

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

PROSEDÜREL EL HAREKETLERİNİN BİLGİSAYARLI GÖRMEYE
DAYALI YORUMLANMASI

DOKTORA TEZİ

Rüstem ÖZAKAR

OCAK 2024
TRABZON



KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

**PROSEDÜREL EL HAREKETLERİNİN BİLGİSAYARLI GÖRMEYE
DAYALI YORUMLANMASI**

Rüstem ÖZAKAR

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünde
"DOKTOR (BİLGİSAYAR MÜHENDİSİ)"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 06 / 12 / 2023

Tezin Savunma Tarihi : 19 / 01 / 2024

Tez Danışmanı : Dr. Öğr. Üyesi Eyüp GEDİKLİ

Trabzon 2024

ÖNSÖZ

“Prosedürel El Hareketlerinin Bilgisayarlı Görmeye Dayalı Yorumlanması” isimli bu tez Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Doktora Programı’nda hazırlanmıştır.

Tez çalışmam boyunca hiçbir desteğini esirgemeyen ve bilim insanı olma yolundaki ilk adımlarımı atarken kıymetli bilgileriyle ve tecrübesiyle yolumu aydınlatan danışman hocam Sayın Dr. Öğr. Üyesi Eyüp GEDİKLİ’ye teşekkürü bir borç bilirim. Ayrıca, yapıcı eleştirileri ve önerileri ile tezime büyük katkıları bulunan saygıdeğer hocalarım Sayın Dr. Öğr. Üyesi Murat AYKUT’a ve Sayın Dr. Öğr. Üyesi Emin TUĞCU’ya, eğitim öğretim hayatımda katkısı olan tüm hocalarıma teşekkür ederim.

Bu tez çalışmasının bundan sonraki çalışmalara katkı sağlamasını temenni ederim.

Rüstem ÖZAKAR

Trabzon 2024

TEZ ETİK BEYANNAMESİ

Doktora Tezi olarak sunduğum “Prosedürel El Hareketlerinin Bilgisayarlı Görmeye Dayalı Yorumlanması” başlıklı bu çalışmayı baştan sona kadar danışmanım Dr. Öğr. Üyesi Eyüp GEDİKLİ’nin sorumluluğunda tamamladığımı, verileri kendim topladığımı, analizleri kendim yaptığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 19/01/2024

Rüstem ÖZAKAR

İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ.....	III
TEZ ETİK BEYANNAMESİ.....	IV
İÇİNDEKİLER.....	V
ÖZET.....	VII
ŞEKİLLER DİZİNİ.....	IX
TABLolar DİZİNİ.....	XIV
SEMBOLLER VE KISALTMALAR DİZİNİ.....	XV
1. GENEL BİLGİLER.....	16
1.1. Bilgisayarlı Görme ve Makine Öğrenmesi.....	16
1.2. Amaç ve Problem Tanımlaması.....	21
1.3. Literatür Taraması.....	23
1.3.1. Görüntü Bazlı El Tanıma Çalışmaları.....	23
1.3.2. Görüntü Bazlı El Hareketi Tanıma Çalışmaları.....	26
1.3.3. Sensör Bazlı El Hareketi Tanıma Çalışmaları.....	32
1.3.4. Görüntü Bazlı El Yıkama Hareketi Tanıma Çalışmaları.....	32
1.3.5. Sensör Bazlı El Yıkama Hareketi Tanıma Çalışmaları.....	33
1.3.6. El Hareketi Tanımadaki Kullanılan Kamera Türleri.....	34
1.3.7. Literatürden Çıkarılan Sonuçlar.....	36
1.4. Görüntü İşleme ile İlgili Temel Bilgiler.....	36
1.4.1. Arka Plan Çıkarımı.....	36
1.4.2. Eşikleme.....	38
1.4.3. Maskeleye.....	40
1.4.4. Morfolojik Operatörler.....	41
1.4.5. Bağlı Bileşenler Analizi.....	42
1.4.6. Şablon Eşleştirme.....	43
1.4.7. Viola-Jones Algoritması.....	44
1.4.8. Hausdorff Mesafesi.....	45
1.4.9. Öznitelik Çıkarma Algoritmaları.....	45
1.5. Bilgisayarlı Görmede Yapay Sinir Ağları ve Değerlendirme Metrikleri.....	49
2. YAPILAN ÇALIŞMALAR.....	64

2.1. Kullanılan Yazılım Araçları	65
2.2. El Tanıma Çalışmaları.....	67
2.2.1. Arka Plan Çıkarımı ile El Tanıma.....	67
2.2.2. Deri Rengine Dayalı El Tanıma.....	68
2.2.3. Şablon Eşleştirmeye Dayalı El Tanıma	69
2.2.4. Viola-Jones Algoritması ile El Tanıma.....	69
2.2.5. Hausdorff Mesafesi ile El Tanıma	70
2.2.6. Öznitelik Çıkarma Algoritmaları ile El Tanıma	71
2.2.7. Yapay Sinir Ağlarıyla El Tanıma.....	71
2.2.8. Kinect 2 Kamerası ile El Tanıma.....	72
2.3. El Yıkama Hareketi Tanıma İçin İzlenen Yöntemler.....	74
2.3.1. Tanımlanan El Hareketleri	76
2.3.2. Kamera Görüntüleri Veri Seti	77
2.3.3. Sentetik Veri Seti	85
2.3.4. Elde Edilen Görüntülerin Niteliği ve Kategorizasyonu	89
2.3.5. Derinlik Görüntüleriyle El Tanıma	92
2.3.6. Kamera Veri Setinde Sınıflandırma.....	94
2.3.7. Sentetik Veri Setinde Sınıflandırma	99
2.3.8. Ağların Eğitimi	100
3. BULGULAR VE TARTIŞMA.....	103
3.1. Kamera Veri Seti ile Sınıflandırma Sonuçları.....	103
3.2. Sentetik Veri Seti ile Sınıflandırma Sonuçları	120
3.2.1. Inception-V3 ile Sabun Miktarı Sınıflandırma	122
3.2.2. YOLO-8n ile Sabun Miktarı Sınıflandırma	123
3.2.3. Inception-V3 ile Hareket Sınıflandırma.....	127
3.2.4. YOLO-8n ile Hareket Sınıflandırma.....	129
3.2.5. Hareket Sınıflandırma Modellerinin Değerlendirme Metrikleri	132
3.2.6. Inception-V3 ile Sabun Miktarı ve Hareket Sınıflandırma	133
3.2.7. YOLO-8n ile Sabun Miktarı ve Hareket Sınıflandırma.....	135
3.3. Literatür ile Karşılaştırma	138
4. SONUÇLAR.....	140
5. ÖNERİLER.....	142
6. KAYNAKLAR	143
ÖZGEÇMİŞ	

Doktora Tezi

ÖZET

PROