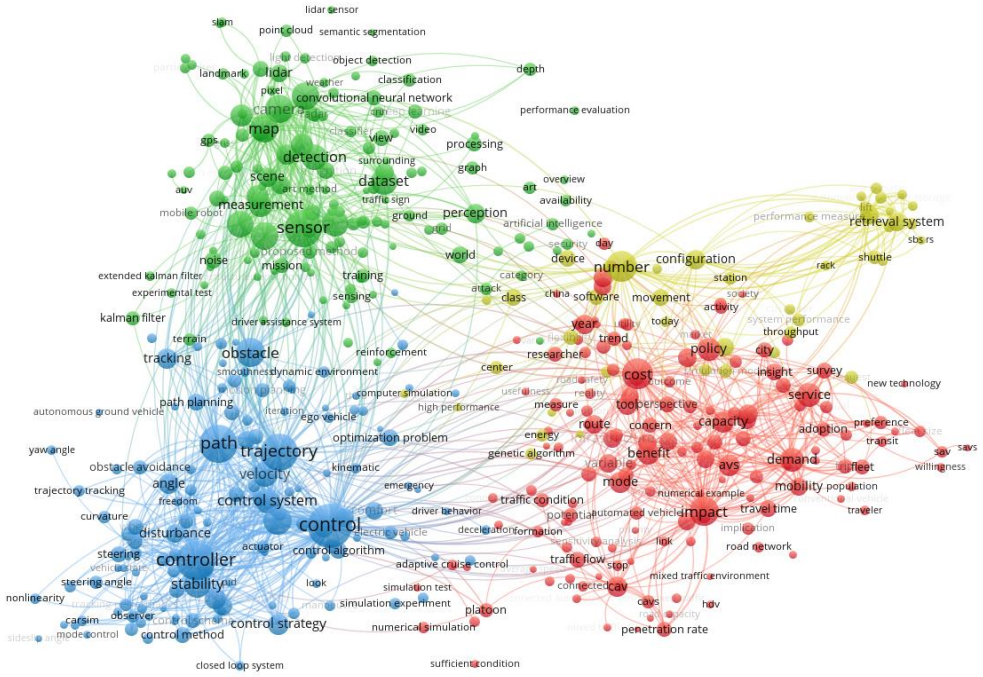


Şekil 2.7. WOS AVA verileri için VOSviewer ekran görüntüsü.



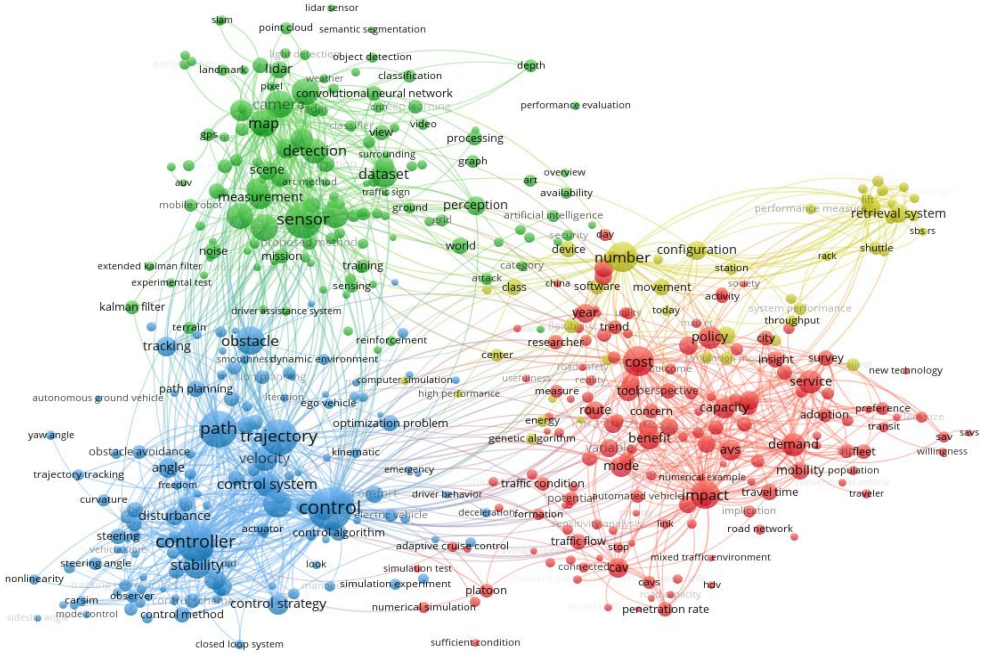
Şekil 2.8. WOS AVA verileri için ilişki alanları gösteren VOSviewer ekran görüntüsü.

WOS aramasında AVADL sonucunda elde edilen veriler VOSviewer' de incelendiğinde şekil 2.9' daki ekran görüntüsü elde edilmektedir. Şekil incelendiğinde kamera (camera), sensör (sensör), algılama (perception), engel (obstacle), izleme (tracking), tespit etme (detection), CNN, sınıflandırma (classification) gibi alanlardaki çalışmaların derin öğrenmedeki gelişmelerle birlikte daha çok öne çıktığı anlaşılmaktadır.



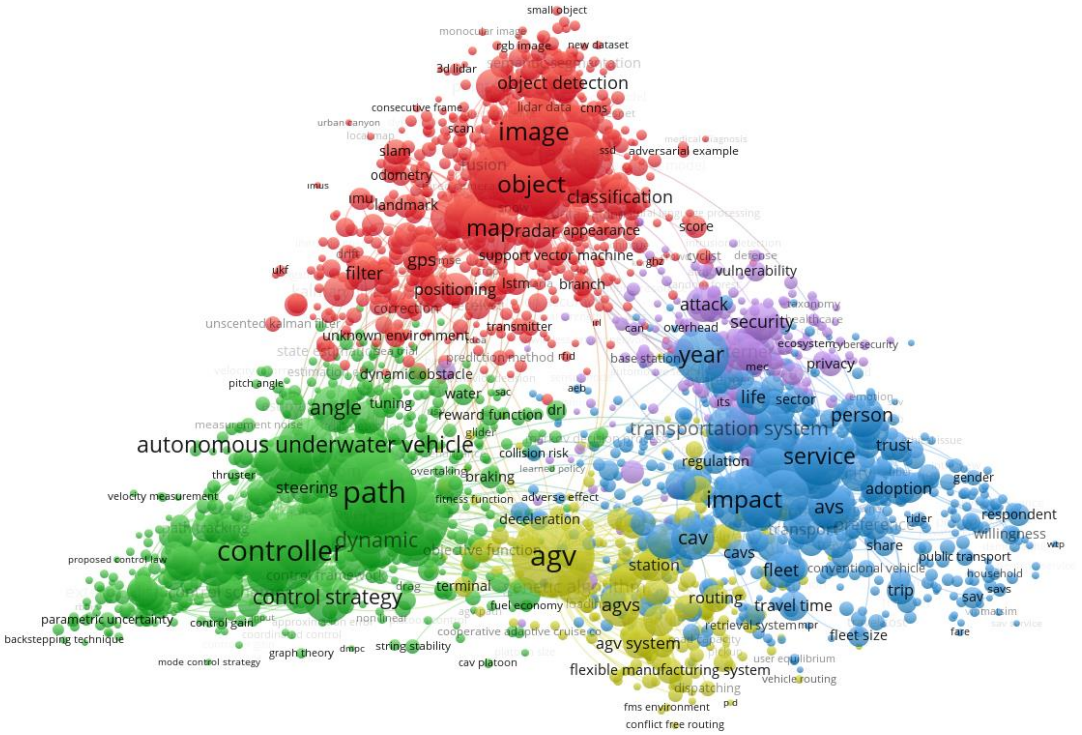
Şekil 2.9. WOS AVADL verileri için VOSviewer ekran görüntüsü.

WOS için yapılan AVA analizleri Scopus verileri için de yapıldığında şekil 2.10' daki görüntü ortaya çıkmaktadır.



Şekil 2.10. Scopus AVA verileri için VOSviewer ekran görüntüsü.

Scopus' da AVADL araması sonucunda elde edilen veriler VOSviewer kullanılarak analiz edildiğinde şekil 2.11' deki görüntü elde edilmektedir.



Şekil 2.11. Scopus AVADL verileri için VOSviewer ekran görüntüsü.

Görsel incelendiğinde derin öğrenmedeki gelişmeler ışığında yol (path), AGV (agv), görüntü (image), nesne (object), sınıflandırma (classification), nesne tanıma (object detection), derin öğrenme (deep learning), CNN (cnn), bilgisayar görmesi (computer vision) gibi alanların daha çok öne çıktığı anlaşılmaktadır.

Yukarıdaki analizlerden görüldüğü gibi otonom araçlar ve ilişkili olan konularla ilgili farklı alanlarda çok fazla çalışma yapıldığı görülmektedir. Bu sonuçlar ise bu çalışmanın yaygın olarak çalışılan zengin bir temele dayandığını ortaya koymaktadır. Bu tez çalışmasıyla bu alanlara katkıda bulunmak istemekteyiz.

3. MATERYAL VE YÖNTEM

3.1. Makine Öğrenmesi

Yapay zekâ (Artificial Intelligence), insan gibi olma veya insan davranışlarını taklit etme düşüncesiyle ortaya çıkmıştır. Makine öğrenmesi verilerden bilgi üreterek öğrenen algoritmaları içeren yapay zekanın alt dalıdır. Makine öğrenmesi verileri kullanarak öğrenebilmekte ve öngörüde bulunabilmektedir [37]. Veri sayısı arttıkça insanların bu verileri analiz etmesi ve bu verilerden sonuca gitmesi zorlaşmakta ve hatta imkânsız hale gelmektedir. Makine öğrenmesi verilerdeki özellikleri çıkarma yeteneğine sahip olmadığı için önce özellik çıkarma işlemi yapılmakta ve bu özellikler makine öğrenme algoritmalarına verilmektedir. Makine öğrenmesi sayesinde verilerin analiz edilmesi ve modellenmesi daha hızlı ve verimli bir şekilde gerçekleşmektedir.

Makine öğrenmesi genel olarak üç bölüme ayrılmaktadır. Bunlardan Denetimli Öğrenmede (Supervised Leaning) verilerdeki değerlerin hangi anlama geldiği etiketlerle önceden belirtilmektedir. Etiketler sınıf olarak adlandırılmaktadır. Girdi verileri ile eğitilen model genelleştirilmektedir. Eğitilmiş modele yeni veriler verildiğinde bu yeni verinin hangi sınıfa ait olduğu model tarafından bir hata değeriyle birlikte çıktı olarak verilmektedir. Denetimsiz Öğrenme (Unsupervised Learning) ikinci makine öğrenimi yöntemidir. Denetimli öğrenmedeki veri etiketleme bu öğrenme türünde yoktur. Bunun yerine veriler modele verilmekte ve veriler uzaklıklara göre kümelenerek hangi verilerin hangi sınıfa ait olduğu bulunmaktadır. Buradaki öğrenme işlemi veriye dayalı algoritmalar kullanılarak gerçekleştirilmektedir. Üçüncü makine öğrenimi yöntemi olan Pekiştirmeli Öğrenmede (Reinforcement Learning) model eğitilirken her doğru ve her hata için puan verilmektedir. Böylece modelin hataları ve doğruları kullanarak öğrenmesi sağlanmaktadır. Bir robota engelli bir ortamda hedefine giderken her engelde ceza ve her doğru ilerlemede ödül verildiğinde robot hedefine hatalardan öğrenerek gitmekte ve model eğitimi gerçekleşmektedir.

Makine öğrenmesinin birçok alanda kullanımı olmakla birlikte bu çalışmada sınıflandırma(classification), regresyon(regression) ve kümeleme (clustering) alanları kısaca açıklanacaktır.

Sınıflandırma görevinde amaç verilen girdinin iki veya daha fazla sınıftan hangisine ait olduğu belirlemektir. Eğitilmiş olan bir modele girdi verilir ve model bu girdiyi sınıflandırmaktadır. Sınıflandırma genel olarak spam algılama, görüntü tanıma gibi uygulamalarda kullanılmaktadır.

Regresyon bir nesnenin zaman içinde ürettiği verileri kullanarak gerçekleşmeyen bir zaman için yeni değerin elde edilmesinde kullanılmaktadır. Regresyon, hisse senedi gibi zaman serisi verilerinde kullanılmaktadır.

Kümeleme, bir veri setindeki benzer veya birbiriyle yakın özellikleri gösteren verilerin kümeler halinde otomatik olarak gruplanmasında kullanılmaktadır ve müşteri segmentasyonu gibi uygulamalarda kullanılmaktadır.

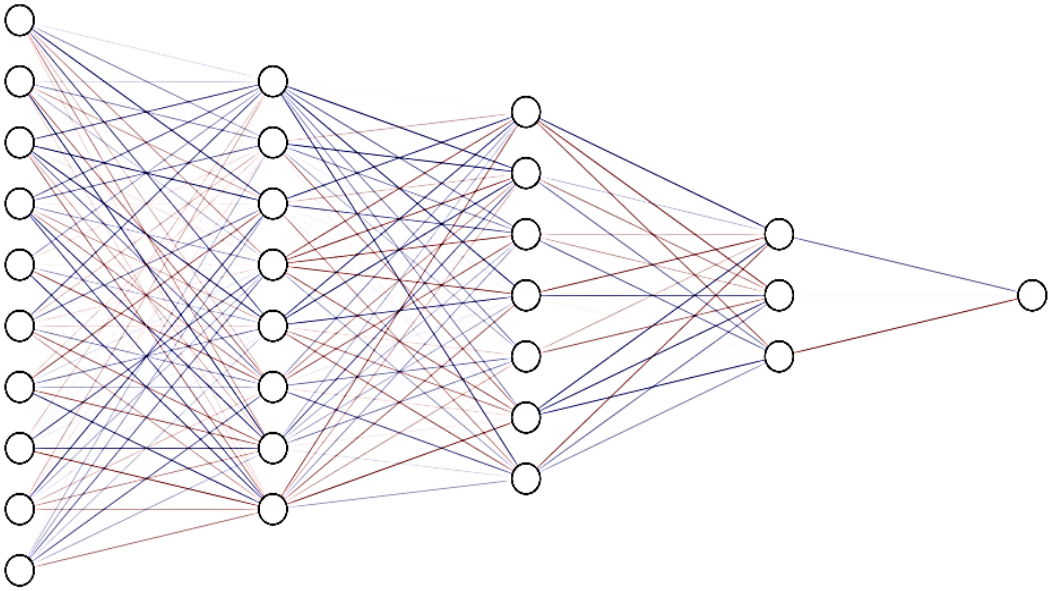
3.2. Derin Öğrenme

İnsan beyninden modellenen derin öğrenme, makine öğrenimi yaklaşımlarının bir alt sınıfıdır. Derin öğrenme teknik olarak çok katmandan oluşan sinir ağlarının kullanılmasıyla elde edilmektedir. Derin öğrenmenin merkezinde yapay sinir ağları vardır. Bir veri modelinin katman sayısı arttırıldıkça derin öğrenmeden bahsedilmektedir. Derin öğrenme algoritmalarında model her katmanda farklı özellikleri öğrenmektedir. Derin öğrenme katmanları ve nöron sayılarını örneği Şekil 3.1' de gösterilmektedir. Şekilden görüldüğü gibi derin öğrenme örnek modelinin girdi katmanı 10 adet, gizli katmanları sırasıyla 8,7 ve 3 adet nöron içermektedir. Derin öğrenme özellikle karmaşık problemleri çoklu katmanlar sayesinde çözebilmektedir [38, 39]. Derin öğrenmenin yapay zekâ içerisindeki yeri Şekil 3.2' de gösterilmektedir.

Derin öğrenme günümüzde teknolojinin gelişmesiyle ortaya çıkan büyük veri kavramı ile birlikte düşünülmektedir. Veriler ne kadar fazla ise derin öğrenme algoritmalarının başarısı da o oranda artmaktadır.

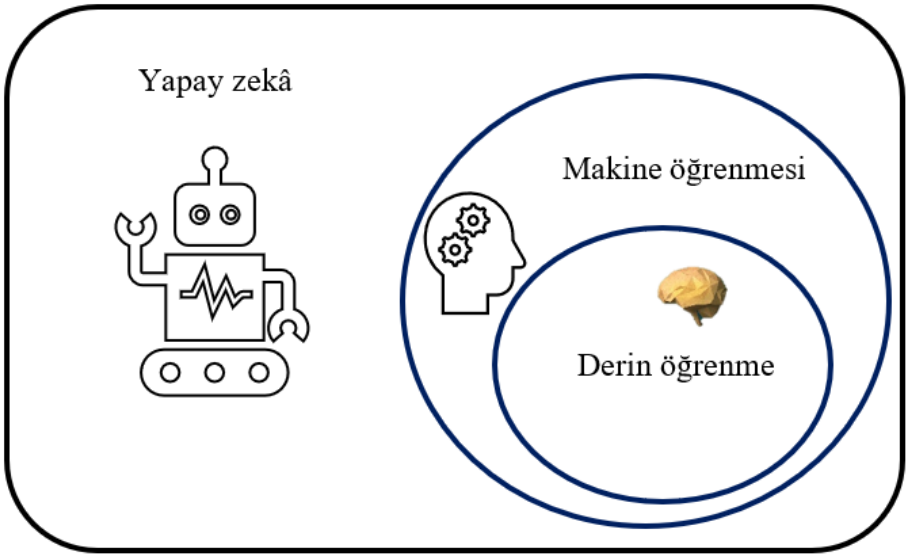
Derin öğrenmenin farklı mimarileri vardır. Görüntü sınıflandırma problemlerinde Evrişimli Sinir Ağları ve Tekrarlayan Sinir Ağıları (Recurrent Neural Network-RNN) en çok kullanılan mimarilerdendir. RNN' in özel bir mimarisi Uzun Kısa Dönemli

Bellek (Long Short-Term Memory- LSTM)' dir. Bu çalışmada kullanılan derin öğrenme mimarisi CNN' dir.



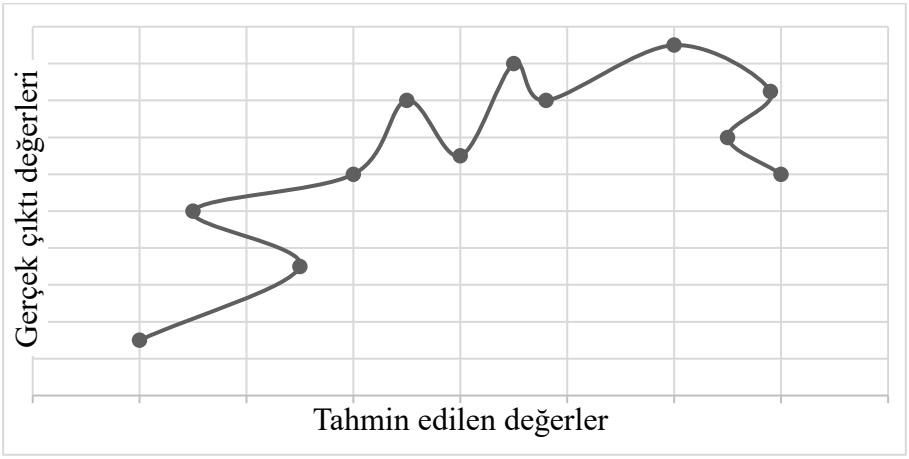
Şekil 3.1. Derin öğrenme örnek modeli katmanları ve nöronları.

Derin öğrenme ağlarında girdiler ile model eğitilirken modelin genelleştirilmesi önemli bir konudur. Şekil 3.3' de gösterilen modelin aşırı öğrenmesi (Overfitting) ve Şekil 3.4' de gösterilen modelin öğrenememesi veya eksik öğrenmesi (Underfitting) modelin genelleştirilmesine engel olan önemli kavramlardandır [40]. Ayrıca yanlılık/varyans dengesi (Bias/variance tradeoff) modelin genelleştirilmesinde göz önünde bulundurulması gereken bir konudur.



Şekil 3.2. Derin öğrenmenin yapay zekadaki konumu.

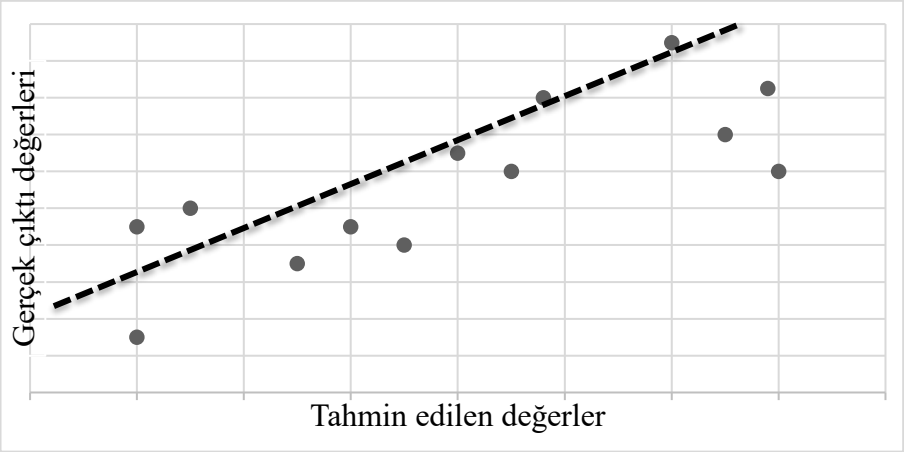
Verileri iyi öğrenen bir derin öğrenme modelinin hata oranı düşük olmalıdır. Bunun anlamı modelin kayıp fonksiyonu değerinin mümkün olduğu kadar az olmasıdır. Eğer eğitim veri setinin karmaşık ilişkileri varsa bu verilerle eğitilen model bu karmaşık ilişkilerin tamamını öğrenemeyebilmektedir. Modele bu karmaşık ilişkileri öğretmek için eğitim süresi uzun tutulabilmektedir. Bu durumda ise model eğitim verilerini çok iyi öğrenmesine rağmen doğrulama verilerinde çok kötü bir performans gösterebilmektedir. Bunun sebebi modelin eğitim veri setindeki birçok parametre arasındaki ilişkiyi öğrenmesi ama bu ilişkileri genelleştirememesidir. Bu sonuç modelin genelleştirme yeteneğini kaybettiğini gösterir ki buna aşırı öğrenme, aşırı uyum veya ezberleme denmektedir. Aşırı öğrenmede model eğitim verilerini ezberleyerek eğitim verilerinde çok iyi bir performans göstermekte ama hiç görmediği test verilerinde bu performansı devam ettirememektedir. Hatta test veri setindeki model performansı beklenmedik kötü bir başarınla sonuçlanabilmektedir. Aşırı öğrenen bir modelin eğitim eğrisi ile doğrulama eğrisi arasındaki fark eğitimin başından itibaren artarak bir sorun olduğunu göstermektedir [41, 42].



Şekil 3.3. Aşırı öğrenme (ezberleme) örnek grafiği.

Başarılı bir eğitimde eğitim sırasında eğitim ve doğrulama eğrilerinin birbirine yakın olarak çizilmesi model eğitiminin sorunsuz olarak gerçekleşmekte olduğunu belirten bir göstergedir. Eğitim veri setinin tek düze olması veya eğitimin çok uzun süre yapılması da aşırı öğrenmeye sorun olabilmektedir. Aşırı öğrenmiş bir modelde düşük yanlılık ve yüksek varyans durumu vardır. Veri setinde veri sayısı (satır sayısı) az ama özellik veya değişken sayısı çok fazla ise model karmaşıktır. Karmaşık bir model veri setindeki ilişkilerden ziyade gürültüleri öğrenmektedir. Eğitim sırasında modelin ne kadar karmaşık olduğunu tespit etmek aşırı öğrenmenin önüne geçen bir yöntemdir. Matematiksel açıdan bakılırsa model aslında sıfırdan farklı olan ağırlıklar (weight)

vektörüdür. Model karmaşıklığı ise bu ağırlıkların sayısıdır. Eğer sıfırdan büyük ağırlık sayısı azaltılabilirse model daha az karmaşık hale gelecek ve aşırı öğrenme engellenebilecektir [43, 44].



Şekil 3.4. Modelin eksik öğrenme örnek grafiği.

Veri setindeki değişken sayısını azaltmak aşırı öğrenme problemini çözmeye yardımcı olmaktadır. Değişken sayısını azaltmak için kullanılan yöntemlerden birisi Temel Bileşen Analizidir (Principal Component Analysis-PCA). PCA istatistiksel bir teknik olup boyut azaltmak için kullanılmaktadır [45]. PCA kullanılarak birbiri ile ilişkili bağımlı değişkenleri içeren eğitim verilerinden daha az sayıda ve kaldırılacak değişkenleri yüksek oranda temsil eden yeni değişkenler elde edilmektedir [46]. PCA uygulanan verideki pasif hale getirilen özellikler yığını temsil gücü çok az olan özelliklerdir. PCA ile veri üzerinde en fazla değişkenliğe sebep olan ve temel bileşen adı verilen değişkenler elde edilmektedir.

İyileştirme (Regularization) aşırı öğrenme sorununun önüne geçmek için kullanılan ve öğrenme sürecinin veri eksikliğini gidermek için eklenen yöntemlerden bir diğeridir. İyileştirme modelin başarımını negatif yönde değiştirmeden karmaşıklığını da azaltmak için kullanılan yararlı bir yöntemdir. İyileştirme algoritmaları derin öğrenme modellerinin eğitimini hızlandırmakta ve modelin ezberlemesini engellemeye çalışarak modeli genelleştirmektedir [47]. Bir derin öğrenme modeli iyileştirme uygulanarak eğitildiğinde modelin doğruluğu artabilmektedir. Derin öğrenmede yaygın olarak kullanılan iyileştirme teknikleri L1 (Least Absolute Shrinkage and Selection Operator-LASSO), L2 (Ridge) ve nöron seyreltme (dropout)'dir [48].

L2 iyileştirmede, modelin ağırlıkların mutlak değerleri yerine kareleri alınarak toplanmakta ve modelin karmaşıklığı elde edilmektedir. L2 iyileştirme model ağırlıklarını en az seviyede tutmaya zorlamaktadır. Yani ağırlık değerleri sıfıra doğru yaklaşmakta ama tam olarak sıfır olmayabilmektedir. Denklem 3.1'de gösterilen ve maliyet fonksiyonuna eklenen bir iyileştirme terimi ile Ridge regresyonu elde edilmektedir. Ridge regresyon formülü denklem 3.2'de verilmiştir. Denklemdeki λ sıfır veya daha büyük değer alabilen bir ayar parametresidir. $\lambda = 0$ olduğunda maliyet teriminin etkisi yok olmakta ve Ridge regresyon en küçük kareler tahminini üretmektedir. Eğer λ sonsuza yaklaşırsa maliyet etkisi büyümekte ve Ridge regresyon katsayı tahminleri sıfıra yaklaşmaktadır. β değerleri model sabitleridir [49].

$$\lambda \sum_{J=1}^{P} \beta_j^2 \tag{3.1}$$

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{J=1}^{p} \beta_j^2 \tag{3.2}$$

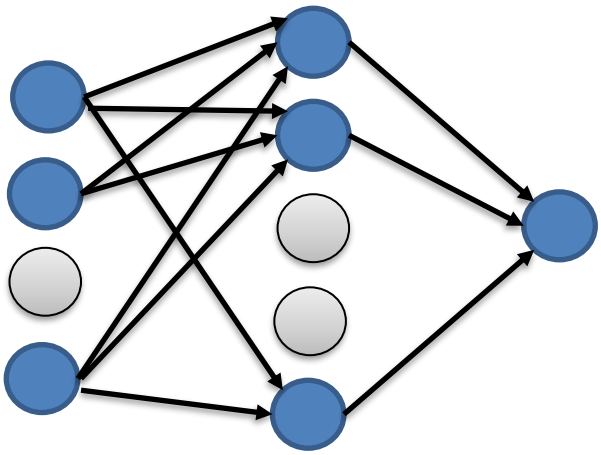
L1 iyileştirmede modelin karmaşıklığı, modelin ağırlıkların mutlak değerlerinin toplamı alınarak elde edilmektedir. L1' de L2' den farklı olarak ağırlık değerleri genelde sıfır olmaktadır. L1 iyileştirmede maliyet fonksiyonuna β katsayılarının mutlak değerlerinin toplamı eklenmektedir. Burada maliyet fonksiyonuna denklem 3.3. deki iyileştirme terimi eklenmektedir. Denklem 3.4, LASSO işlevini göstermektedir. LASSO verilerdeki çok az öneme sahip olan özniteliklerin ağırlıklarını dikkate almayarak öznitelik seçimi yapmaktadır [49].

$$\lambda \sum_{J=1}^{P} |\beta_j| \tag{3.3}$$

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{J=1}^{P} |\beta_j| \tag{3.4}$$

Nöron seyreltme, model eğitimi yapılırken bir P olasılıkla katmanlardaki nöronların pasif hale getirilerek kullanılmamasını ifade etmektedir. Şekil 3.5' de gösterildiği gibi, model eğitiminde nöron seyreltilmesi yapılarak bazı nöronlar devre dışı bırakılmaktadır. Nöron seyreltmede kullanılan P olasılığı 0 ile 1 arasında değişmektedir ve her ileri yayılım ile ağırlık güncelleme işleminde uygulanmaktadır. Olasılık değerine bire yakın bir değer atamak model performansını olumsuz yönde etkilemektedir. Bu olasılık değeri genel olarak 0.25 olarak kulanılmaktadır. Nöron seyreltme genel olarak tam bağlantılı katmanlardaki bağları kopartmak için kullanılarak nöronların ağırlık değişimlerinin birbirinden etkilenmemesini sağlamaktadır. Bu çalışmada kullanılan evrişimli sinir ağının eğitimi esnasında tam bağlantılı katmanından sonra aşırı uyumu engellemek için ayrıntıları ilgili bölümde belirtilen nöron seyreltme kullanılmıştır.

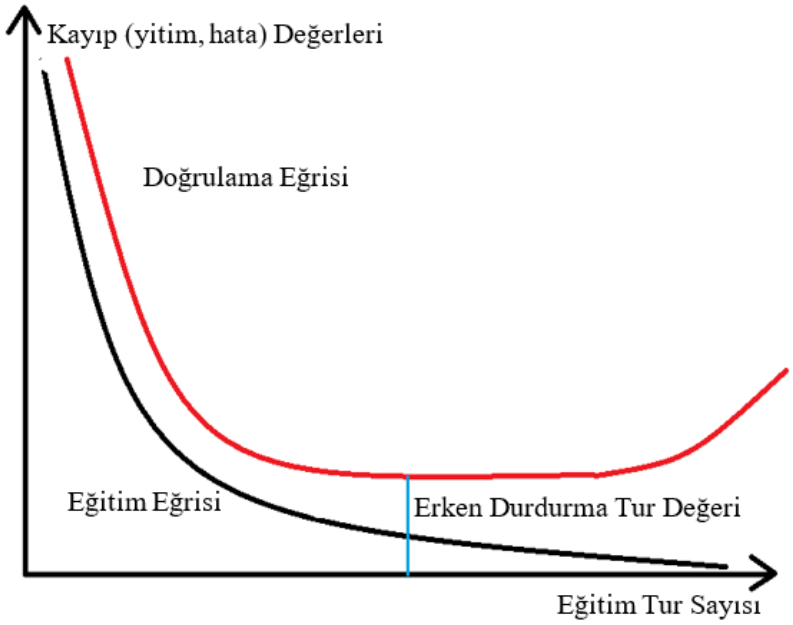
Özellikle evrişimli sinir ağlarındaki evrişim katmanlarında yığın normalizasyon (Batch normalization) yöntemi kullanılarak model performansı arttırılabilmektedir. Normalizasyon kullanılarak verilerin dağılımı ortalaması sıfır ve standart sapması bir olan standart normal dağılıma ve tüm veriler -1 ve +1 arasına dönüştürülmektedir. Yığın normalizasyon kullanıldığında gradyanların yok olmasının önüne geçilmektedir. Ayrıca daha yüksek öğrenme oranı kullanılabilmektedir. Yığın normalizasyon kullanıldığında kayıp değerleri (loss) oldukça azalmaktadır.



Şekil 3.5. Nöron seyreltme.

Aşırı öğrenmeyi engellemek için kullanılan diğer bir yol da erken durdurma (Early Stopping) yöntemidir. Bu yöntemde derin öğrenme modeli eğitilirken elde edilen kayıp grafiğindeki eğitim ve doğrulama eğrileri incelenmektedir. Şekil 3.6' da gösterildiği gibi eğitim eğrisi ile doğrulama eğrisinin yönü birbirinin tersine hareket

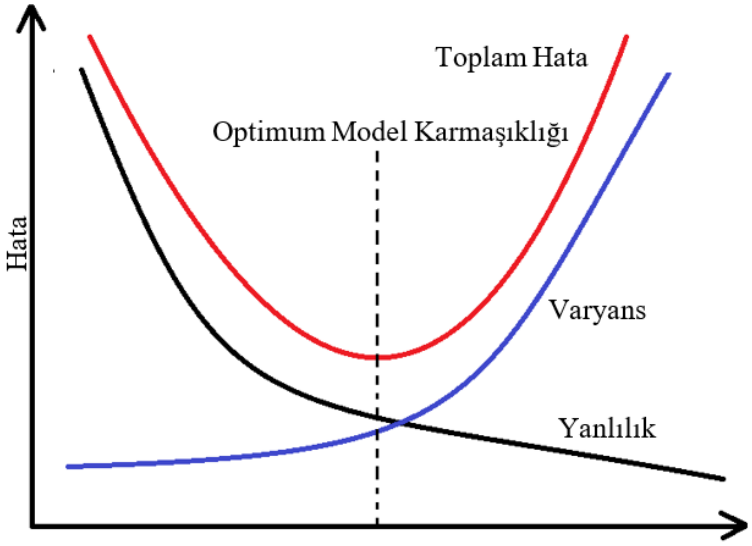
etmeye başladığında eğitim hatası azalırken doğrulama hatası artmaktadır. Bu durumda aşırı öğrenmeyi engellemek için eğitim tur sayısı (epoch) değeri belirlenerek eğitim durdurulmaktadır. Yeni yapılacak model eğitiminin tur sayısı grafik üzerinden elde edilen tur sayısı olarak ayarlanarak model eğitimine başlanmaktadır [50, 51].



Şekil 3.6. Erken durdurmanın belirlenmesi.

Derin öğrenme modelinin öğrenememesi verilerin eğitim için yetersiz kaldığını ve model doğruluğunun çok düşük olacağını göstermektedir. Model eğitimi için verilerin yetersiz olması, kurulan model yapısının verilerle uyumsuz olması veya öznitelik sayılarının az olması eğitimin başarısız olmasına sebep olmaktadır. Ayrıca model genelleştirilememektedir. Yani girdi verileri ile çıktılar arasındaki ilişkiler model tarafından yeterli olarak öğrenilememektedir. Bu tip modellerde yanlılık yüksektir ama varyans düşüktür. Yanlılığın yüksek olması modelin verilere uymadığının bir göstergesidir. Öğrenememiş bir modele eğitim ya da test verilerinden alınan girdiler verildiğinde doğruluk çok düşük çıkmaktadır. Modelin eksik öğrenmesinin önüne geçmek için eğitim tur sayısı arttırılabilmekte, veri sayısı arttırılabilmekte, verilerdeki ayrık (uç) değerler ile gürültü temizlenebilmekte ve modelin karmaşıklığı arttırılabilmektedir [44].

Bir modelin yanlılık eğrisi ile varyans eğrisi arasındaki ilişkiye bakılarak modelin karmaşıklık drumu anlaşılabilmektedir. Modelin yanlılık ve varyans grafiği modelin karmaşıklığını vermektedir. Model karmaşıklığı grafiği Şekil 3.7'de verilmiştir.



Şekil 3.7. Model karmaşıklığı.

Modelin yanlılığı düşükse ve varyans da düşükse model genel olarak tutarlıdır. Yanlılık düşük ama varyans büyükse ortalama olarak tutarsız bir model elde edilmektedir. Yanlılık yüksek ve varyans da yüksekse elde edilen model tutarsız ve aynı zamanda hatalıdır. Yanlılık yüksek ama varyans düşükse ortalama olarak hata oranı yüksektir ama model tutarlıdır. Şekil 3.7' den de görüldüğü gibi aşırı öğrenme ve eksik öğrenmenin önüne geçmek için gerekli olan model karmaşıklığı yanlılık ve varyans dengesinde toplam hatayı minimum seviyede tutmalıdır.

3.2.1. Derin öğrenme ağlarında görüntü sınıflandırma

Görüntü tabanlı sınıflandırma yapılırken piksel piksel tahmin yapmak yerine her bir görüntü veya bu görünntüden çoğaltılmış benzer görüntüler için bir etiket atanmaktadır. Kameradan elde edilen görüntü işlenerek sınıflandırılacak olan ilgili alan görüntü işleme yöntemleriyle elde edilmektedir. Elde edilen bu görüntü derin öğrenme modeline girdi olarak verilerek sınıflandırma yapılmaktadır. Görüntü sınıflandırma problemini çözmek için kullanılan derin öğrenme tabanlı sınıflandırırıcılar CNN tabanlı sınıflandırıcılar, CNN-RNN tabanlı sınıflandırıcılar, Otomatik kodlayıcı (Auto-encoder) tabanlı sınıflandırıcılar ve Çekişmeli üretici ağ (Generative adversarial networks-GANs) tabanlı sınıflandırırcılardır.

CNN tabanlı sınıflandırıcılardan ilki LeNet [52] modelidir. LeCun ve ark. tarafından 1988 yılında el yazısı rakamlarını sınıflandırmak için geliştirilmiştir. Bu model evrişim bloğunun üç katmanı, ortalama havuzlama bloğunun iki katmanı, bir adet tam bağlı katman ve bir adet çıktı katmanını içeren yedi ağırlık barındıran katmandan

oluşmaktadır. Modeli eğitmek ve test etmek için MNIST (Modified National Institute of Standards and Technology) el yazısı rakam veri seti kullanılmıştır. Modeli eğitmek için yirmi iterasyon, 0.02 değerli momentum ile her bir iterasyonda daha düşük bir öğrenme oranı kullanılmıştır. Öğrenme için SGD (Stochastic Gradient Descent-Stokastik Gradyan İnişi) kullanılmıştır.

AlexNet [53] Krizhevsky ve ark. tarafından 2012 yılında görüntü sınıflandırma için geliştirilmiş, LeNet ağının tasarımıyla aynı olan fakat sikiz ağırlık katmanından oluşan daha büyük olan bir CNN tabanlı modeldir. Model eğitiminin daha hızlı olması için evrişim ve tam bağlantılı katmanlardan sonra ReLU fonksiyonu kullanılmıştır. Modeli eğitiminde ImageNet görüntü veri seti kullanılmıştır. Model eğitimi için bu veri setindeki farklı çözünürlüğe sahip görüntülerin boyutu 256 x 256 piksel boyutuna ayarlanmıştır. Bu mimaride belirlenen bin adet sınıf sayısı LeNet mimarisinden çok fazladır. Modelin ezberleme sorununu azalmak için PCA (Temel bileşen analizi) ve veri artırma içeren düzenliliştirme yöntemleri kullanılmıştır. Eğitime verilen parça eğitim boyutu (Mini-Batch Size) 128 ve momentum 0.9 olarak alınmıştır.

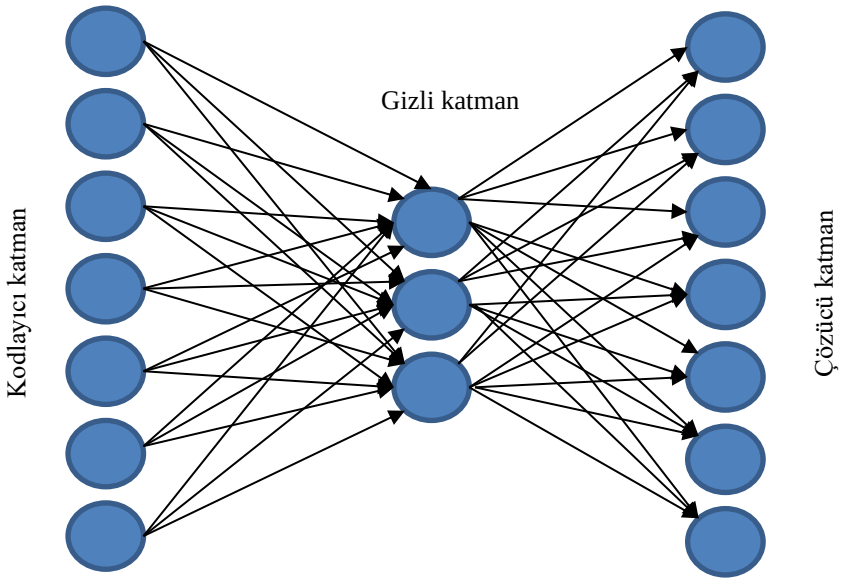
AlexNet modelinin daha derin konfigürasyonu kullanan Simonyan ve Zisserman VGGNet [54] modelini geliştirmişlerdir. Bu modelde evrişim fitresi ağ boyunca 3 x 3 olarak sabitlenmiştir. Eğitim için girdi görüntü boyutları 224 x 224 boyutuna ayarlanmıştır. A, A-LRN, B, C, D ve E olarak beş adet CNN konfigürasyonu kullanılmıştır. Burada D VGG16, E ise VGG19 olarak adlandırılmıştır. Her bir konfigürasyon sırasıyla 11, 22, 13, 16 ve 19 ağırlık katmanına sahiptir. Eğitim için parça eğitim boyutu 256 ve momentum değeri 0.9 olarak kullanılmıştır. Düzenlileştirme için L2 ve nöron seyreltme kullanılmış ayrıca eğitim öğrenme oranı 0.02 olarak alınmıştır.

Geleneksel CNN mimarisinden faklı bir bir mimari olan ve kod adı Inception (Başlangıç) olan yirmi iki katmanlı GoogLeNet [55] mimarisini Szegedy ve arkadaşları geliştirmişlerdir. Her bir evrişim katmanında paralel filtreler olan ve inception olarak adlandırılan 1 x 1, 3 x 3 ve 5 x 5 boyutlarında fitreler kullanılmıştır. Böylece her bir konvolüsyon katmanındaki birim sayısı arttırılmıştır. Bu modeli eğitmek için dağıtık makine öğrenimi çerçevesi kullanılmıştır. Momentum değeri 0.9 ve asenkron SGD kullanılmıştır. Öğrenme oranı her sekiz tur sayısında yüzde dört azaltılarak eğitim gerçekleştirilmiştir. Eğitim için görüntü boyutu 224 x 224 olarak alınmıştır.

Geleneksel CNN mimarisindeki gizli katmanların sayısındaki artışın fazla olması aktivasyon fonksiyonunun çıktı değerinin çok fazla sıfır olmasını sağlamakta ve kaybolan gradyan (vanishing gradient) sorunu ortaya çıkmaktadır. Bu sorunu ele alan He ve ark.VGG ağlarından sekiz kat fazla olan ve 152 katmandan oluşan ResNet [56] mimarisini önermişlerdir. Eğitimde 128 parça eğitim boyutu ve SGD kullanılmaktadır. Öğrenme oranı 0.1 değerinden başlatılmaktadır. Momentum değeri 0.0001 lik ağırlık düşüşü ile 0.9 olarak alınmaktadır.

Evrişimli ağların girdiye ve çıktıya yakın katmanlar arasında daha kısa bağlantılar içermesi durumunda eğitilmesinin önemli ölçüde daha derin, daha doğru ve daha verimli olabileceği düşüncesine benimseyen Huang ve ark. DenseNet [57] mimarisini önermişlerdir. Bu mimaride her katman diğer tüm katmanlara ileri beslemeli olarak bağlanmaktadır. Önceki katmanların özellik haritaları mevcut katman için girdi olarak kullanılmaktadır. Ayrıca önceki katmanların özellikleri özellik kalitesini iyiliştirmek ve kaybolan gradyan sorununu en aza indirmek için kullanılmaktadır. Bir yoğun (dense) bloktaki doğrusal olmayan transformasyon fonksiyonları 3 x 3 evrişim işlemi, ReLU ve toplu normalleştirmenin (batch normalization) bir kombinasyonudur. 1 x 1 boyutunda bir darboğaz (bottleneck) fonksiyonu kullanılarak boyutsallık en aza indirilmektedir. Farklı veri setlerinde 64 ve 256 gibi farklı parça eğitim boyutu kullanılmıştır. Optimize için SGD kullanılmaktadır. Öğrenme oranı 0.1 değerinden başlatılmakta ve sonra 1/10 oranında azaltılmaktadır. Momentum olarak Nesterov momentumu ve nöron seyreltme olarak 0.2 oranı kulanılmaktadır.

CNN-RNN tabanlı görüntü sınıflandırmasında CNN girdi görüntülerinin ayırt edici temel özelliklerini çıkarılmaktadır. CNN ile RNN mimarisinin beraber kullanılmasıyla da hiyerarşik etiketlerin bağımlılığı verimli olarak manüpüle edilerek ince va kaba sınıflar için iyileştirilmiş sınıflandırma sornuçları elde edilmektedir. Böylece CNN-RNN mimarisi beraber kullanılarak CNN ile elde edilen sınıflandırmadan daha iyi performans gösteren sonuçlar ortaya koymaktadır. RNN ile ara katman çıktısının bağımlılık özellikleri hesaplanmakta ve sınıflandırma için bu ara katmanların özellikleri son tam bağlantılı katmana bağlanmaktadır. Ayrıca Fourier dönüşümü ve



Şekil 3.8. Temel bir oto kodlayıcı mimarisi örneği

dalgacık dönüşümü (Wavelet transform- WT) kullanılarak girdi verileri fitrelenmekte böylece gradyan patlaması veya gradyan kaybolması sorunlarının önüne geçilmektedir [58–61].

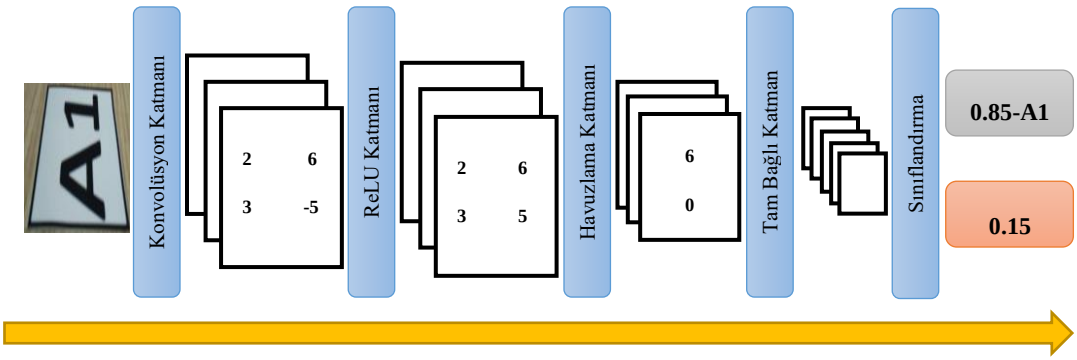
Bir otomatik kodlayıcı girdi verisini sıkıştırıp sonra bu sıkıştırılmış veriden tekrar girdi verisini elde etmek için kullanılan ve kodlayıcı ile kod çözücüden meydana gelen iki ağdan oluşan bir yapay sinir ağı mimarisidir. Otomatik kodlayıcılar girdi verilerinin boyutunu küçülttüğü için derin öğrenmede ön işleyici olarak kullanılmaktadırlar. Bir otomatik kodlayıcıya eğitim sırasında görüntü verisi bozuk bir nesnenin farklı açıdan görüntüleri verilerek görüntülerdeki nesnenin bozuklukları giderilmiş bir gerçek nesne görüntüsü elde edilebilir. Bu işlemde modele hangi görüntünün bozuk hangi görüntünün düzgün olduğu öğretilmemektedir. Mimari, görüntü verilerindeki kalıpları kendi kendine öğrenmektedir. Otomatik kodlayıcılar Şekil 3.8’ de görüldüğü gibi kodlayıcı (encoder), bellek (hidden space, bottleneck layer) ve kod çözücü (decoder) olarak üç katmana ayrılmaktadırlar. Kodlayıcı katmanındaki nöron sayısı ile kod çözücü katmanındaki nöron sayısı aynıdır. Kodlayıcı çok boyutlu olan girdi verilerinin boyutunu azaltmakta ve bu veriyi gizli katmanda tutmaktadır. Gizli katmandaki boyutu azaltılmış bu veri ise çözücü katmanda girdi katmanıyla aynı boyutta yeniden oluşturulmaktadır. Otomatik kodlayıcılar görüntü verisindeki en önemli özellikleri çıkarmakta ve bu özellikler sınıflandırmada kullanılmaktadır [62-64].

GANs [65] eğitim verilerini inceleyerek bu verilerin olasılık dağılımını öğrenen, daha sonra bu dağılımı kullanarak daha fazla yeni benzer veri üreten bir derin öğrenme

algoritmasıdır. GANs birbirine ters olarak çalışan üretici (generator) ve ayrıştırıcı (discriminator) olarak bilinen iki alt yapay sinir ağı birleşiminden oluşmaktadır. Üretici rastgele ağırlıkları kullanarak sahte veriler üreterek ayrıştırıcıya göndermekte ve ayrıştırıcı bu verileri gerçeği ile karşılaştırarak elde ettiği sahtelik derecesine göre bir yitim değeri üretmekte, bu değeri geri besleme ile üreticiye göndermektedir. Üretici gelen yitim değerine göre ağırlık değerlerini sürekli güncelleyerek yeni ve gerçek veriye daha çok benzeyen bir üretim yapmaktadır. Bu üretim ve ayırıcı döngüsü, her biri için ağırlık değerleri güncellenerek üretilen verinin gerçek veri olarak sınıflandırılmasına kadar devam etmektedir [66, 67].

3.2.2. Evrişimli sinir ağları

CNN, yüksek doğrulukta görüntü sınıflandırması sağlayan bir derin öğrenme algoritmasıdır. CNN algoritmaları görüntü tanıma problemlerinde çok fazla kullanılmaktadırlar. CNN modelindeki her bir katman görüntüdeki farklı özellikleri öğrenmektedir. CNN modellerinde önce özellikler öğrenilmekte sonra ise bu özellikler sınıflandırılmaktadır. Geleneksel CNN, Şekil 3.9' da gösterildiği gibi genellikle dört katmandan oluşmaktadır. Bunlar evrişim (convolution, konvolüsyon), ReLU, havuzlama (pooling) ve tam bağlantılı (Fully connected) katmanlardır. Evrişim katmanı, giriş katmanındaki verilere filtreler (kernel) uygulayarak bir özellik haritası çıkarmaktadır. ReLU katmanı, evrişim katmanındaki verilerin negatif değerlerini ortadan kaldırmak için kullanılmaktadır. Havuzlama katmanı, verilerin temel özelliklerini korurken giriş boyutunu küçültmektedir. Tam bağlantılı katman, tamamen birbirine bağlı nöronlardan oluşan bir sinir ağı katmanıdır ve görüntüyü sınıflandırmak için kullanılmaktadır [53, 68, 69]. Sınıf sayısı ikiden fazla olduğunda Softmax aktivasyon fonksiyonu sınıflandırma için kullanılmaktadır.



Şekil 3.9. Genel CNN topolojisi.

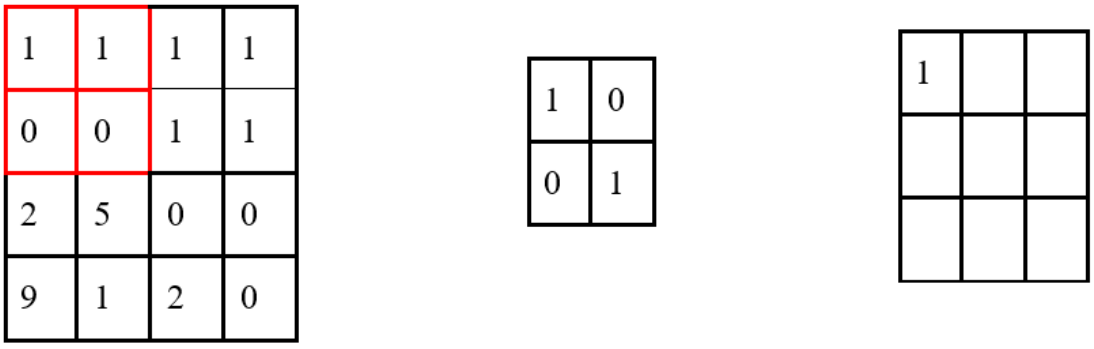
Görüntüler piksel olarak isimlendirilen parçacıklardan oluşmaktadır. Filtreler bu piksellere uygulanmaktadır. Filtreler piksellerden oluşan matriste kaydırılarak ve genel olarak soldan sağa yönlü olarak uygulanmaktadırlar. Filtreler görüntüye uygulanırken kaydırma adımı genel olarak bir alınmakla birlikte kaç adım kaydırılacağı da belirtilebilmektedir. Şekil 3.10' da bir konvolüsyon işlemi ve denklem 3.5, 3.6, 3.7, 3.8' de hesaplama örnekleri gösterilmiştir.

$$1x1+1x0+0x0+0x1 = 1 \quad (3.5)$$

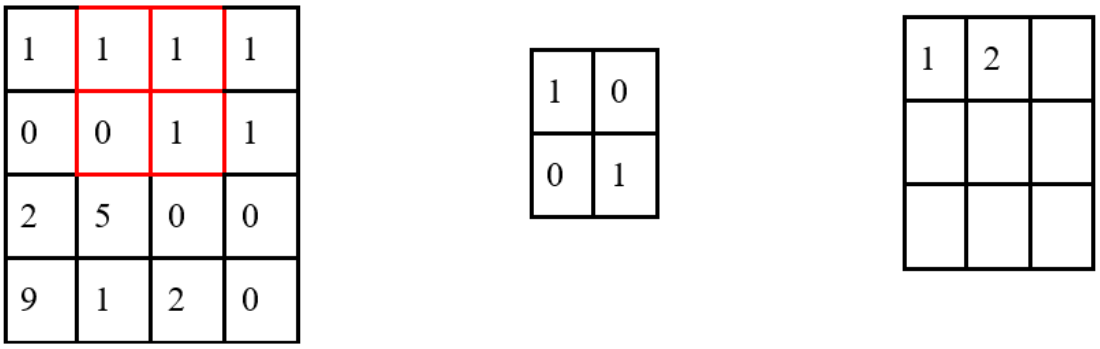
$$1x1+1x0+0x0+1x1 = 2 \quad (3.6)$$

$$1x1+1x0+1x0+1x1 = 2 \quad (3.7)$$

$$0x1+0x0+0x2+5x1 = 5 \quad (3.8)$$

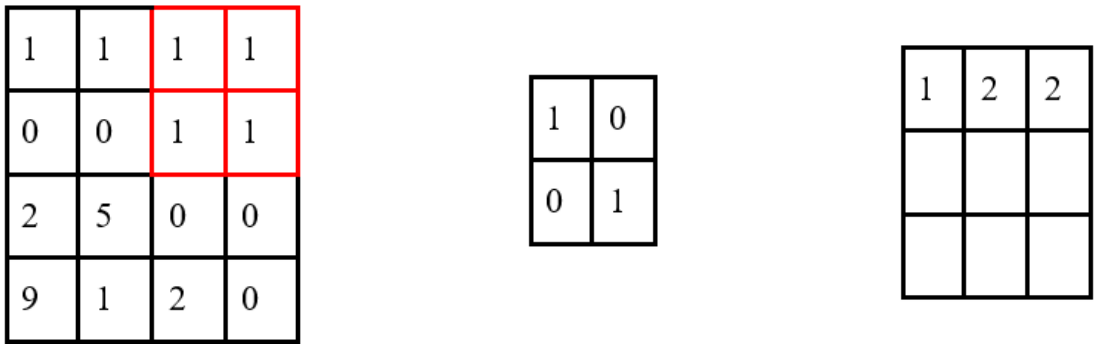


a) Adım 1

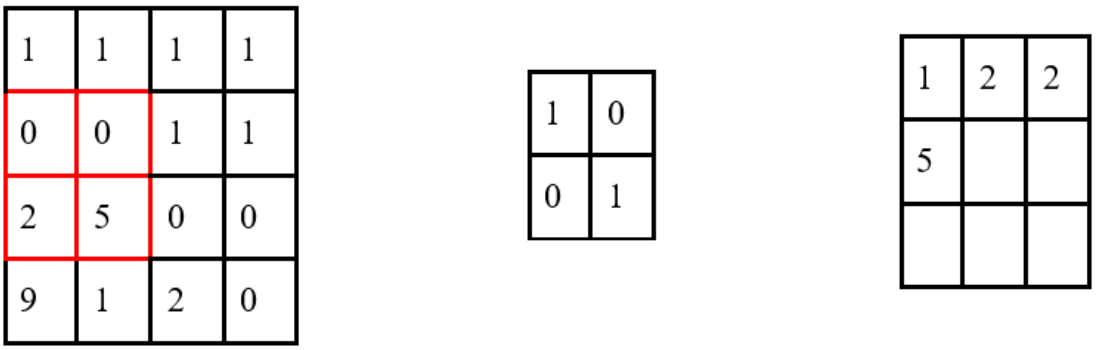


b) Adım 2

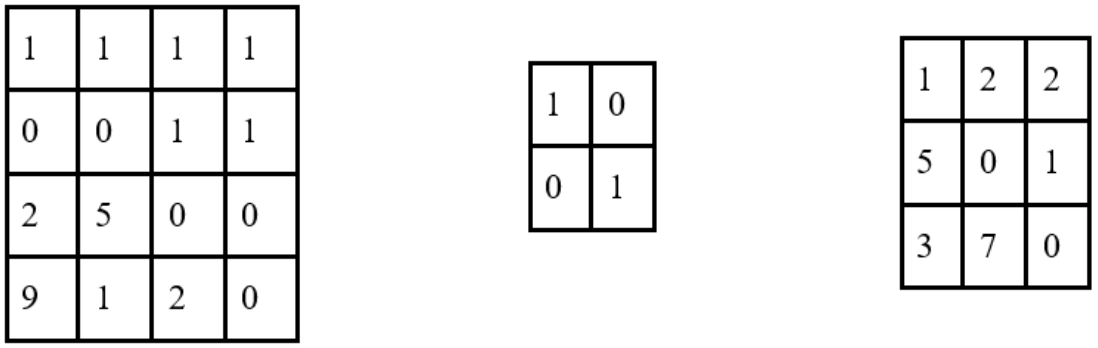
Şekil 3.10. Konvolüsyon işlemi.



c) Adım 3



d) Adım 4

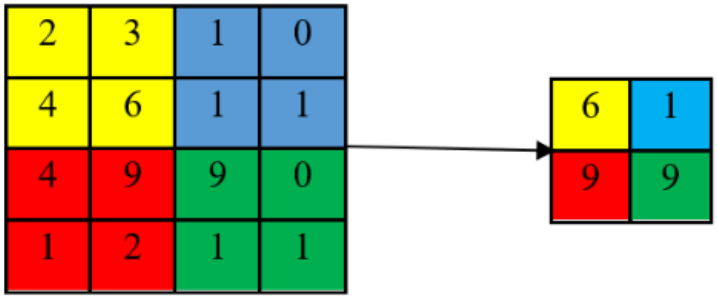


e) Son adım

Girdi Filtre Elde edilen özellikler

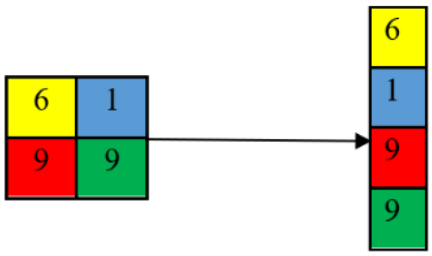
Şekil 3.10. (Devamı) Konvolüsyon işlemi.

Öznitelikler matrisi elde edildikten sonra bu matrise boyut azaltma için havuzlama işlemi uygulanmaktadır. Havuzlama için özellikler matrisi genel olarak 2x2 boyutundaki alt matrislere ayrılmakla birlikte 3x3 veya daha fazla boyuttaki matrislere de ayrılabilmektedir. Ayrılan her bir alt matrisin hücrelerindeki değerler ya ortalama (Average) ya da en büyük değeri alınarak (Max pooling) Şekil 3.11' de görüldüğü gibi boyut düşürme yapılmaktadır.



Şekil 3.11. En büyük değer alınarak yapılan havuzlama işlemi.

Öz nitelikler matrisine uygulanan boyut azaltma işleminden sonra 4x4 boyutundaki bir görüntü 2x2 boyutuna indirgenmektedir (Downsampling). Havuzlama işlemi sayesinde ağ tarafından işlenecek parametrelerin sayısı azalmaktadır. Havuzlama katmanı sonunda elde edilen matris tam bağlantılı katmanda düzleştirilerek kullanılabilmektedir. Düzleştirme (Flattening) işlemi için matrisin tüm satırları Şekil 3.12' de gösterildiği gibi sırayla tek bir vektör olarak dizilmektedir.



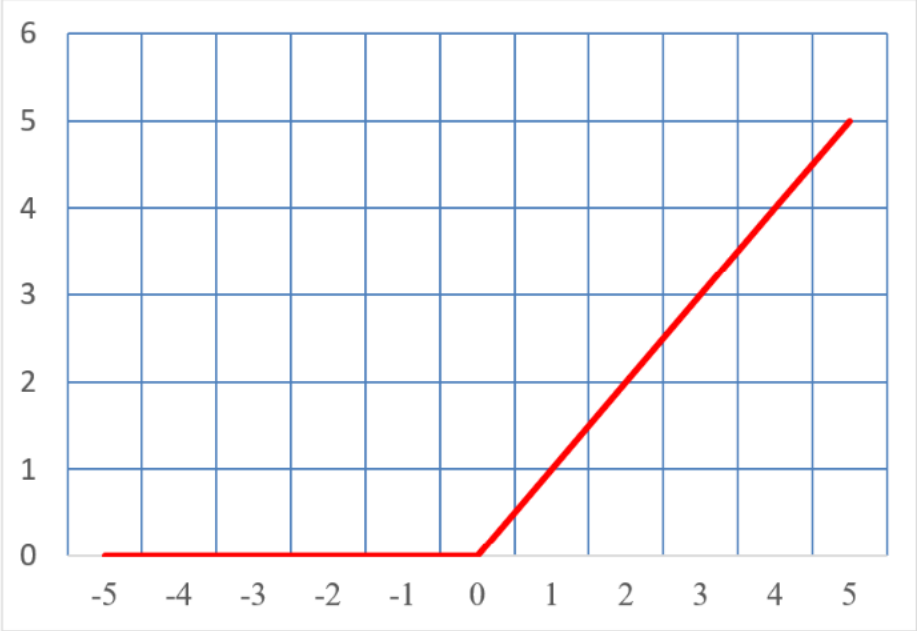
Şekil 3.12. Düzleştirme işlemi.

Düzleştirilmiş olan yeni vektör tam bağlantılı katmanın girdilerini oluşturmaktadır. Düzleştirmeden sonra sınıflandırma (çıktı) katmanı tarafından kayıp fonksiyonu kullanılarak bir sonuç ve gradyan hatası hesaplanmaktadır.

Derin öğrenmede katmanlar arasında aktivasyon fonksiyonları kullanılmaktadır. Bir modele doğrusal olmayan özellikleri eklemek için aktivasyon fonksiyonları kullanılmaktadır.

Doğrultulmuş Lineer Birim (Rectified linear unit-ReLU) doğrusal olmayan özellikleri modele eklemek için kullanılan ve denklem 3.9' da gösterilen bir aktivasyon fonksiyonudur. Girdi belirli bir değerin üzerinde olduğunda etkin hale gelmektedir. Girdi sıfırın altında olduğunda fonksiyonun değeri sıfır olmaktadır. ReLU fonksiyonunun örnek bir grafiği Şekil 3.13' de gösterilmektedir.

$$f(x) = max(0, x) \tag{3.9}$$



Şekil 3.13. ReLU aktivasyon fonksiyonu.

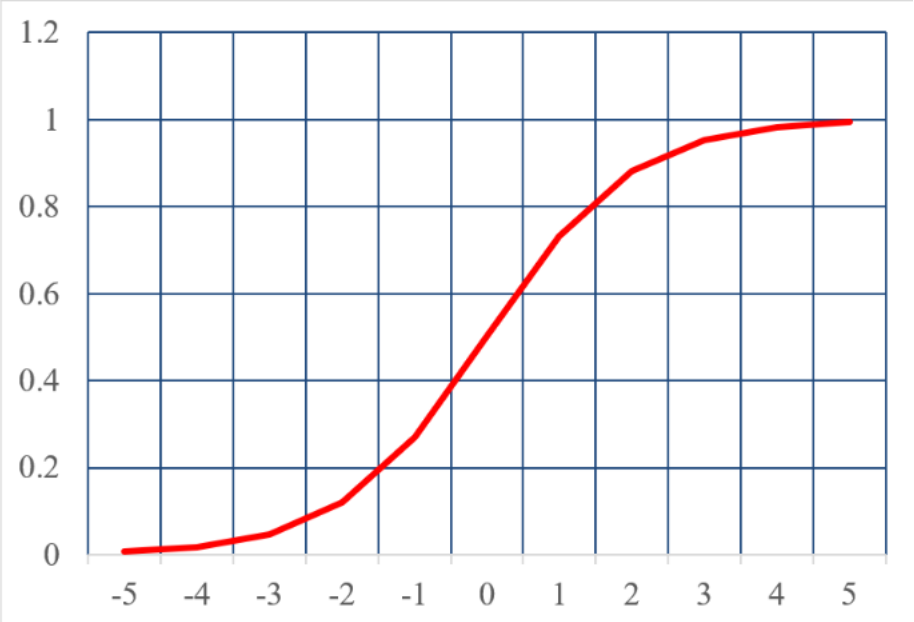
Şekil 3.13' de görüldüğü gibi fonksiyon negatif değerler için sıfır üretmektedir. Bu durumda geri yayılım esnasında parametre güncellemesi olmayacağından öğrenme de olmamaktadır. Bu durumda kullanılan fonksiyonlardan birisi Sızıntı ReLU (Leaky ReLU) fonksiyonudur.

Sigmoid fonksiyonu veride sonsuz aralıkta var olan bağımsız değişkenleri 0 ile 1 arasındaki olasılıklara dönüştüren, verideki aykırı değerleri kaldırmadan azaltabilen bir fonksiyondur. Her bir sınıf için fonksiyon bağımsız olasılıklar üreten fonksiyonun denklem 3.10 ile hesaplanmaktadır. Sigmoid fonksiyonunun grafiği Şekil 3.14' de verilmiştir.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.10}$$

Bu fonksiyon sıfır ve bire yaklaştıkça türevi çok küçük olacağından ve sıfıra yakınsayacağından öğrenme en az düzeyde gerçekleşmektedir. Türev sıfır olduğunda ise öğrenme olmamaktadır. Türevin çok küçük olmasıyla yerel minimumlar genel minimum gibi görülecek ve model daha fazla öğrenemeyecektir.

Softmax aktivasyon fonksiyonu sigmoid fonksiyonuna benzemektedir. Sigmoid fonksiyonundan farklı olarak ikiden fazla sınıflandırma gerektiğinde çıktı katmanında kullanılan bir aktivasyon fonksiyonudur.

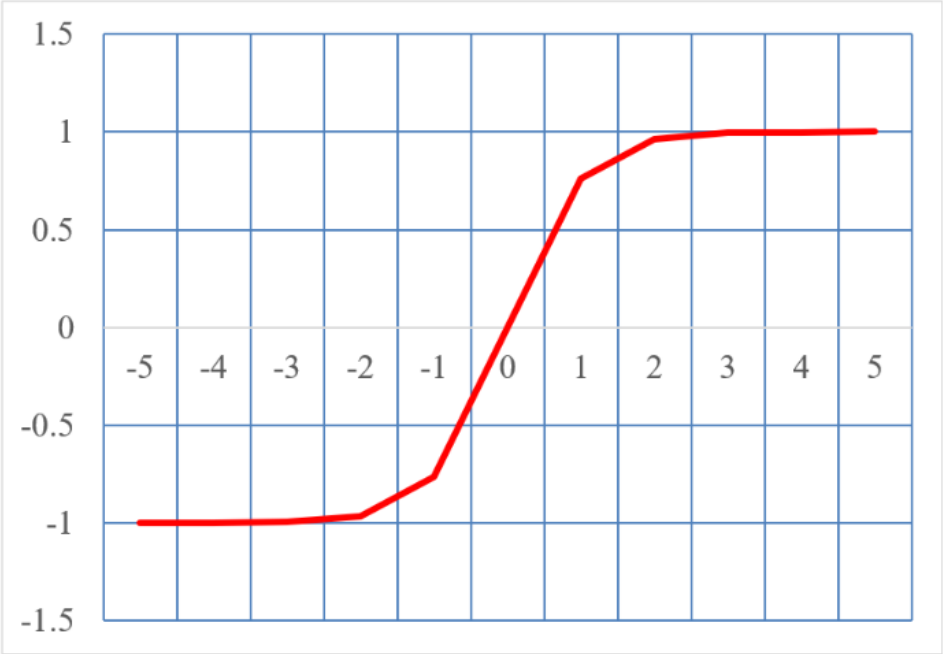


Şekil 3.14. Sigmoid fonksiyonu.

Hiperbolik bir trigonometrik fonksiyon olan tanh (Hiperbolik Tanjant) aktivasyon fonksiyonu, bir dik üçgendeki hiperbolik sinüsün hiperbolik kosinüse oranını göstermektedir. Sigmoid fonksiyonundan farklı olarak aralık değerleri -1 ve +1 dir. Özellikler negatif sayılarda kolaylıkla çalışabilmektedir. Fonksiyonun türevi dik olduğundan öğrenme daha verimlidir. Şekil 3.15' de gösterilen bu fonksiyon -1 ve +1 değerlerine yaklaşınca sıfıra yaklaşan bir türeve ve sınır değerlerinde sıfır türeve sahip olduğundan öğrenme sorunları ile karşı karşıya kalmaktadır.

Hiperbolik tanjant fonksiyonu denklem 3.11' de gösterilmektedir.

$$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})} \tag{3.11}$$

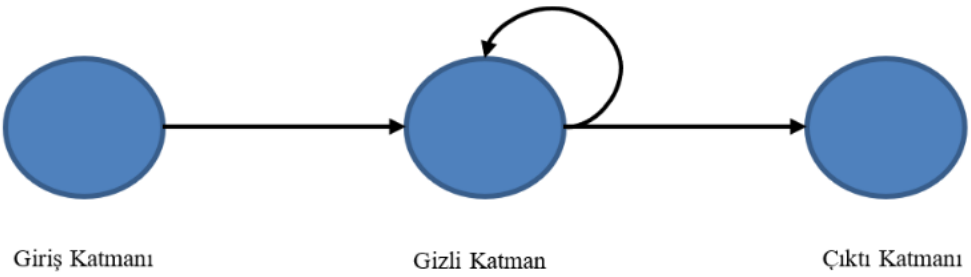


Şekil 3.15. Tanh (Hiperbolik tanjant) fonksiyonu.

3.2.3. Tekrarlayan sinir ağları

İleri beslemeli sinir ağları ailesinden olan Tekrarlayan sinir ağları (Recurrent neural networks-RNNs) her bir vektörü bir dizi girdi vektöründen alarak bu girdileri birer birer modellemektedir. RNNs iki veya daha çok katman arasında geri beslemeye sahip olabileceğinden zaman serisi verilerinden öğrenebilmektedirler. Girdi dizilerini işlemek için dahili bellek kullanmaktadırlar. RNNs zaman boyutunu modelleme özelliğine sahiptir. RNNs bağlantılarındaki döngüler modele ses, dil, zaman serisi gibi alanlarda zamansal davranış kazanma doğruluğunu sağlamaktadırlar.

RNN en çok kullanımına göre Temel RNN (Simple RNN-SRNN), Geçit Tekrarlayan Birim (Gated Recurrent Unit-GRU) ve LSTM olarak üç ağa ayrılmaktadır. SRNN yapısı ile yoğun bağlı ağ yapısı karşılaştırıldığında RNN blokta Şekil 3.16'daki gibi bir geri beslemenin olduğu görülmektedir.

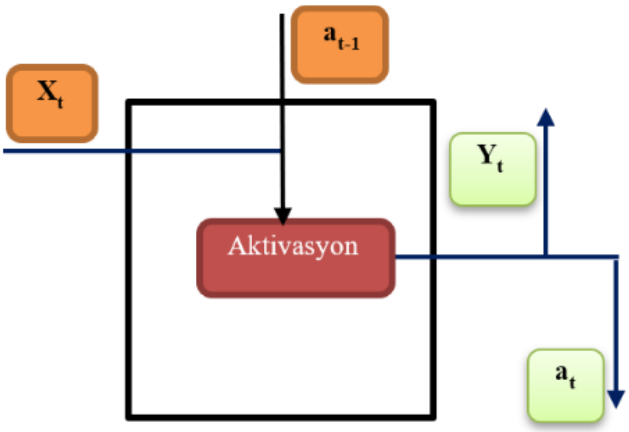


Şekil 3.16. Temel RNN yapısı.

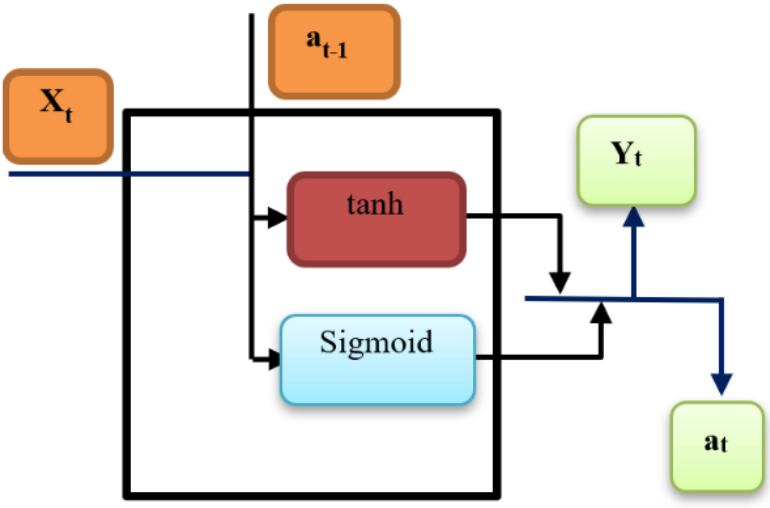
RNN mimarisinde mevcut olan hücrelerin içerisi Şekil 3.17' de [70] gösterildiği gibidir. Burada X t zamanındaki girdiyi, a değerleri ağırlıkları ve Y ise t zamanındaki hedef değişkeni göstermektedir.

SRNN mimarisinin uzun zaman öncesini hatırlayamaması gibi bazı sorunlar içermesinden dolayı Şekil 3.18' de [70] gösterilen GRU mimarisi geliştirilmiştir.

LSTM diğer türlerine göre daha çok parametrenin öğrenilmesini sağlayan en gelişmiş RNN mimarilerinden birisidir. LSTM mimarisinin Şekil 3.19' da [70, 71] görüldüğü gibi bir hücresinde üç adet sigmoid ve iki adet tanh fonksiyonu bulunmaktadır. Burada a ve c sembolleri zamana göre ağırlıkları, X sembolü girdiyi ve Y sembolü de çıktıyı göstermektedir.

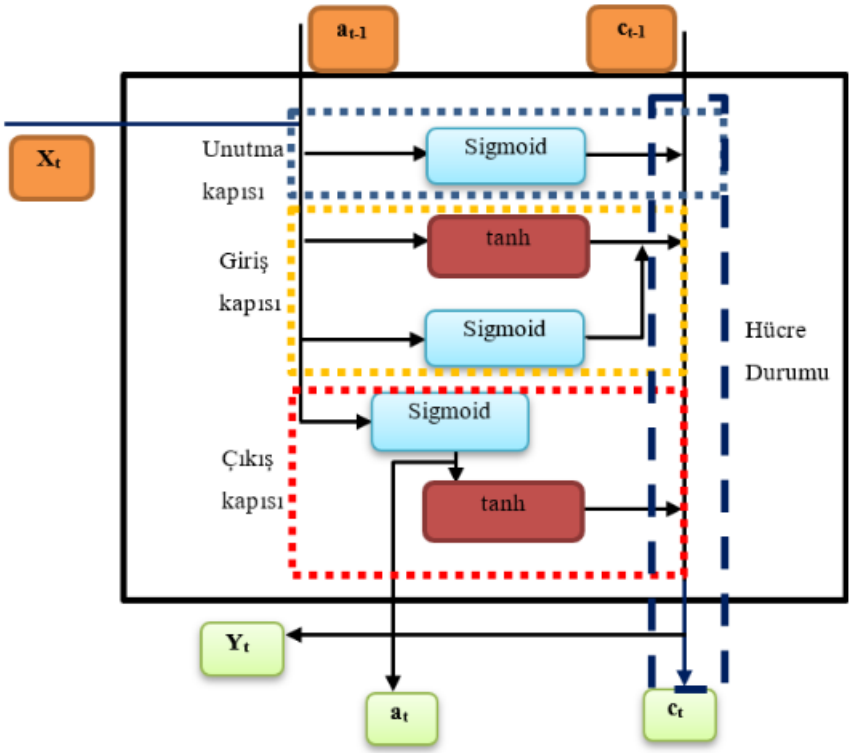


Şekil 3.17. RNN hücre yapısı



Şekil 3.18. GRU hücre yapısı

LSTM hücresinde anlamlı olan bilgileri diğer hücrelere ileten, aynı zamanda önceki hücrelerdeki bilgileri tutan ağ hafızası Hücre Durumu (Cell State) kapısı, önceki hücrelerden gelen bilgiler ile mevcut bilgilerin aktivasyon fonksiyonu sonucu unutulup iletileceğini belirleyen Unutma Kapısı (Forget Gate), önceki hücrelerden gelen bilgi ile mevcut bilginin aktivasyon sonucu güncellenip güncelleştirilmeyeceğini belirleyen Giriş Kapısı (Input Gate) ve sonraki hücreye gönderilecek bilgileri taşıyacak aynı zamanda tahmin yapmaya yarayacak Çıktı Kapısı (Output Gate) bulunmaktadır.



Şekil 3.19. LSTM hücre yapısı.

3.2.4. Transfer öğrenme

Derin öğrenme sistemleri, bir modeli eğitmek için büyük miktarda veri gerektirmektedir. Veri miktarı arttıkça modelin öğrenme başarısı da o kadar artmaktadır. Eğitim için çok az miktarda veri olduğunda, eğitilmiş bir model elde etmek büyük bir sorundur. Ayrıca, modeli eğitmek için önemli ölçüde zamana ihtiyaç vardır. Benzer bir alanda eğitilmiş bir modelin ağırlıkları, problemleri çözmek için transfer öğrenme yöntemi kullanılarak aktarılmaktadır. Böylece eğitim için daha az miktarda veri kullanılabilmekte ve bu yöntem kullanılarak model eğitimi daha az zaman almaktadır [72, 73].

Bu çalışmada, Python'da Keras kütüphanesini kullanarak tek grafik işleme birimine (GPU) göre transfer öğrenme yöntemini kullanarak CNN modeli tasarlanmıştır. CNN modelinin eğitiminde veri yöntemleri ile elde edilen 160 × 160 × 3 boyutlu işaretleyici görüntüler kullanılmıştır. Orijinal görüntüler veri büyütme yöntemleri kullanılarak çoğaltılmıştır. Veri setimizin az olması nedeniyle transfer öğrenme (Transfer Learning) yöntemi kullanılmıştır. Çalışmalar, Imagenet ile eğitilmiş Mobile Net [74] derin öğrenme CNN modeli kullanılarak gerçekleştirilmiştir. MobileNet küçük, düşük gecikmeli bir modeldir ve mobil cihazlarda verimli bir şekilde çalıştırılabilmektedir. MobileNet modelleri, gecikme ve doğruluk açısından popüler modellerin başarısını elde edebilmektedir.

3.3. Otonom Yönlendirmeli Araçlar

Otonom yönlendirmeli araçlar iç ve dış ortamlarda zemin üzerinde hareket ederek belirli görevleri yerine getiren genel olarak tekerlekli mobil robotlardır. AGV' ler endüstri ortamlarında genel olarak depolar, fabrikalar gibi birçok alanlarda malzeme taşımak için kullanılmaktadırlar. Bu araçlar önceleri tamamen değişmez ortamlarda kullanılırken günümüzde yapay zekâ alanındaki gelişmeler ile dinamik ortamlarda da yükleme-boşaltma alanlarında taşıma yapmaktadırlar. Sabit ortamlarda kullanılan araçlar genel olarak akıllı olmadıklarından araçların çalışma alanı insanların erişimine kapatılmaktadır. Bu sayede işçi sağlığı ve güvenliği sağlanmaktadır. Günümüzde AGV' ler yapay zekâ sayesinde dinamik ortamlarda sensörleri kullanarak insanları ve çevreyi algılayabilmekte, görevlerdeki değişiklikleri yeniden programlanmaya gerek kalmadan uyum sağlayabilmekte, dolu rota üzerinde ilerlerken kendilerine farklı boş rotalar bulabilmektedirler.

Son yıllarda otonom araçlar iç ortamda ve dış ortamda oldukça popüler bir araştırma konusu olmuştur ve özellikle otonom sürüş araştırmalarına olan ilgi artmıştır.

Otonom sürüş sisteminin mimarisi karmaşık teknolojilerden oluşan birçok alt sistemin birleşimidir [75]. Bu alt bileşenler aşağıda verilmiştir.

3.3.1. Ortam algılama

Ortam algılama (Sensing) bir veya daha fazla sensör yoluyla ortamın kullanılabilecek ham verilere dönüştürülmesidir. Sensör olarak GPS (Global Positioning System), LiDAR (Light Detection and Ranging), kamera veya radar sistemleri ile bu sistemlerin

bazıları beraber kullanılmaktadır [76, 77]. AGV' ler bu farklı sensör tiplerini kullanarak içinde bulunduğu ortamı algılamakta ve ortamın haritasını oluşturarak bu ortamda hareket edebilmektedirler. Ayrıca bu otonom araçlar, ortam harita bilgisini kullanarak engelleri algılayabilmekte ve çarpışmadan kaçınarak ortamda dolaşabilmektedirler [78].

GPS, uyduları kullanarak yer tespiti için kullanılan bir sistemdir. GPS iç ortamlarda çalışmadığı için kullanılmamakla birlikte açık havadaki ortamlarda kullanılabilecek uygun bir sensördür. Araçlar bu sinyalleri kullanarak ortamda kendini konumlandırabilmektedir. GPS sinyalleri uydudan sağlandığından veri elde etmede gecikme olmaktadır. Ek olarak hassas yer belirlemede sıkıntılar ortaya çıkabilmektedir. GPS' in kullanılamayacağı ortamlarda IMU (Inertial Measurement Unit) cihazları kullanılabilir. Araçların konumunu iç ortamlarda izlemek için kullanılan IMU cihazı temel olarak ivmeölçer ve jiroskop olarak iki sensör içermektedir. Üç eksen için üç farklı değer üreten ivmeölçer yer çekiminden etkilenmektedir. Açısal dengenin korunmasını sağlayan jiroskop yönün belirlenmesinde kullanılmaktadır. Her iki sensörün tek tek kullanıldıklarında elde edilen verilerden daha doğru ve net veriler iki sensörün verileri birleştirilerek elde edilmektedir ve bu veriler IMU cihazından alınmaktadır [79, 80].

LiDAR haritalama, konumlanma ve engellerden kaçınmak için kullanılan yüksek doğrulukta bir sensör türüdür. LiDAR sensörleri genellikle çarpışma uyarısı gibi bildirimlerde bulunan kısa mesafeli ve uzun mesafeli olarak sınıflandırılabilmektedirler. Kısa mesafeli olan sensörlerin ölçüm mesafesi ışık yoğunluğunun dağılmasından dolayı düşüktür. Ayrıca bu sensörler fiyat olarak da ucuzdurlar. Uzun mesafeli sensörler uzak mesafedeki bir nesnenin belirli bir bölümünü aydınlatacak kadar güçlü ışık göndermektedirler. Nesneyi piksel bazında aydınlatarak nesnenin üç boyutlu görüntüsünü çıkarabilmektedirler. Böylece nesnelerin ayrıntılı tespitini hassas bir şekilde yapmaktadırlar. LiDAR çevresine ışık gönderip gelen yansımanın süresini ölçerek mesafe hesaplamasını da yapmaktadır. LiDAR' daki bir lazer modülü tarafından üretilen ışık hedefe çarparak geri gelmekte ve alıcı bu gelen ışığı algılamaktadır. İletilen ışıktan alınan ışığa kadar geçen süreden mesafe hesaplanmaktadır. Ortamdaki zeminin segmentasyonu da LiDAR kullanılarak oldukça hassas bir şekilde yapılabilmektedir [81-83].

Derin öğrenmenin ortaya çıkmasıyla birlikte otonom robotlarda kameralar sıklıkla kullanılmaya başlanmıştır. Kameralar nesne tanıma ve takibinde hem iç ortam hem de dış ortamlarda oldukça başarılı sonuçlar vermektedirler. Bir araca monte edilen kameralar farklı yönleri, nesneleri, işaretleri tespit etmede kullanılmaktadırlar. Özellikle derin öğrenmenin yaygınlaşmasıyla kameraların kullanım oranı artmıştır.

Bir otonom araçta yukarıda bahsedilen sensörlerin yanında Radar (Radio Detection and Ranging) sistemleri de kullanılmaktadır. Radar verileri kullanılarak aracın nesnelerden, engellerden ne kadar uzaklıkta olduğu hesaplanmaktadır. Ek olarak radar ile aracın hızı da elde edilmektedir. Radar verileri radar sistemi tarafından üretilmekte ve genel olarak ek veri işlemeye gerek duymayabilmektedirler. Bu kategoriye giren sonar veya ultrasonik sensörler kısa mesafeli ölçüm sensörleridirler. Bu sensörler de mesafe ve hız hesaplamada kullanıldıkları gibi kısa mesafedeki çarpışmaları da engellemek için kullanılmaktadırlar [84].

3.3.2. Ortamı anlama

Ortamı anlama (Perception), aracın üzerinde monte edilmiş olan veya üzerinde olmayıp da bağlantılı olduğu diğer sensör verilerini kullanarak içinde bulunduğu ortamı anlamasını sağlamaktadır. Ortamı anlamada sensör verileri anlam kazanmaktadır. Otonom sürüşte ortam anlama sayesinde aracın çevresindeki dinamik ortam algılanarak sensör verilerine dayalı güvenli ve ayrıntılı bir ortam ortaya konmaktadır. Böylece araç ortamındaki bir engeli, yol yüzeyini, yer işaretlerini, hareketli nesneleri algılayarak bunlardan kaçınma, izleme gibi görevleri daha akıllı olarak yerine getirebilmektedir [85, 86]. Ortamı anlama LiDAR, GPS, IMU, kamera gibi sensörlerden gelen verilerin işlenmesiyle gerçekleşmektedir.

Ortamı anlama sistemi birçok alt sistemden oluşmaktadır. Bunlardan birisi lokalizasyondur (localization). Lokalizasyon sensörlerden gelen verileri kullanarak bir otonom aracın bulunduğu ortam içerisindeki konumunu hassas bir şekilde belirlemesidir ki bu işlem nesne tanıma ve nesne takibi seviyesindeki algoritmalarla yapılmaktadır. Lokalizasyon kamera, GPS, IMU gibi sensör verileriyle ve bu verileri işleyen algoritmalarla belirlenmektedir.

Ortamı anlamanın bir diğer alt sistemi nesne algılamadır. Bir otonom araç için çevresindeki nesneleri algılaması aracın otonom hareketi için büyük önem taşımaktadır. Özellikle bir engelle karşılaşan aracın bu engeli algılayarak durması veya

algoritmasına göre engeli geçmesi aracın ilerlemesi için önemlidir. Otonom hareket eden aracın yolu üzerindeki ve çevredeki nesneleri tanıması ve uzaklıklarını tespit etmesi için LiDAR hassas veriler sağlarken bu çalışmada kullanıldığı üzere son zamanlarda derin öğrenmenin gelişmesiyle CNN kullanılarak da nesne tanıma başarılı bir şekilde yapılmaktadır.

Nesne takibi (object tracking) aracın hareket ederken çevresindeki diğer araç, insan ve nesnelerle çarpışmasını önlemek için bunları izlemede kullanılan bir teknolojidir. Derin öğrenmenin gelişmesiyle doğal görüntüler kullanılarak model eğitimi yapılmakta ve eğitilen model kullanarak araç, insan, nesne ve zemindeki işaretçiler aracın hareketi esnasında tanınmaktadır. Böylece araç nesnelerin takibini yapabilmektedir [87]. Nesne takibi ayrıca aracın hareket eden nesnelere çarpmasını engellemek için kullanılmaktadır.

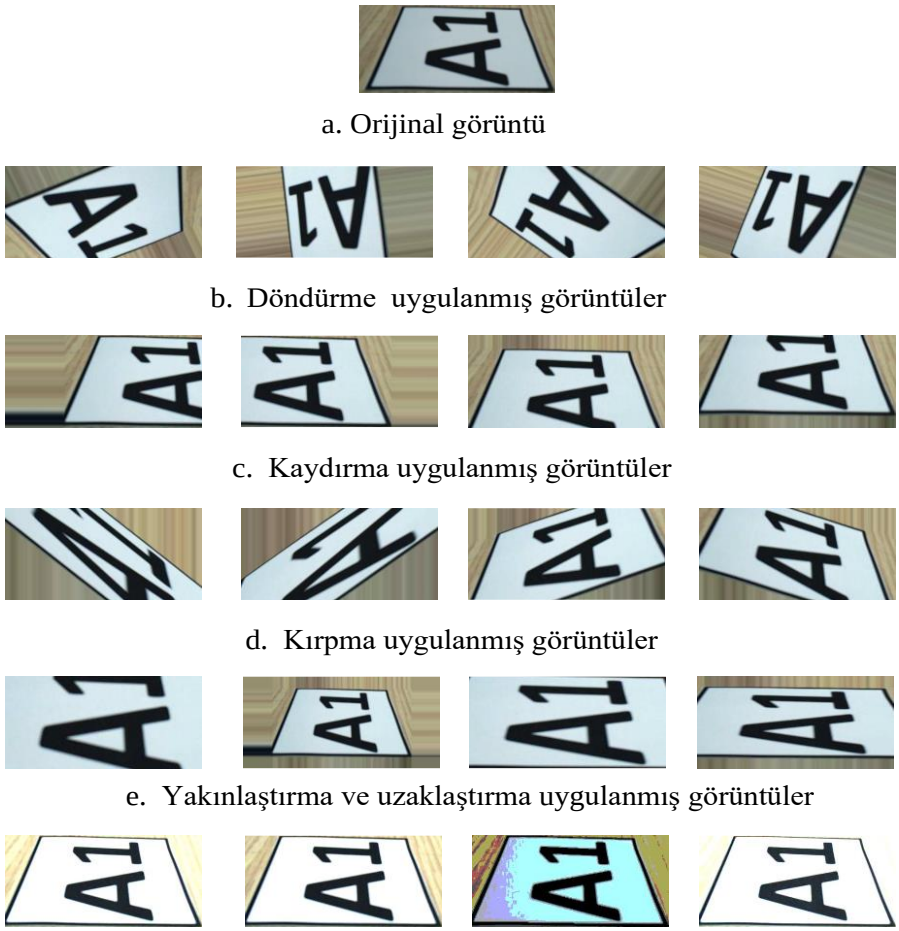
3.3.3. Karar verme

Otonom sürüş mimarisinde ortamı algılama ve anlamadan sonra karar verme (decision making) önemli bir yer tutmaktadır. Karar verme aşamasında gerçek zamanlı olarak güvenli bir eylem planı oluşturulur ve yol planlama (path planning) ile ortamdaki tüm olası yollar aranır ve hedefe giden uygun yol veya yollar en düşük zaman, mesafe maliyetiyle belirlenmektedir. Hareket halindeki aracın belirlenen yolda hedefine giderken sürekli olarak yol planlama yapması uygun algoritmik bir yaklaşımla uygun bir zaman maliyetiyle sağlanabilmektedir. Yol planlama sonucunda hareket eden aracın engellere çarpmadan hedefine güvenli olarak gitmesi önemli sorunlardan birisidir [88, 89]. Engellere çarpmanın engellenmesi için kameradan alınan görüntüler yanında diğer sensörlerden gelen görüntüler de kullanılmaktadır. Bunların ötesinde engel tahmin sistemleri de geliştirilerek engelleri daha görmeden aracın bu tahmin engele göre hareketi kontrol edilebilmektedir.

3.4. Veri Setinin Hazırlanması

Eğitim veri setini hazırlamak için araç model kamerasından her bir işaretçi için sekiz adet görüntü alınmıştır. İşaretçiler araç kamerasından farklı mesafelere yerleştirilmiş ve dört farklı boyutta görüntü alınmıştır. 17 farklı işaretçi görüntüsünden toplam 544 görüntü elde edilmiştir. Bir CNN modelini eğitmek için birçok örnek görüntüye ihtiyaç vardır [90]. Bu nedenle 544 orijinal görüntüden veri büyütme kullanılarak yeni görüntüler elde edilmiştir. Veri büyütme için döndürme, kaydırma, kesme,

yakınlaştırma ve parlaklık yöntemleri kullanılmıştır. Bu veri büyütme yöntemleri bir görüntüye sırayla uygulanmıştır. Oluşturulan veri setinden birbirine benzer görseller silinmiştir. Bu işlemler sonucunda veri seti için 18.085 görüntü elde edilmiştir. Örnek bir görüntüye uygulanan veri büyütmenin sonucu Şekil 3.20' de gösterilmektedir.

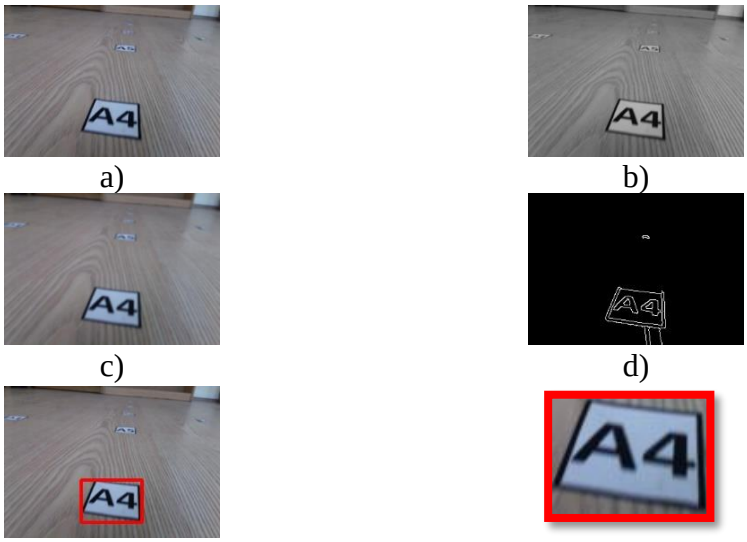


Şekil 3.20. CNN modelinin eğitiminde kullanılacak işaretçilerin üretiminde kullanılan veri artırma yöntemleri.

3.5. Görüntüdeki İlgi Alanının Elde Edilmesi

Kameradan alınan görüntüler bilgisayara gönderilmektedir. Renkli görüntüler, ilgilenilen bölgeyi (ROI) bulmak için gri seviyeli görüntülere dönüştürülmektedir. Gri seviye görüntülerde bölgeleri segmente etmen için farklı eşikleme algoritmaları bulunmaktadır. Kümeleme tabanlı, entropi tabanlı, şekil tabanlı, öznitelikteki benzerlik tabanlı farklı algoritmalar bulunmaktadır. Kümeleme tabanlı en çok kullanılan eşikleme yöntemi Otsu' dur. Görüntüdeki bölge veya nesne kenarlarını bulmak için literatürde farklı kenar bulma algoritmaları geliştirilmiştir. Bu çalışmada

kullanılan görüntüler Gauss bulanıklığı, Otsu ve Canny kenar algılama algoritmaları kullanılarak işlenmektedir. Otsu yöntemi literatürde en yaygın kullanılan eşikleme algoritmalarından biridir. Bu yöntemle görüntüdeki tüm piksellerin değerleri alınmakta ve dağılım kriterlerine göre varyansları hesaplanmaktadır. Pikseller bu varyans değerlerine göre kümelenmektedir [91, 92]. Görüntüye bir yumuşatma filtresi uygulanmakta ve filtrelenen görüntü Canny algoritmasının ilk adımında türetilmektedir. Bu işlem sonucunda yönlü gradyan vektörlerine bağlı olarak tek piksel kenar çizgileri elde edilmektedir [93]. Gri görüntü dönüşümü, Gauss filtresi bulanıklaştırma, Otsu bölütleme ve Canny kenar algılama, kameradan alınan görüntülere sırayla uygulanmaktadır. İşlenen görüntüdeki ROI alanları hesaplanmakta ve en geniş alana sahip işaretleyici görüntü kırpılmaktadır. ROI bölgesini elde etme adımları Şekil 3.21' de gösterilmektedir. Kırpılan işaretçi görüntüsü, işaretleyici bilgisini sınıflandırmak için CNN modeline uygulanmaktadır.



a) b) c) d) e) f)

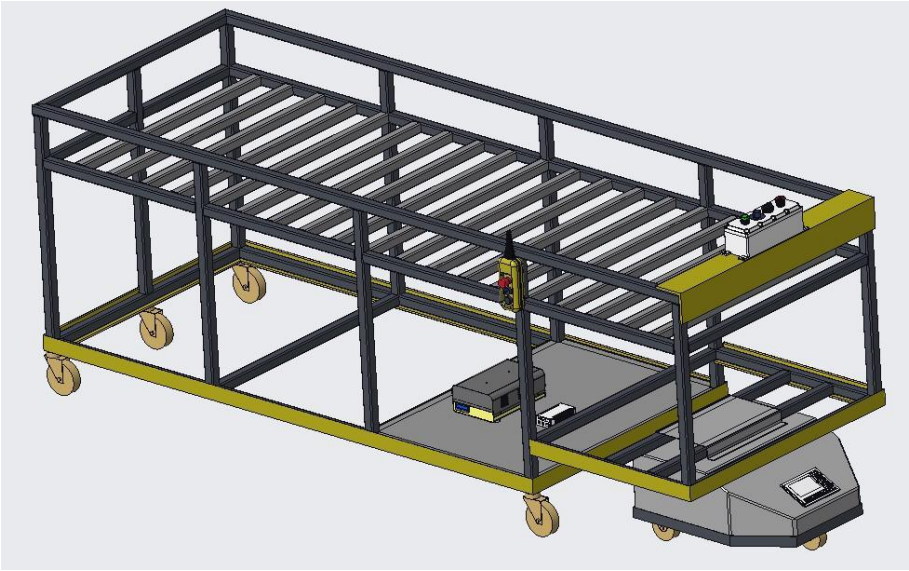
Şekil 3.21. Görüntü işleme adımları. a) Alınan görüntü, b) Grileşme, c) Bulanıklık, d) Otsu ve Canny, e) İşaret tespiti, f) Kırpılmış işaretçi görüntüsü.

3.6. Rota Takibi için Derin Öğrenme Tabanlı Zemin Yolu Modeli

Bu çalışmada geliştirilen sistem için tasarlanan zemin yolu modeli, ayrı hücresel işaretçilerden oluşmaktadır. Diğer model bileşeni ise önerilen zemin yolu modelinde rotaları takip eden kameralı araçtır. İşaretleyiciler, bir harf ve bir sayıdan oluşan temel bir yapıdır. Modele eklenecek sayılar artan ve azalan harf-sayı çiftlerinden elde edilir. Önce bir harf eklenmekte ve harften sonra dokuza kadar sayılar eklenmektedir. A ve 1 yan yana eklendikten sonra işaretçiye yazılacak A1 değeri elde edilmektedir. Bu

değer ayrıca işaretçinin konum adresini de göstermektedir. Bu adres kullanılarak aracın konumu belirlenmektedir. Sonraki işaretçi adresi için A harfine 2 sayısı eklenmektedir. Bu işlem A9 işaretçi adresi elde edilene kadar devam etmektedir. Bu nedenle, bir harften dokuz işaret elde edilmektedir. Zeminde aynı yönde devam ettiği sürece harf değişikliği olmamaktadır. İşaretçileri oluşturmak için artık tek karakterli harf kalmadığında iki harf ve bir sayıdan oluşan bir dizi kullanılmaktadır. Bu işaretleyiciler kullanılarak zemin yolu modeli elde edilmektedir.

Zemin yolu modelinde rotalar oluşturulurken başlangıç konumundan hedef konuma giden artan işaretçiler seçilmektedir. Artan ve azalan harf-sayı çiftleri dahil olmak üzere işaretçilerden oluşturulan rotalar da artan veya azalan rotaları oluşturmaktadır. Araçlar rotalar üzerinde tek yönlü hareket etmektedir. Yani rotalar ya artan sırada ya da azalan sırada oluşturulmaktadır. Artan veya azalan sıradaki işaretleyicileri kullanmanın bir avantajı, otonom aracın merkezi sistemden bağlantısı kesildiğinde, artan veya azalan yönündeki işaretçileri takip ederek başlangıç noktasına ulaşabilmesidir. Bu yöntemle oluşturulan örnek bir rota, A1, A2, A3, A4, A5, D1 ve D2 işaretleyicilerini içeren bir vektördür. Bu rota vektöründe A5 adresinden sonra aracın yönü değişeceği için D1 marker adresi kullanılmaktadır. Rotalar, artan veya



Şekil 3.22. Otonom araç modeli.

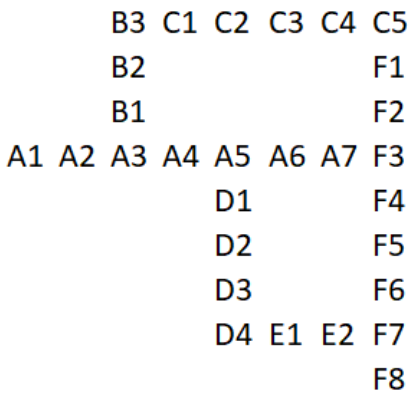
azalan işaret vektörlerinden oluşmakta ve araç modelinin bu rotaları takip etmesi sağlanmaktadır. İşaretçinin algılanmasını kolaylaştırmak için işaretçilerin etrafına kalın çizgiler çizilmektedir. Kavşaklar 90 derecelik bir açıyla oluşturulmaktadır.

İşaretçiler arasında yirmi beş santimetre uzaklık vardır. Bu mesafenin belirlenmesinde model aracın uzunluğu önemlidir.

Önerilen modele uygun özel otonom araçlar geliştirilmiştir. Geliştirilen AVM üzerine Raspberry Pi 3 kartı, Raspberry Pi motor sürücü kartı, 640×480 piksel renkli Raspberry Pi kamera ve 7.4 V 1750 mAh Li-po pil takılmıştır. Ayrıca, AVM hareketi ve yön değişikliği, AVM tekerleklerine bağlı dört doğru akım (DC) motoru kullanılarak kontrol edilmiştir. Araç iki farklı hızda sürülebilmektedir. Araç üzerindeki kamera tek görüş açısına sahiptir ve sağa sola hareket etmemektedir. Bu çalışmada tasarlanan AVM Şekil 3.22' de gösterilmiştir.

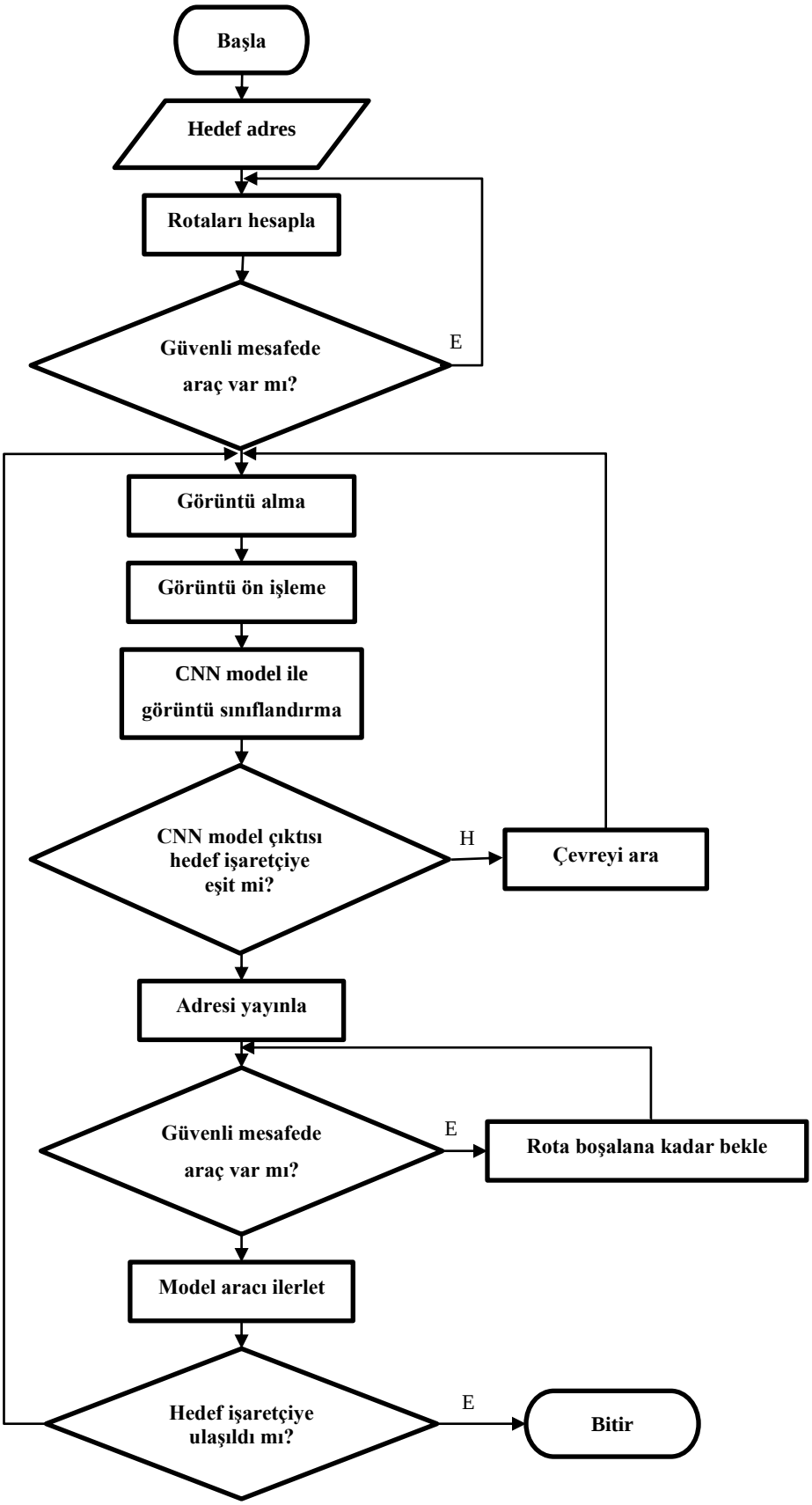
İlk olarak önerilen sistemde zemin yolu modeli bir sunucu tarafından otonom araçlara vektör olarak gönderilmektedir. Aracın hareket etmesi için Şekil 3.24' de gösterildiği gibi hedef işaretçi adresi gerekmektedir.

Örnek zemin yolu modeli temsili olarak Şekil 3.23’ de gösterildiği gibi olsun. Hedef olarak araca F8 işaretçi adresi verilsin. Bu zemin yolu modelinin sunucudan her bir araca gönderildiği vektör yapısı şu şekilde olacaktır: A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C1, C2, C3, C4, C5, D1, D2, D3, D4, E1, E2, F1, F2, F3, F4, F5, F6, F7, F8, A3-B1, B3-C1, C5-F1, A5-D1, D4-E1, E2-F7, A7-F3.



Şekil 3.23. Örnek zemin yolu modeli.

Aracın rotaları oluşturması aşağıdaki gibi olacaktır:
Zemin yol modelini içeren vektörde işlem yapılacağı için bu vektörün bir kopyası oluşturulmaktadır. Kopya vektörden ilk değer olan A1 değeri alınmakta ve yeni boş bir Rota vektörüne eklenmektedir. A1 değerinin bir düğüm olup olmadığına bakılmaktadır.



Şekil 3.24. Geliştirilen rota takibi akış şeması.

Burada A1 değeri bir düğüme sahip değildir. A3 değeri ise hem B1 değerine hem de A4 değerine dallandığından bir düğümdür. Bu şekildeki bir düğüm zemin vektöründe A3-B1 şeklinde gösterilmektedir. Bunun gibi belirtilen bir düğüm değeri alt değerler içermektedir. A1 değerinden sonra gelen A2 değeri de bir düğüm olmadığından Rota vektörüne eklenmektedir. A3 değerinin bir düğüm olduğu A3-B1' den elde edilmekte ve Rota vektörüne A3 ve B1 değerleri ile B2 değeri eklenmektedir. B3 değerinin de bir düğüm olduğu B3-C1 birleşim değerinden bulunmakta ve Rota vektörüne hem B3 hem de C1 eklenmektedir. C2 değerinden C4 değerine kadar düğüm bulunmadığından bu değerler Rota vektörüne eklenmektedir. C5-F1 bir düğümdür ve Rota vektörüne eklenmektedir. F2 değerinden F8 değerine kadarki değerler Rota vektörüne eklenmektedir. Burada her bir değer alındıktan sonra hedef değer olan F8 değeri ile karşılaştırılmaktadır. Eğer hedef değer ile alınan değer birbirine eşitse Rota vektörü Rotalar vektörünün ilk değeri olarak atanmaktadır. Rota vektörü bir adet rota içeren bir vektörken Rotalar vektörü ise birden fazla rota içeren ve her bir vektörü değişken olan vektörler kümesidir. Rota vektörü Rotalar vektörüne eklendikten sonra A3-B1, B3-C1, C5-F1 düğümleri kopya zemin vektöründen silinmekte, Rota vektörünün içeriği temizlenmekte ve işlem yeniden baştan başlamaktadır. İkinci vektör oluşturulurken yeniden A1 değerinden başlayarak A4 değerine kadar ilerlenmekte ve bu değerler Rota vektörüne eklenmektedir. A5 değeri bir düğüm olduğundan A5, D1, D2, D3 değerleri ve D4-E1 düğümündeki D4, E1, E2 Rota vektörüne dahil edilmektedir. E2 değerinin ise zemin vektöründe E2-F7 belirtilen bir düğüm olduğu görülmektedir. F7 değeri ve artan olarak gelen F8 değeri de Rota vektörüne eklenmektedir. Yine burada F8 değeri hedef vektör değeri olan F8 değerine eşit olduğundan Rota vektörü Rotalar vektörünün ikinci vektörü olarak atanmakta ve kullanılan A5-D1, D4-E1 ve E2-F7 düğümleri kopya vektörden silinmektedir. Yeni kopya vektörü A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C1, C2, C3, C4, C5, D1, D2, D3, D4, E1, E2, F1, F2, F3, F4, F5, F6, F7, F8, A7-F3 değerlerini içermektedir. Üçüncü rota için yukarıdaki işlemler yeniden uygulandığında Rota vektörüne A1, A2, A3, A4, A5, A6 eklenecektir. A7 ise bir düğüm olduğundan A7, F3, F4, F5, F6, F7, F8 değerleri de Rota vektörünün içeriğinde yer alacaklardır. Rota vektörü Rotalar vektörünün üçüncü vektörü olarak atanacak ve son düğüm kopya vektörden silinecektir. Başka bir düğüm kalmadığından rotaları bulma işlemi sonlandırılacaktır. Bu işlemler sonunda Rotalar vektörünün içeriği aşağıdaki gibi olacaktır:

R1 = Rotalar [0]: A1, A2, A3, B1, B2, B3, C1, C2, C3, C4, C5, F1, F2, F3, F4, F5, F6, F7, F8

R2 = Rotalar [1]: A1, A2, A3, A4, A5, D1, D2, D3, D4, E1, E2, F7, F8

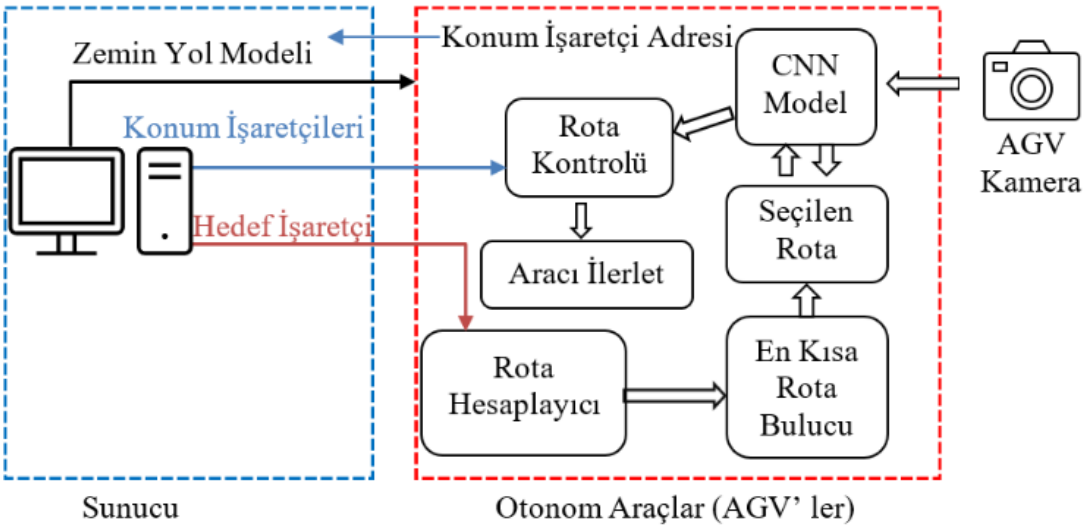
R3 = Rotalar [2]: A1, A2, A3, A4, A5, A6, A7, F3, F4, F5, F6, F7, F8

Hesaplanan bu rotalar arasından öncelikle en kısa olan R2 ya da R3 rotaları bulunmaktadır. R2 ve R3 rotalarında başka bir aracın varlığı diğer araçların sunucuya gönderildiği bilgiler kullanılarak kontrol edilmektedir. Her ikisinde de başka bir araç yoksa ve her ikisi de aynı uzunlukta olduğundan düşük indise sahip olan R2 rotası aktif rota olarak seçilmektedir. R2 rotası dolu ama R3 rotası boş ise R3 rotası aktif rota olarak belirlenmektedir. Her ikisinde de başka bir araç varsa R1 rotasında diğer bir aracın var olup olmadığı kontrol edilmektedir. R1 rotası boş ise aktif rota olarak kabul edilmektedir. Bulunan rotalardan hiçbirisi boş olmadığında araç beklemektedir. Aktif rotada araçtan en az dört işaret mesafesi uzakta başka araç yoksa model araç etkinleştirilmektedir. Yapılan çalışmalar sonucunda belirlenen ve kullanılan dört işaretçi mesafesi, araçların çarpışmasını önleyen güvenli mesafe olarak belirlenmiştir. Her araç, rotasında derin öğrenme modeli tarafından doğru olarak sınıflandırılan son adres bilgilerini yayınlamaktadır. Seçilen rota üzerinde başka bir araç varsa hedefe daha yakın olan diğer rota için de aynı kontroller yapılmaktadır. Görüntü alma adımında AVM kamera ile elde edilen video görüntüsünden saniyede bir kare alınmaktadır. Bu görüntü birden fazla işaretçi içerebilmektedir. AVM kameradan alınan renkli görüntü, görüntü işleme adımında gri seviyeli görüntüye dönüştürülmektedir. Gri seviye görüntüye Gauss bulanıklığı, Canny kenar algılama ve Otsu algoritmaları uygulanarak görüntüdeki uzak mesafedeki işaretçiler bulanıklaştırılmakta ve yakın olanlar keskinleştirilmektedir. Bu işaretçilerden en büyük alana sahip olan işaretçi, görüntüdeki işaretçilerin alan bilgileri karşılaştırılarak belirlenmektedir. Bu işaretleyici renkli görüntüden kırpılmaktadır. Kırpılan görüntü, bir sonraki algoritma adımında önceden eğitilmiş CNN modeline girdi olarak verilmektedir. CNN modeli tarafından sınıflandırılan işaretçi görüntüsü, hedef vektördeki karşılık gelen işaretçi bilgileri ile karşılaştırılmaktadır. İşaretçi bilgisinin ağın çıkışından alındığında ve bu bilginin hedef vektör işaretleyici bilgisine eşit olduğu durumda rota üzerinde hareket eden aracın yolladığı adres bilgisi kullanılarak rota üzerinde başka bir araç olup olmadığı bilgisi belirlenmektedir. Rotada başka bir araç

varsa, rotayı takip edecek yeni araç çarpışmayı önlemek için beklemektedir. Rota üzerinde başka araç yoksa araç rota boyunca hareket etmektedir.

İki aracın rotasında aynı işaretçi varsa, araçların hareketleri aşağıdaki gibidir: Ortak işaretleyiciye hangi araç daha yakınsa, o araç rotada ilerlemeye devam etmektedir; ortak işaretçinin adres bilgisi alınmadığında diğer araç hareketine devam etmektedir. Bu sayede araçların çarpışması önlenmektedir. Model aracın hareket etmesi için hedef vektördeki işaretçiler artan sırada olmalıdır. Bu iki işaretçi farklı ise araç modeli sağa dönerek zemin yolundaki işaretçinin görüntülerini görmekte ve sistemin çalışması görüntü alma aşamasına geçmektedir. Araç modelinin sağa dönüşü diğer işaretçilerin tespiti için önemlidir. Bu işlemler, hedef vektördeki son işaretçiye ulaşılana kadar devam etmektedir. Bir hata durumunda araç başlangıç adresine dönmektedir.

Yukarıda da anlatıldığı gibi artan düzende adresleme yöntemi, hedef işaretçiye giden yolları hesaplamak için kullanılmaktadır. Araçlar, zemin yolu modelindeki vektörlerden artımlı adreslemeyi ve düğümleri kullanarak rotaları hesaplamakta ve bu rotaları vektör olarak tutmaktadırlar.



Şekil 3.25. Sistemin blok şeması.

Tasarlanan sistemin çalışma prensibi Şekil 3.25' de gösterildiği gibi kısaca şu şekilde özetlenebilir. Bir alandaki tüm otonom araçlar merkezi bir sunucu ile iletişim halinde hareket etmektedirler. Sunucu önce Zemin Yolu Modelini araçlara vektör olarak göndermektedir. Ardından kullanıcı veya sunucu hareket edecek araca hedef işaretçi adresini vermektedir. Araç, bu adrese giden tüm olası yolları hesaplamakta ve en kısa olanı seçmektedir. Bu seçilen rotada araç hedefe ulaşmak için ilerlemektedir. Araç rota

üzerinde hareket ederken son okuduğu işaretçiyi sunucuya göndermektedir. Diğer otonom araçlar bu şekilde sağlanan iletişim ile o güzergahta bir araç olduğunu bilmektedirler. Şekil 3.25’ in sağ tarafında kırmızı ile gösterilen bölgedeki gerçekleştirilen veri işleme uç hesaplama (Edge Computing) ile yapılmaktadır. Sol taraftaki mavi renkle gösterilen bölge sunucuda tutulan verilerin işlenmesini göstermektedir.

4. DENEYSEL SONUÇLAR

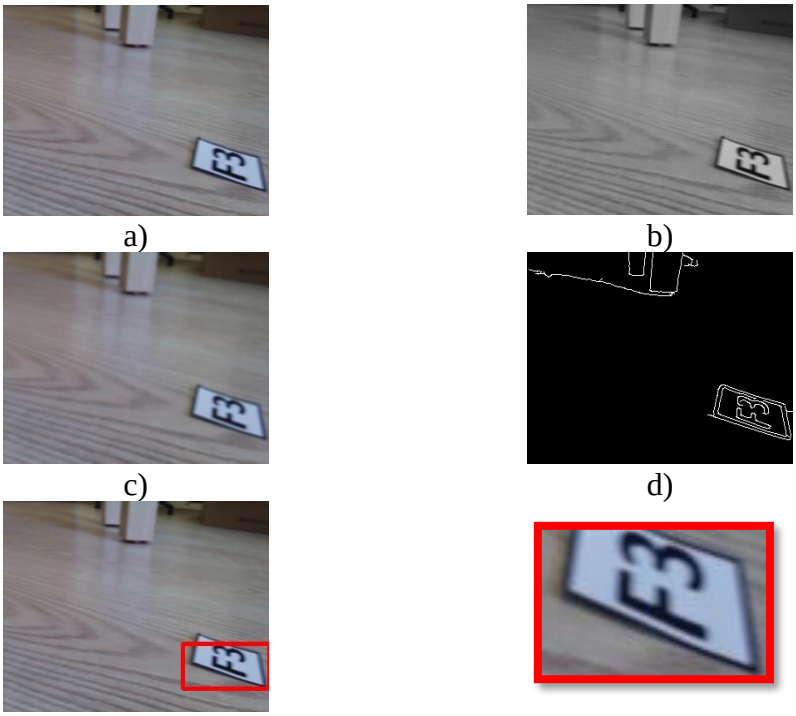
Derin öğrenmeye dayalı konumlandırma sisteminin performansı deneylerle test edilmiştir. Bu çalışmada farklı rotalar oluşturulmuş ve kullanılmıştır. Örneğin, ilk rota A1, A2, A3, A4, A5, D1, D2; ikinci rota, A1, A2, A3, A4, A5, A6, F1, F2, F3, G1, G2 ve üçüncü rota A1 A2, A3, B1, B2, E1, E2, F2, F3, G1, G2 işaretçilerinden oluşmaktadır. Her rota için model aracın iki farklı hız seviyesi için çalışmalar yapılmıştır. Araç modelinin rotalarının tamamlanma süreleri ve hız seviyeleri Tablo 4.1.'de verilmiştir.

Model aracın takip ettiği rotanın tamamlanma süresi, işaretçiler arasındaki mesafeye göre değişmektedir. İşaretçiler arasındaki doğrusal mesafe 25 cm olarak alınmıştır. Araç kamerasından alınan görüntüler Tablo 4.1.'de 0,22 m/s ve 0,27 m/s olarak verilen hız değerleri kullanılarak net bir şekilde görülebilmektedir. Ancak daha yüksek hızlar kullanıldığında araç kamerasından alınan görüntüler bozulmaktadır.

A1, A2, A3, A4, A5, A6, B1, B2, D1, D2, E1, E2, F1, F2, F3, G1, G2 işaretçilerinin görüntüleri, veri artırma yöntemleri kullanılarak arttırılmıştır. Bu verilerin %71'i eğitim, %25'i doğrulama ve %4'ü test için ayrılmıştır. CNN modeli, transfer öğrenme kullanılarak elde edilmiş ve orijinal ve artırılmış görüntülerle birlikte eğitilmiştir. Tam bağlantılı katmanla birleştirilen görüntüler, soft-max işleviyle 17 farklı sınıfa sınıflandırılmıştır. Araç kamerasından alınan görüntülerin işlenerek rota üzerinde ilgilenilen görüntüyü elde etme adımları Şekil 4.1' de gösterilmektedir.

Tablo 4.1. Model aracın farklı hızlarında rota takip süreleri.

Rota Numarası	Hız (m/sn)	Rota Tamamlama Süresi (sn)	Ortalama Tamamlanma Süresi (sn)
1	0.22	15.41	15.77
		16.35	
		15.56	
	0.27	15.09	14.72
		14.15	
		14.93	
2	0.22	22.89	22.58
		21.87	
		22.97	
	0.27	20.05	19.08
		18.31	
		18.87	
3	0.22	30.40	31.15
		31.45	
		31.60	
	0.27	28.48	28.44
		27.48	
		29.35	



a) b) c) d) e) f)

Şekil 4.1. Görüntü işleme adımları. a) Alınan görüntü, b) Grileşme, c) Bulanıklık, d) Otsu ve Canny, e) İşaret tespiti, f) Kırpılmış işaret resmi.

Otsu ve Canny algoritmalarının görüntüye birlikte uygulanması, model araç kamerasından alınan bir görüntüdeki işaretçileri elde etmek için Canny algoritmasının tek başına uygulanmasından daha başarılı sonuçlar vermiştir. Her iki algoritmanın da uygulandığı görüntüde, işaretçilerde ve kameradan uzaktaki nesnelerde daha fazla kenar bozulması olmakta ve kameraya yakın işaretçileri tespit etme başarısı artmaktadır. İlk olarak, bozulmayı arttırmak için görüntüye Gauss bulanıklığı uygulanmıştır. Şekil 4.2' de gösterildiği gibi, Şekil 4.2.a ve Şekil 4.2.b görüntüleri karşılaştırıldığında, Şekil 4.2. a' da kenar bozulmalarının meydana geldiği açıkça görülmektedir.

a) b)

Şekil 4.2. Otsu ve Canny algoritmalarının uygulanması. a) Otsu ve Canny algoritmaları uygulanmış görüntü, b) Canny algoritması uygulanmış görüntü.

Zemin Yolu Modelinin Testi ve Değerlendirilmesi

Bu çalışmada kapalı ortamlarda malzeme taşıma amaçlı derin öğrenmeye dayalı bir zemin yol modeli ve algoritması tasarlanmıştır. Modeli kullanan otonom araçların derin öğrenmeyi de kullanarak başarılı sonuçlar elde ettiği gözlemlenmiştir. Derin öğrenmenin model araç tarafından başarılı olarak kullanıldığı test edilmiştir. Ayrıca iç mekân bir endüstriyel ortamda sistemin başarısını değerlendirmek için simülasyonlar gerçekleştirilmiştir. Simülasyonlar AnyLogic [94] yazılımı kullanılarak yapılmıştır.

Zemin yolu modelinden oluşturulan rotalarda ilerleyen araç sayısı arttığında trafik yoğunluğu oluşmakta mıdır? Aracın ilerlediği rotada trafik yoğunluğu oluşması durumunda araç önceden belirlenen rotası dışındaki farklı bir rotaya nasıl yönlendirilmektedir? Bu soruların cevaplarını bulmak için simülasyon çalışması gerçekleştirilmiştir.

Önerdiğimiz modelin gerçek uygulamasının performansını görmek için AnyLogic yazılımında bir simülasyon senaryosu tasarlanmıştır. Senaryo için tasarlanan üretim ortamında farklı üretim departmanları, üretim hatları, CNC tezgâhları ve depo alanı bulunmaktadır. AGV' ler fabrika içerisinde malzeme taşımaktadırlar. Ortamda dolaşımda olacak AGV sayısı değişkendir. AGV' lerin rotalardaki dolaşımının test edilmesi, trafik yoğunluğunun belirlenmesi, trafik yoğunluğunda araçların davranışının izlenmesi için AGV sayısının değişken olması gerekmektedir. Bir adet araç kullanılarak aracın uygun olan en kısa rotadan veya boş olan uygun bir rotadan hedefine gitmesi izlenebilirken trafik yoğunluğunda araçların test edilmesi için rotalarda trafik yoğunluğunu oluşturacak araç sayısının olması önemli olmaktadır.

Simülasyon ortamında AGV' lerin dolaşacağı tasarlanan zemin yolu modeli fabrika zeminine yerleştirilmiştir. Zemin modelini oluşturan işaretçiler şu şekildedir: A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C1, C2, C3, C4, C5, D1, D2, D3, D4, E1, E2, F1, F2, F3, F4, F5, F6, F7, F8.

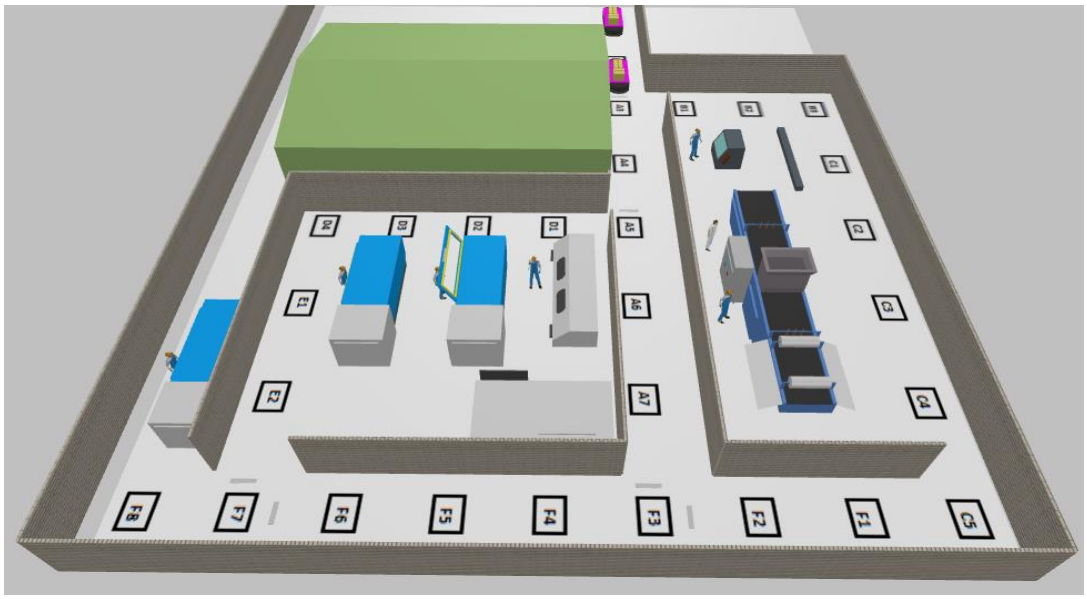
Kapalı bir ortamda bir rota üzerinde trafik yoğunluğunun meydana gelmesi araçların verilen hedefe gitmelerini geciktirmektedir. Bu nedenle bir aracın ilerleyeceği rotasını belirlemesinde trafik yoğunluğu da önemlidir. Bu çalışmada rotalardaki trafik yoğunluğunu belirlemede rotada ilerleyen araç sayıları dikkate alınmıştır. Rotalardaki araçların sayıları araçların uzunluklarına, rotaların araç kapasitelerine ve rotaların uzunluklarına göre değişkenlik göstermektedir. Aracın rotasını belirlemede yukarıda

önerilenlere ek olarak aracın ilerleyeceği rotadaki araç sayısı da önemlidir. Yapılan simülasyonlarda önerilen zemin yolu modeli kullanılarak tasarlanan örnek modelde farklı araç sayıları ile çalışmalar yapılmıştır.

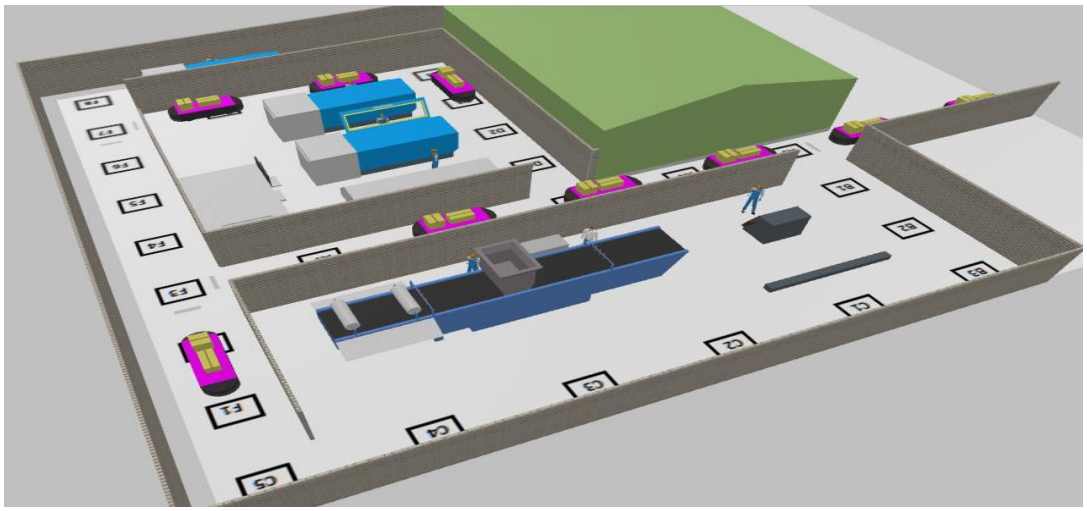
AnyLogic çok metotlu simülasyon modelleme yazılımıdır. AnyLogic etmen tabanlı modellemeyi ve ayrık olay modellemeyi desteklediği gibi sistem dinamiklerine de destek vermektedir.

AnyLogic görsel bir tasarım ekranına sahiptir. Bu yazılımda bir model oluşturmak için sürükle-bırak ile oluşturulan mantıksal algoritma kullanılmakla birlikte bu mantıksal algoritmanın bileşenlerine AnyLogic içinden veya dışarıdan eklenebilen görseller bağlanarak model hem iki boyutlu hem de üç boyutlu olarak izlenebilmektedir.

Yapılan simülasyon çalışmalarında önce tüm rotalar hesaplanmakta ve bu rotalardan en kısa olanı veya olanları bulunmaktadır. Rotaların seçiminde en kısa olanı, rotadaki araç sayısı, hareketine başlayacak araç için rotadaki güvenli mesafede başka bir aracın var olup olmadığı kontrol edilmektedir. AnyLogic yazılımı kullanılarak tasarlanan üretim ortamında çalıştırılan simülasyon çalışmasından alınan örnek ekran görüntüleri sekil 4.3, şekil 4.4 ve sekil 4.5 ve şekil 4.6’ da verilmiştir.

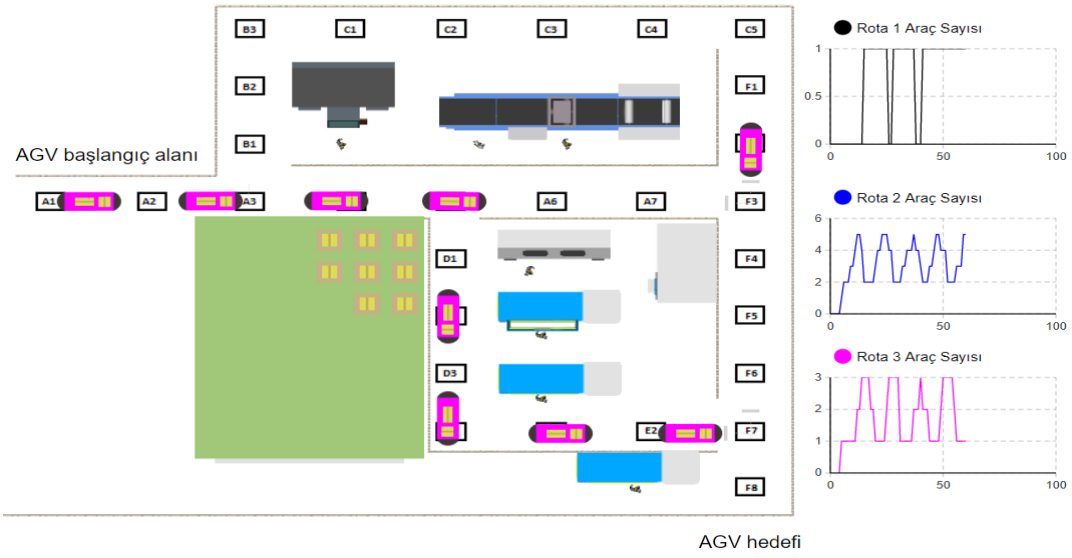


Şekil 4.3. Simülasyon çalışması örnek ekran görüntüsü 1.



Şekil 4.4. Simülasyon çalışması örnek ekran görüntüsü 2.

Şekil 4.3 ve şekil 4.4' de görüldüğü gibi AGV' ler baskın bir renk ile belirtilmiştir. Malzeme taşıma için araçların üst kısmı kullanılmaktadır. Ayrıca üç adet departman, bu departmanlarda bulunan üretim hatları mevcuttur. Şekil 4.5 tasarlanan fabrika ortamının üstten görünümü göstermektedir. Burada belirtilmiş olan AGV Başlangıç Alanı araçların bekleme alanını belirtmektedir. Şekil 4.6 ise sistemin çalışmasındaki mantığı yani algoritmayı görsel olarak sunmaktadır.



Şekil 4.5. Simülasyon çalışması iki boyutlu örnek ekran görüntüsü.

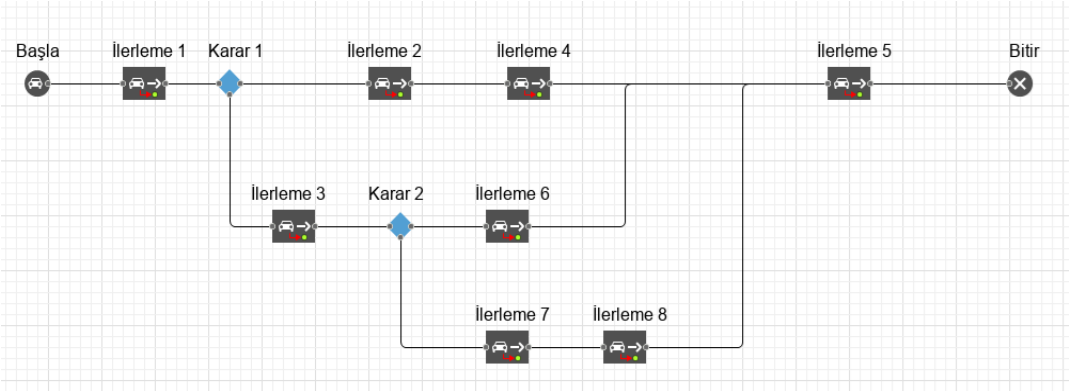
Şekil 4.6' da görünen Başla adımı simülasyonun başlamasını sağlamaktadır. Bu adımda simülasyona hedef olarak F8 adresi ve zemin modeli A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C1, C2, C3, C4, C5, D1, D2, D3, D4, E1, E2, F1, F2, F3, F4, F5, F6, F7, F8, A3-B1, B3-C1, C5-F1, A5-D1, D4-E1, E2-F7, A7-F3 vektörü olarak

verilmektedir. Yine bu adımda bu vektör kullanılarak aşağıda gösterilen rotalar Java kodları kullanılarak hesaplanmaktadır:

R1 = Rotalar [0]: A1, A2, A3, B1, B2, B3, C1, C2, C3, C4, C5, F1, F2, F3, F4, F5, F6, F7, F8

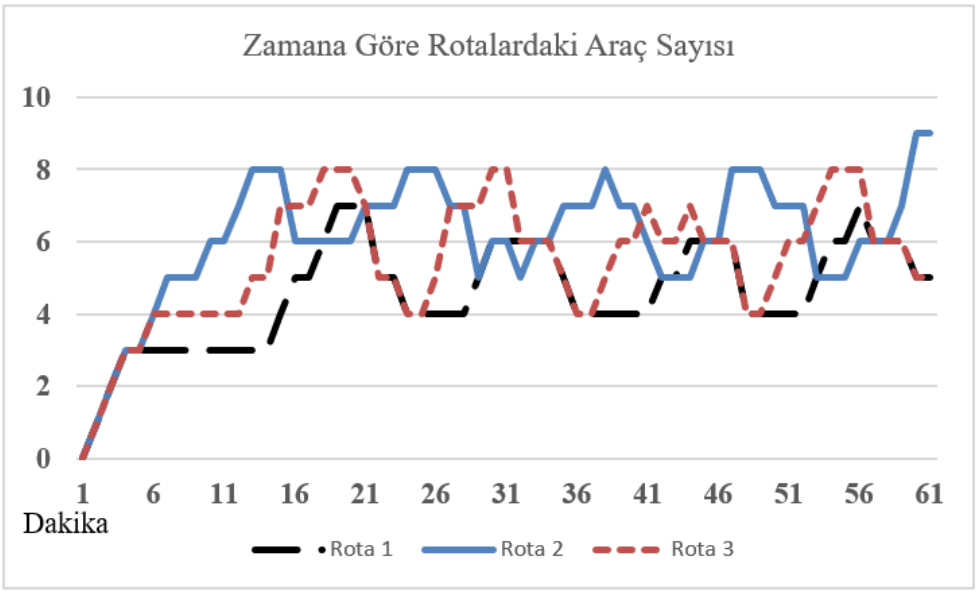
R2 = Rotalar [1]: A1, A2, A3, A4, A5, D1, D2, D3, D4, E1, E2, F7, F8

R3 = Rotalar [2]: A1, A2, A3, A4, A5, A6, A7, F3, F4, F5, F6, F7, F8



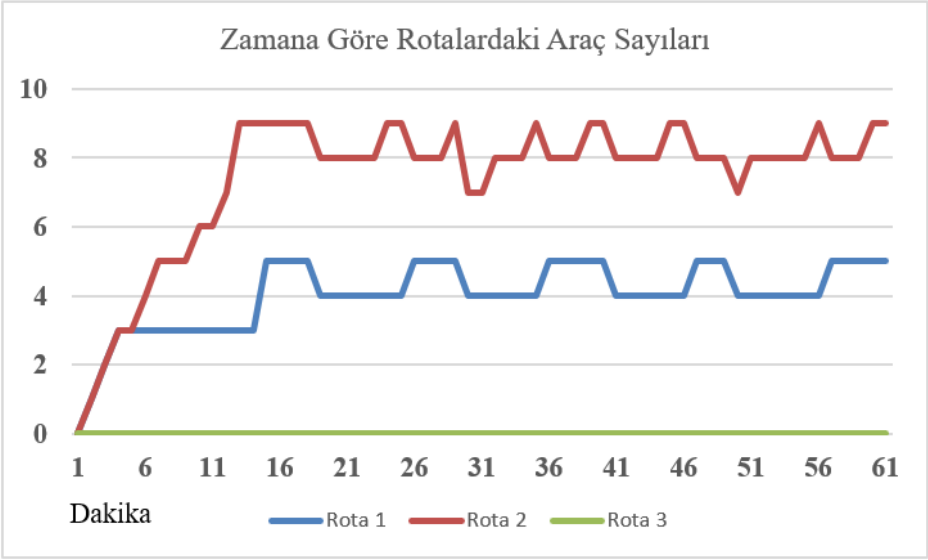
Şekil 4.6. Simülasyon çalışması mantık bölümü ekran görüntüsü.

Geliştirilen algoritmaya göre bu rotalardan en kısa olanı hesaplanmaktadır. Bu vektörde hedefe giden en kısa rota iki tanedir. Bu durumda rota indisi küçük olan R2 rotası seçilmektedir. R2 rotasında araç yoğunluğu olduğunda ise R3 rotası ve her iki rotada trafik yoğunluğu olduğunda R1 rotası seçilmektedir. Eğer rota başlangıcında başka bir araç bulunmuyorsa yeni bir araç üretilmekte ve seçilen rotada İlerleme 1 adımı kullanılarak aracın ilerlemesi sağlanmaktadır. Karar 1 ve Karar 2 adımlarında seçilen rotaya göre yönlendirme yapılmaktadır. İlerleme 2, İlerleme 3, İlerleme 4, İlerleme 5, İlerleme 6, İlerleme 7 ve İlerleme 8 adımları seçilen rotaya göre araçların ilerlemesini sağlamaktadırlar. Her bir adımda araçların hangi işaretçilerde bulunduğu bilgileri tutulmaktadır. Bu bilgiler kullanılarak rotalardaki araçların konumu ve sayıları elde edilmektedir.



Şekil 4.7. Zamana göre üç rotanın araç sayısındaki değişim.

Algoritmaya göre zamana göre rotalardaki araç sayısını gösteren grafik şekil 4.7' de verilmiştir. Şekilden de görüldüğü gibi düz çizgiyle gösterilen R2 rotası ilk seçilen rota olduğundan araçların en çok kullandığı rotadır. R2 rotasındaki araç sayısı belirli bir değere ulaştığında bu rotadaki trafik yoğunluğunu azaltmak için bir sonraki araç R3 rotasını seçmektedir. R2 ve R3 rotalarında trafik yoğunluğu olduğunda en uzun rota olan R1 rotası seçilerek R2 ve R3 rotalarındaki araç yoğunluğu azaltılmaktadır. Kullanılan algoritmaya göre ilk önce en kısa rota önceliği kullanılarak daha sonra da rotalardaki araç yoğunluğuna göre rotalar arasından seçim yapılmaktadır.



Şekil 4.8. Zamana göre iki rotanın araç sayısındaki değişim.

R3 rotasındaki A6 işaretçisine çalışma zamanında engel eklenerek bu işaretçi kapatılırsa araçlar önce R2 rotasını seçecek, R2 rotasında trafik yoğunluğu oluştuğunda ve R3 rotasında bir sorun olduğundan R1 rotasını seçeceklerdir. Bu durumda rotalardaki araç sayısı şekil 4.8' de gösterildiği gibi olmaktadır.

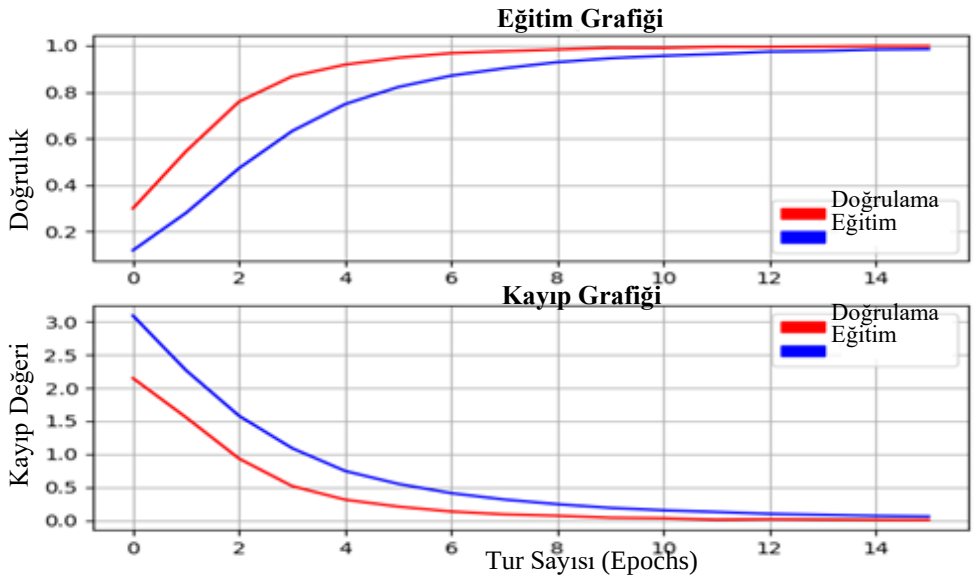
Simülasyon sonuçlarından da görüldüğü gibi geliştirilen zemin yolu modeli ve algoritması kullanılarak rotalardaki trafik yoğunluğu dengelenmektedir.

5. TARTIŞMA VE SONUÇ

MobileNet, ImageNet veri seti ile eğitilmiş bir modeldir. Bu modelin ağırlıkları transfer öğrenme kullanılarak elde edilmiştir. Çalışmada kullanılan eğitim seti, ImageNet veri setinden farklıdır. Bu nedenle model katmanlarının ağırlıkları eğitim sırasında kullanılan yeni görüntülerle güncellenmiştir. Son tam bağlı katmanlar hariç tüm katmanlar MobileNet modelinde kullanılmıştır. Modeldeki sınıflandırma katmanına toplu normalleştirme katmanı eklenmiştir. Toplu normalleştirme, bazı durumlarda bırakma ihtiyacını ortadan kaldırarak bir düzenleyici görevi görür [95]. Ezberlemeyi engellemek için toplu normalleştirme kullanılmıştır.

Model araçlardaki düşük işlemci gücünden dolayı görüntü işleme süreci yavaş olmaktadır. Bu nedenle model eğitimi için görüntü boyutu 160 × 160 × 3 piksel olarak ayarlanmıştır. Başlangıç öğrenme oranı 0.0001 ve kayıp fonksiyonu kategorik çapraz entropi olarak seçilmiştir. Sınıf sayısı ikiden fazla olduğu için kategorik çapraz entropi fonksiyonu kullanılmıştır. Belirlenen öğrenme oranı ile kayıp değeri arka arkaya üç kez azaltılamıyorsa öğrenme oranı belirlenen delta değeri kadar azaltılmaktadır. Belirlenen minimum delta değeri 0.0001 olarak alınmıştır. Veri setinde birbirine benzeyen sınıflar bulunmaktadır. Bu, modeli eğitirken performans grafiğinde sorunlara neden olmaktadır. Parti boyutu değeri büyük olduğunda, model aracının verileri işlemesi daha fazla zaman almaktadır. Bu değer küçük olduğunda model gürültüleri öğrenmekte ve genellemeden uzaklaşmaktadır. Bu nedenle, testlerin sonunda her tur sayısı (epoch) için 32'lik bir mini parti boyutu (mini batch size) seçilmiştir. Optimize edici olarak Adam kullanılmıştır. Son katmanın aktivasyon fonksiyonu 17 sınıftan oluştuğu için Softmax seçilmiştir. Doğrulama kaybı değeri 0.009'dan az olduğunda model eğitimi sonlandırılmıştır. Model eğitimi 16 tur sayısında tamamlanmıştır.

Modelin eğitim ve kayıp grafikleri Şekil 5.1' de verilmiştir. Şekilden de görüldüğü gibi eğitim kaybı ve doğrulama kaybı eğrilerinin birbirini takip ettiği ve çok yakın seviyelerde ilerlediği görülmektedir. Modelimizin performansını değerlendirmek için kesinlik oranı, geri çağırma oranı ve F1 puanı ölçümleri kullanılmıştır.



Şekil 5.1. Model için eğitim ve kayıp grafikleri.

Bu performans ölçütleri aşağıda verilmiştir:

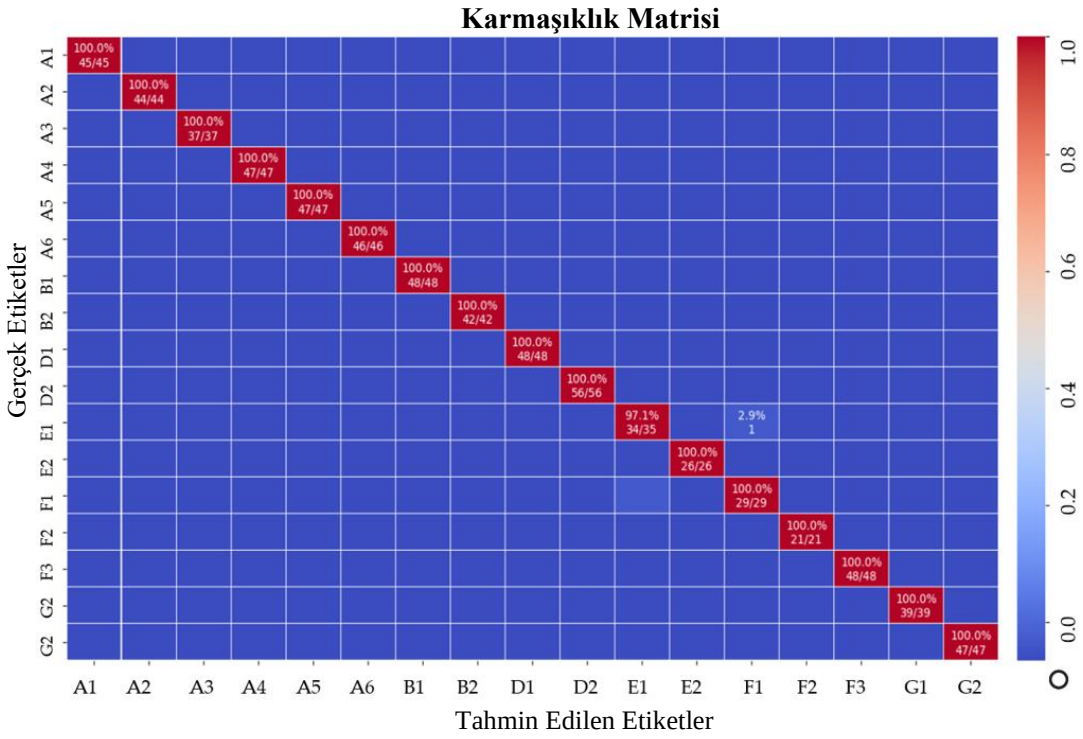
$$Kesinlik = \frac{Doğru\ Pozitif}{Doğru\ Pozitif + Yanlış\ Pozitif}$$

$$Duyarlılık = \frac{Doğru\ Pozitif}{Doğru\ Pozitif + Yanlış\ Pozitif}$$

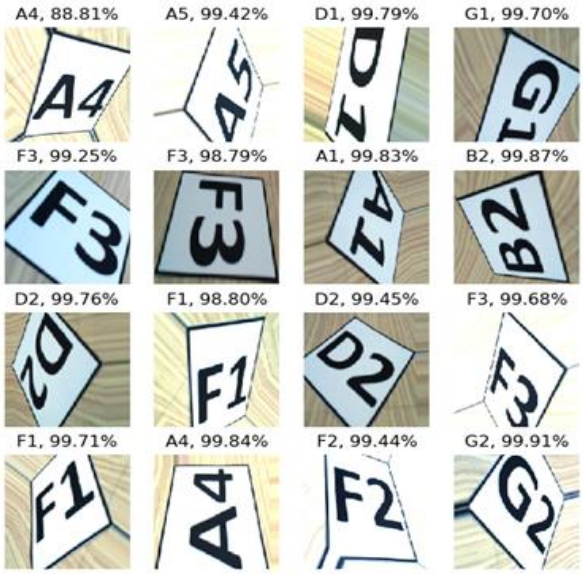
$$F1 - puan = \frac{2\ x\ Kesinlik\ x\ Duyarlılık}{Kesinlik + Duyarlılık}$$

Kesinlik metriği, model tarafından pozitif olarak tahmin edilen değerlerden kaçının pozitif olduğunu göstermektedir. Geri çağırma metriği, modelin pozitif olarak tahmin etmesi gereken değerlerden kaçının pozitif tahmin edildiğini göstermektedir. Çalışmada veri setinde birden fazla sınıf olduğu için model performansı için sınıflandırma doğruluk metriğinin yanı sıra kesinlik, hatırlama ve F1-skor metrikleri de kullanılmıştır. Şekil 5.2' de gösterilen karışıklık matrisinde E1 işaretçisi dışındaki tüm işaretçilerin doğru şekilde sınıflandırıldığı görülmektedir. Matris incelendiğinde E1 olarak sınıflandırılması gereken işaretçinin yanlış şekilde F1 olarak sınıflandırıldığı görülmektedir. 35 E1 işaretçisinden otuz dördü doğru şekilde sınıflandırılmıştır. Bunun nedeni, E1 ve F1 işaretçilerinin benzer olmasıdır. Eğitilmiş derin öğrenme

modelini test etmek için, daha önce hiç verilmemiş toplam 796 işaretçi görüntüsü modele girdi olarak girilmiştir.



Şekil 5.2. Modelin karmaşıklık matrisi.



Şekil 5.3. Eğitilen CNN modeli için test seti sınıflandırma sonucu.

AVM kameradan alınan görüntüden ön işleme ile elde edilen ROI görüntüsü, eğitilen modele girdi olarak verilmiştir. Bu görüntüyü sınıflandıracak modelin performans metrikleri Tablo 5.1' de, gecikme süreleri ise Tablo 5.2.'te verilmiştir. Tablodan ortalama gecikme süresinin 0.01449 s olduğu görülmektedir.

Tablo 5.1. Modelin performans metrikleri.

Etiket	Hassasiyet (Precision)	Geri çağırma (Recall)	F1-Skor	Test edilen görüntü sayısı
A1	1.00	1.00	1.00	45
A2	1.00	1.00	1.00	44
A3	1.00	1.00	1.00	37
A4	1.00	1.00	1.00	47
A5	1.00	1.00	1.00	47
A6	1.00	1.00	1.00	46
B1	1.00	1.00	1.00	48
B2	1.00	1.00	1.00	42
D1	1.00	1.00	1.00	48
D2	1.00	1.00	1.00	56
E1	1.00	0.97	0.99	35
E2	1.00	1.00	1.00	26
F1	0.97	1.00	0.98	29
F2	1.00	1.00	1.00	21
F3	1.00	1.00	1.00	48
G1	1.00	1.00	1.00	39
G2	1.00	1.00	1.00	47
Ortalama	99.82	99.8	99.8	

Tablo 5.2. Model için gecikme testi süreleri.

Ölçüm Numarası	Model Gecikme Süreleri (Saniye)
1	0.01447
2	0.01401
3	0.01511
4	0.01440
5	0.01447
Ortalama	0.01449

Üretim alanlarında kullanılan geleneksel zemin yolu yöntemleri, trafik sıkışıklığına neden olabilecek sınırlı sayıda yol güzergahına sahiptir. Bu çalışmada, otonom araçlar için bir zemin yolu modeli ve algoritmasına dayalı bir iç mekân konumlandırma sistemi geliştirilmiştir. İşaretçilerden oluşan bir zemin yolu modeli geliştirilmiştir. Zemin yolu modelinde belirli bir hedefe giden tüm rotalar, dinamik rota planlayıcı kullanılarak oluşturulur. Araç, bu rotalardan en az araç sayısı ile en kısa alternatif rotayı seçer ve başka bir araca çarpmadan hedefine gider. Zemin yolu modelindeki işaretler, düşük çözünürlüklü bir kamera ile tespit edilebilmektedir. Zemin yolu modeli artan ve azalan işaretlerle oluşturulduğundan, araç modeli art arda artan veya azalan bir rotayı kolaylıkla takip edebilir. Bu alternatif güzergahlar kullanılarak trafik yoğunluğu en aza indirilebilir ve iş yükü dengelenebilir. Zemin yolu modelindeki araçlar, tüm araçlar için işaretleyici adreslerini yayınlamaktadırlar. Aracın güzergahında başka bir araç varsa önerilen algoritmada araçlar arasında en az dört işaretçi olana kadar beklemektedir. Zemin yolu modeli ve algoritması kullanıldığında, bir işaretçi adresinde birden fazla aracın bulunmasına izin verilmemekte ve araçların çarpışması engellenmektedir. Geliştirilen zemin yolu modelini kullanarak hedefe birden fazla rota oluşturmanın mümkün olduğu gösterilmiştir.

Transfer öğrenme kullanılarak eğitilen CNN modelinin sonuçları incelendiğinde 35 E1 belirtecinden 34'ünün doğru sınıflandırıldığı görülmüştür. Test setinde 35 E1 işaretleyici görüntüsü içeren E1 işaretçisi için en düşük hatırlama metrik değeri 0.97 ve F1-skor metriği 0.99 elde edilmiştir. En düşük kesinlik metriği 0.97 ve F1-skor metriği 0.98, 29 F1 işaretleyici test görüntüsü içeren F1 işaretçisi için elde edilmiştir. Test seti görüntüleri eğitilmiş CNN modeline uygulandığında en düşük % 88.81 ile A4 işaretleyici görüntü, en yüksek oran ise % 99.91 ile G2 işaretleyici görüntü olarak

sınıflandırılmıştır. Ayrıca modelin tüm belirteçleri doğru bir şekilde sınıflandırdığı gözlemlenmiştir.

Bu çalışmada, iç mekân konumlandırma sistemi kullanılarak birçok alternatif güzergâh arasından belirlenen güzergâh seçilmiş ve takip edilmiştir. Sonuçlar, geliştirilen zemin yolu modeli, algoritması ve derin öğrenme modelinin hem gerçek bir üretim ortamında hem de depolama ve sağlık gibi hizmet sektörü alanlarında başarıyla kullanılabileceğini göstermiştir.

Gelecekteki potansiyel çalışmalardan biri, deneysel olarak tasarlanan ve modellenen zemin yolu modelinin gerçek endüstriyel ortamlarda uygulanabilmesidir. Bu çalışmada kullanılan otonom araçta kamera sabitlenmiştir. Aksiyon kameraları gelecekteki çalışmalarda kullanılabilir.

KAYNAKLAR

[1] S. H. Jung, D. G. Woo, J. do Han, J. K. Shin, ve K. R. Kim, "Corresponding Author : kris@seiltnn.com CNN based Factory Lane Marker Recognition for Indoor Path tracking of Automated Guided Vehicle".

[2] H. Q. T. Ngo, T. P. Nguyen, ve H. Nguyen, "Research and develop of AGV platform for the logistics warehouse environment", içinde *Advances in Intelligent Systems and Computing*, 2019, c. 881, ss. 455-465. doi: 10.1007/978-3-030-02683-7_32.

[3] W. Xing, L. Peihuang, Y. Jun, Q. Xiaoming, ve T. Dunbing, "Intersection recognition and guide-path selection for a vision-based agv in a bidirectional flow network", *Int J Adv Robot Syst*, c. 11, sy 1, ss. 1-17, 2014, doi: 10.5772/58218.

[4] P. Reddy Gutta, V. Sai Chinthala, R. Venkatesh Manchoju, V. Charan Mvn, ve R. Purohit, "A Review on Facility Layout Design of An Automated Guided Vehicle in Flexible Manufacturing System", *Mater Today Proc*, c. 5, sy 2, ss. 3981-3986, 2018, doi: 10.1016/j.matpr.2017.11.656.

[5] L. Yao ve J. Miller, "Tiny ImageNet Classification with Convolutional Neural Networks", 2015, [Çevrimiçi]. Erişim adresi: http://cs231n.stanford.edu/reports/leonyao_final.pdf

[6] X. Yu, X. Wu, C. Luo, ve P. Ren, "Deep learning in remote sensing scene classification: a data augmentation enhanced convolutional neural network framework", *GIsci Remote Sens*, c. 54, sy 5, ss. 741-758, 2017, doi: 10.1080/15481603.2017.1323377.

[7] N. Smolyanskiy, A. Kamenev, J. Smith, ve S. Birchfield, "Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness", *IEEE International Conference on Intelligent Robots and Systems*, c. 2017-Septe, ss. 4241-4247, 2017, doi: 10.1109/IROS.2017.8206285.

[8] Q. Wang ve C. Wang, "Exploration of port intelligent AGV path tracking based on vision", *Journal of Intelligent & Fuzzy Systems*, c. 38, sy 2, ss. 1281-1285, 2019, doi: 10.3233/jifs-179491.

[9] Y. Wang, H. Ding, J. Yuan, ve H. Chen, "Output-feedback triple-step coordinated control for path following of autonomous ground vehicles", *Mech Syst Signal Process*, c. 116, ss. 146-159, 2019, doi: 10.1016/j.ymssp.2018.06.011.

[10] İ. Cil, F. Arisoy, E. Özgürbüz, A. Y. Cil, ve H. Kılınç, "Indoor Positioning Technology Selection Using a Combined AHP and PROMETHEE Method at SEDEF Shipyard", *Journal of ETA Maritime Science*, c. 10, sy 2, ss. 108-123, Haz. 2022, doi: 10.4274/jems.2022.47550.

[11] Y. Zhou, H. Hu, Y. Liu, ve Z. Ding, "Collision and deadlock avoidance in multirobot systems: A distributed approach", *IEEE Trans Syst Man Cybern Syst*, c. 47, sy 7, ss. 1712-1726, Tem. 2017, doi: 10.1109/TSMC.2017.2670643.

[12] H.-W. Cheong ve H. Lee, "Concept Design of AGV (Automated Guided Vehicle) Based on Image Detection and Positioning", *Procedia Comput Sci*, c. 139, ss. 104-107, 2018, doi: 10.1016/j.procs.2018.10.224.

[13] W. Małopolski, "A sustainable and conflict-free operation of AGVs in a square topology", *Comput Ind Eng*, c. 126, ss. 472-481, Ara. 2018, doi: 10.1016/j.cie.2018.10.002.

[14] M. A. Guney ve I. A. Raptis, "Dynamic prioritized motion coordination of multi-AGV systems", *Rob Auton Syst*, c. 139, May. 2021, doi: 10.1016/j.robot.2020.103534.

[15] Y. Zhao, X. Liu, G. Wang, S. Wu, ve S. Han, "Dynamic Resource Reservation Based Collision and Deadlock Prevention for Multi-AGVs", *IEEE Access*, c. 8, ss. 82120-82130, 2020, doi: 10.1109/ACCESS.2020.2991190.

[16] K. Guo, J. Zhu, ve L. Shen, "An Improved Acceleration Method Based on Multi-Agent System for AGVs Conflict-Free Path Planning in Automated Terminals", *IEEE Access*, c. 9, ss. 3326-3338, 2021, doi: 10.1109/ACCESS.2020.3047916.

[17] A. Vale, R. Ventura, P. Lopes, ve I. Ribeiro, "Assessment of navigation technologies for automated guided vehicle in nuclear fusion facilities", *Rob Auton Syst*, c. 97, ss. 153-170, Kas. 2017, doi: 10.1016/j.robot.2017.08.006.

[18] E. A. Oyekanlu *vd.*, "A review of recent advances in automated guided vehicle technologies: Integration challenges and research areas for 5G-based smart manufacturing applications", *IEEE Access*, c. 8. Institute of Electrical and Electronics Engineers Inc., ss. 202312-202353, 2020. doi: 10.1109/ACCESS.2020.3035729.

[19] L. Lynch, T. Newe, J. Clifford, J. Coleman, J. Walsh, ve D. Toal, "Automated Ground Vehicle (AGV) and Sensor Technologies- A Review", içinde *2018 12th International Conference on Sensing Technology (ICST)*, 2018, ss. 347-352. doi: 10.1109/ICSensT.2018.8603640.

[20] K. Murakami, "Time-space network model and MILP formulation of the conflict-free routing problem of a capacitated AGV system", *Comput Ind Eng*, c. 141, Mar. 2020, doi: 10.1016/j.cie.2020.106270.

[21] C. Lu *vd.*, "Deep Reinforcement Learning for Solving AGVs Routing Problem", içinde *Verification and Evaluation of Computer and Communication Systems: 14th International Conference, VECoS 2020, Xi'an, China, October 26–27, 2020, Proceedings 14*, 2020, ss. 222-236.

[22] I. Holovatenko ve A. Pysarenko, "Energy-Efficient Path-Following Control System of Automated Guided Vehicles", *Journal of Control, Automation and Electrical Systems*, c. 32, sy 2, ss. 390-403, Nis. 2021, doi: 10.1007/s40313-020-00668-8.

[23] C. Hu, R. Wang, F. Yan, ve N. Chen, "Output constraint control on path following of four-wheel independently actuated autonomous ground vehicles", *IEEE Trans Veh Technol*, c. 65, sy 6, ss. 4033-4043, 2016, doi: 10.1109/TVT.2015.2472975.

[24] V. John, Z. Liu, S. Mita, C. Guo, ve K. Kidono, "Real-time road surface and semantic lane estimation using deep features", *Signal Image Video Process*, c. 12, sy 6, ss. 1133-1140, 2018, doi: 10.1007/s11760-018-1264-2.

[25] M. B. B. Mahaleh ve S. A. Mirroshandel, "Harmony search path detection for vision based automated guided vehicle", *Rob Auton Syst*, c. 107, ss. 156-166, 2018, doi: 10.1016/j.robot.2018.06.008.

[26] J. L. Diaz Bayona, J. D. Gonzalez Pilonieta, R. M. Molina Lache, ve S. Roa Prada, "Development of a control system for path following applications in an AGV using computer vision", içinde *2020 9th International Congress of Mechatronics Engineering and Automation, CIIMA 2020 - Conference Proceedings*, Kas. 2020. doi: 10.1109/CIIMA50553.2020.9290184.

[27] Y. Zhu ve N. Guo, "Unmanned vehicle route tracking method based on video image processing", *Jordan Journal of Mechanical and Industrial Engineering*, c. 14, sy 1, ss. 139-147, 2020.

[28] J. Lee, C. H. Hyun, ve M. Park, "A vision-based automated guided vehicle system with marker recognition for indoor use", *Sensors (Switzerland)*, c. 13, sy 8, ss. 10052-10073, 2013, doi: 10.3390/s130810052.

[29] M. Revilloud, D. Gruyer, ve M. C. Rahal, "A lane marker estimation method for improving lane detection", *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, ss. 289-295, 2016, doi: 10.1109/ITSC.2016.7795569.

[30] Y. Liu, X. Zhang, T. Qu, D. Yin, ve S. Deng, "Intelligent robot motion trajectory planning based on machine vision", *International Journal of Systems Assurance Engineering and Management*, 2022, doi: 10.1007/s13198-021-01559-0.

[31] W. Lin, X. Ren, J. Hu, Y. He, Z. Li, ve M. Tong, "Fast, robust and accurate posture detection algorithm based on Kalman filter and SSD for AGV", *Neurocomputing*, c. 316, ss. 306-312, 2018, doi: 10.1016/j.neucom.2018.08.006.

[32] Y. Sun, T. Su, ve Z. Tu, "Faster R-CNN based autonomous navigation for vehicles in warehouse", *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, sy Figure 1, ss. 1639-1644, 2017, doi: 10.1109/AIM.2017.8014253.

[33] R. Kamoshida ve Y. Kazama, "Acquisition of Automated Guided Vehicle Route Planning Policy Using Deep Reinforcement Learning", *6th IEEE International Conference on Advanced Logistics and Transport*, ss. 1-6, 2017.

[34] "Web Of Science", *Web Of Science Veritabanı*. https://www.webofscience.com/wos/woscc/basic-search (erişim 25 Aralık 2022).

[35] "Scopus", *Scopus Veritabanı*. https://www.scopus.com/search/form.uri?display=basic#basic (erişim 25 Aralık 2022).

[36] "VOSviewer", *VOSviewer Bibliyografik Görselleştirme Yazılımı* . https://www.vosviewer.com/ (erişim 25 Aralık 2022).

[37] Z. Liu, Q. Liu, W. Xu, L. Wang, ve Z. Zhou, "Robot learning towards smart robotic manufacturing: A review", *Robot Comput Integr Manuf*, c. 77, Eki. 2022, doi: 10.1016/j.rcim.2022.102360.

[38] Y. Lecun, Y. Bengio, ve G. Hinton, "Deep learning", *Nature*, c. 521, sy 7553. Nature Publishing Group, ss. 436-444, 27 Mayıs 2015. doi: 10.1038/nature14539.

[39] C. Shorten, T. M. Khoshgoftaar, ve B. Furht, "Deep Learning applications for COVID-19", *J Big Data*, c. 8, sy 1, Ara. 2021, doi: 10.1186/s40537-020-00392-9.

[40] S. Chisaki, R. Fuji-Hara, ve N. Miyamoto, "Combinatorial designs for deep learning", *Journal of Combinatorial Designs*, c. 28, sy 9, ss. 633-657, Eyl. 2020, doi: 10.1002/jcd.21720.

[41] O. Deniz, A. Pedraza, N. Vallez, J. Salido, ve G. Bueno, "Robustness to adversarial examples can be improved with overfitting", *International Journal of Machine Learning and Cybernetics*, c. 11, sy 4, ss. 935-944, Nis. 2020, doi: 10.1007/s13042-020-01097-4.

[42] J. Lei, B. Y. Zhang, ve H. F. Ling, "Deep learning face representation by fixed erasing in facial landmarks", *Multimed Tools Appl*, c. 78, sy 19, ss. 27703-27718, Eki. 2019, doi: 10.1007/s11042-019-07892-8.

[43] S. H. Oh, S. W. Han, B. S. Choi, G. W. Kim, ve K. S. Lim, "Deep feature learning for person re-identification in a large-scale crowdsourced environment", *Journal of Supercomputing*, c. 74, sy 12, ss. 6753-6765, Ara. 2018, doi: 10.1007/s11227-017-2221-5.

[44] X. Hu, L. Chu, J. Pei, W. Liu, ve J. Bian, "Model complexity of deep learning: a survey", *Knowl Inf Syst*, c. 63, sy 10, ss. 2585-2619, Eki. 2021, doi: 10.1007/s10115-021-01605-0.

[45] H. Rajadurai ve U. D. Gandhi, "An empirical model in intrusion detection systems using principal component analysis and deep learning models", *Comput Intell*, c. 37, sy 3, ss. 1111-1124, Ağu. 2021, doi: 10.1111/coin.12342.

[46] Z. Cui, F. Li, ve W. Zhang, "Bat algorithm with principal component analysis", *International Journal of Machine Learning and Cybernetics*, c. 10, sy 3, ss. 603-622, Mar. 2019, doi: 10.1007/s13042-018-0888-4.

[47] R. Moradi, R. Berangi, ve B. Minaei, "A survey of regularization strategies for deep models", *Artif Intell Rev*, c. 53, sy 6, ss. 3947-3986, Ağu. 2020, doi: 10.1007/s10462-019-09784-7.

[48] A. Thakkar ve R. Lohiya, "Analyzing fusion of regularization techniques in the deep learning-based intrusion detection system", *International Journal of Intelligent Systems*, c. 36, sy 12, ss. 7340-7388, Ara. 2021, doi: 10.1002/int.22590.

[49] G. James, D. Witten, T. Hastie, ve R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R* . Springer, 2013. [Çevrimiçi]. Erişim adresi: https://faculty.marshall.usc.edu/gareth-james/ISL/

[50] A. Effland, E. Kobler, K. Kunisch, ve T. Pock, "Variational Networks: An Optimal Control Approach to Early Stopping Variational Methods for Image Restoration", *J Math Imaging Vis*, c. 62, sy 3, ss. 396-416, Nis. 2020, doi: 10.1007/s10851-019-00926-8.

[51] M. Bentoumi, M. Daoud, M. Benaouali, ve A. Taleb Ahmed, "Improvement of emotion recognition from facial images using deep learning and early stopping cross validation", *Multimed Tools Appl*, c. 81, sy 21, ss. 29887-29917, Eyl. 2022, doi: 10.1007/s11042-022-12058-0.

[52] Y. LeCun, L. Bottou, Y. Bengio, ve P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, c. 86, sy 11, ss. 2278-2323, 1998, doi: 10.1109/5.726791.

[53] A. Krizhevsky, I. Sutskever, ve G. E. Hinton., "Imagenet classification with deep convolutional neural networks. In Advances in neural information", *Adv Neural Inf Process Syst*, ss. 1097-1105, 2012, doi: 10.1016/B978-008046518-0.00119-7.

[54] K. Simonyan ve A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", Eyl. 2014, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1409.1556

[55] C. Szegedy *vd.*, "Going Deeper with Convolutions", Eyl. 2014, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1409.4842

[56] K. He, X. Zhang, S. Ren, ve J. Sun, "Deep residual learning for image recognition", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, c. 2016-Decem, ss. 770-778, 2016, doi: 10.1109/CVPR.2016.90.

[57] G. Huang, Z. Liu, L. van der Maaten, ve K. Q. Weinberger, "Densely Connected Convolutional Networks", Ağu. 2016, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1608.06993

[58] Y. Guo, Y. Liu, E. M. Bakker, Y. Guo, ve M. S. Lew, "CNN-RNN: a large-scale hierarchical image classification framework", *Multimed Tools Appl*, c. 77, sy 8, ss. 10251-10271, Nis. 2018, doi: 10.1007/s11042-017-5443-x.

[59] Q. Yin, R. Zhang, ve X. Shao, "CNN and RNN mixed model for image classification", *MATEC Web of Conferences*, c. 277, s. 02001, 2019, doi: 10.1051/matecconf/201927702001.

[60] J. Donahue *vd.*, "Long-term Recurrent Convolutional Networks for Visual Recognition and Description", Kas. 2014, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1411.4389

[61] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, ve W. Xu, "CNN-RNN: A Unified Framework for Multi-label Image Classification".

[62] E. Kordi Ghasrodashti ve N. Sharma, "Hyperspectral image classification using an extended Auto-Encoder method", *Signal Process Image Commun*, c. 92, Mar. 2021, doi: 10.1016/j.image.2020.116111.

[63] A. Gogna, A. Majumdar, ve R. Ward, "Semi-supervised Stacked Label Consistent Autoencoder for Reconstruction and Analysis of Biomedical Signals", *IEEE Trans Biomed Eng*, c. 64, sy 9, ss. 2196-2205, Eyl. 2017, doi: 10.1109/TBME.2016.2631620.

[64] W. Luo, J. Li, J. Yang, W. Xu, ve J. Zhang, "Convolutional sparse autoencoders for image classification", *IEEE Trans Neural Netw Learn Syst*, c. 29, sy 7, ss. 3289-3294, Tem. 2018, doi: 10.1109/TNNLS.2017.2712793.

[65] I. Goodfellow *vd.*, "Generative adversarial networks", *Commun ACM*, c. 63, sy 11, ss. 139-144, Eki. 2020, doi: 10.1145/3422622.

[66] S. Motamed ve F. Khalvati, "Multi-class Generative Adversarial Nets for Semi-supervised Image Classification", Şub. 2021, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/2102.06944

[67] D. Wang, Z. Lu, Y. Xu, Z. Wang, A. Santella, ve Z. Bao, "Cellular Structure Image Classification with Small Targeted Training Samples", *IEEE Access*, c. 7, ss. 148967-148974, 2019, doi: 10.1109/ACCESS.2019.2940161.

[68] J. Cao, Y. Pang, X. Li, ve J. Liang, "Randomly translational activation inspired by the input distributions of ReLU", *Neurocomputing*, c. 275, ss. 859-868, 2018, doi: 10.1016/j.neucom.2017.09.031.

[69] M. Eder, M. Reip, ve G. Steinbauer, "Creating a robot localization monitor using particle filter and machine learning approaches", *Applied Intelligence*, c. 52, sy 6, ss. 6955-6969, Nis. 2022, doi: 10.1007/s10489-020-02157-6.

[70] O. Christopher, "Understanding LSTM networks", *Understanding LSTM Networks-Colah's Blog. Github*, 2015.

[71] S. Hochreiter ve J. Schmidhuber, "Long Short-Term Memory", *Neural Comput*, c. 9, sy 8, ss. 1735-1780, Kas. 1997, doi: 10.1162/neco.1997.9.8.1735.

[72] K. Weiss, T. M. Khoshgoftaar, ve D. D. Wang, *A survey of transfer learning*, c. 3, sy 1. Springer International Publishing, 2016. doi: 10.1186/s40537-016-0043-6.

[73] O. Day ve T. M. Khoshgoftaar, "A survey on heterogeneous transfer learning", *J Big Data*, c. 4, sy 1, 2017, doi: 10.1186/s40537-017-0089-0.

[74] A. G. Howard *vd.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", Nis. 2017, [Çevrimiçi]. Erişim adresi: http://arxiv.org/abs/1704.04861

[75] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, ve E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles", *IEEE Transactions on Intelligent Vehicles*, c. 1, sy 1, ss. 33-55, 2016, doi: 10.1109/TIV.2016.2578706.

[76] S. Chen, Z. Jian, Y. Huang, Y. Chen, Z. Zhou, ve N. Zheng, "Autonomous driving: cognitive construction and situation understanding", *Science China Information Sciences*, c. 62, sy 8. Science in China Press, 01 Ağustos 2019. doi: 10.1007/s11432-018-9850-9.

[77] M. Fazekas, P. Gáspár, ve B. Németh, "Calibration and improvement of an odometry model with dynamic wheel and lateral dynamics integration", *Sensors (Switzerland)*, c. 21, sy 2, ss. 1-29, Oca. 2021, doi: 10.3390/s21020337.

[78] J. Patoliya, H. Mewada, M. Hassaballah, M. A. Khan, ve S. Kadry, "A robust autonomous navigation and mapping system based on GPS and LiDAR data for unconstraint environment", *Earth Sci Inform*, 2022, doi: 10.1007/s12145-022-00791-x.

[79] A. Valldeperes *vd.*, "Wireless inertial measurement unit (IMU)-based posturography", *European Archives of Oto-Rhino-Laryngology*, c. 276, sy 11, ss. 3057-3065, Kas. 2019, doi: 10.1007/s00405-019-05607-1.

[80] F. Wang, T. Su, X. Jin, Y. Zheng, J. Kong, ve Y. Bai, "Indoor Tracking by RFID Fusion with IMU Data", içinde *Asian Journal of Control*, Tem. 2019, c. 21, sy 4, ss. 1768-1777. doi: 10.1002/asjc.1954.

[81] H. W. Yoo *vd.*, "MEMS-based lidar for autonomous driving", *Elektrotechnik und Informationstechnik*, c. 135, sy 6, ss. 408-415, Eki. 2018, doi: 10.1007/s00502-018-0635-2.

[82] E. Shang, X. An, T. Wu, T. Hu, Q. Yuan, ve H. He, "LiDAR Based Negative Obstacle Detection for Field Autonomous Land Vehicles", *J Field Robot*, c. 33, sy 5, ss. 591-617, Ağu. 2016, doi: 10.1002/rob.21609.

[83] A. Aboutaleb, A. S. El-Wakeel, H. Elghamrawy, ve A. Noureldin, "LiDAR/RISS/GNSS dynamic integration for land vehicle robust positioning in challenging GNSS environments", *Remote Sens (Basel)*, c. 12, sy 14, Tem. 2020, doi: 10.3390/rs12142323.

[84] G. Reina, J. Underwood, G. Brooker, ve H. Durrant-Whyte, "Radar-based perception for autonomous outdoor vehicles", *J Field Robot*, c. 28, sy 6, ss. 894-913, Kas. 2011, doi: 10.1002/rob.20393.

[85] A. M. C. Rezende *vd.*, "An integrated solution for an autonomous drone racing in indoor environments", *Intell Serv Robot*, c. 14, sy 5, ss. 641-661, Kas. 2021, doi: 10.1007/s11370-021-00385-4.

[86] Y. Bi, M. Lan, J. Li, S. Lai, ve B. M. Chen, "A lightweight autonomous MAV for indoor search and rescue", içinde *Asian Journal of Control*, Tem. 2019, c. 21, sy 4, ss. 1732-1744. doi: 10.1002/asjc.2162.

[87] V. Mandal ve Y. Adu-Gyamfi, "Object detection and tracking algorithms for vehicle counting: a comparative analysis", *Journal of big data analytics in transportation*, c. 2, sy 3, ss. 251-261, 2020.

[88] O. Boutalbi, K. Benmahammed, K. Henni, ve B. Boukezata, "A high-performance control algorithm based on a curvature-dependent decoupled planning approach and flatness concepts for non-holonomic mobile robots", *Intell Serv Robot*, c. 12, sy 2, ss. 181-196, Nis. 2019, doi: 10.1007/s11370-018-00270-7.

[89] J. Woo ve N. Kim, "Collision avoidance for an unmanned surface vehicle using deep reinforcement learning", *Ocean Engineering*, c. 199, Mar. 2020, doi: 10.1016/j.oceaneng.2020.107001.

[90] D. Han, Q. Liu, ve W. Fan, "A new image classification method using CNN transfer learning and web data augmentation", *Expert Syst Appl*, c. 95, ss. 43-56, Nis. 2018, doi: 10.1016/j.eswa.2017.11.028.

[91] T. Y. Goh, S. N. Basah, H. Yazid, M. J. Aziz Safar, ve F. S. Ahmad Saad, "Performance analysis of image thresholding: Otsu technique", *Measurement (Lond)*, c. 114, sy March 2017, ss. 298-307, 2018, doi: 10.1016/j.measurement.2017.09.052.

[92] J.F.~Canny, "A Computational Approach To Edge Detection", *IEEE Trans Pattern Anal Mach Intell*, c. 8, sy 6, ss. 679-714, 1986.

[93] S. Ioffe ve C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift”, *32nd International Conference on Machine Learning, ICML 2015*, c. 1, ss. 448-456, 2015.

[94] “AnyLogic”, *AnyLogic Simülasyon Yazılımı*. https://www.anylogic.com/ (erişim 05 Ocak 2023).

[95] A. ben Khalifa, I. Alouani, M. A. Mahjoub, ve N. E. ben Amara, “Pedestrian detection using a moving camera: A novel framework for foreground detection”, *Cogn Syst Res*, c. 60, ss. 77-96, 2020, doi: 10.1016/j.cogsys.2019.12.003.

ÖZGEÇMİŞ

Ad-Soyad : Mustafa ERGİNLİ

ÖĞRENİM DURUMU:

- **Lisans** **:** 1993, Yıldız Üniversitesi, Kocaeli Müh.Fak, Endüstri
- **Yükseklisans** **:** 2006, Kütahya Dumlupınar Üniversitesi, Endüstri, Endüstri

MESLEKİ DENEYİM VE ÖDÜLLER:

TEZDEN TÜRETİLEN ESERLER:

- Erginli, Mustafa, and Ibrahim Cil. 2022. "Deep-Learning-Based Floor Path Model for Route Tracking of Autonomous Vehicles" Systems 10, no. 3: 83. https://doi.org/10.3390/systems10030083

DİĞER ESERLER: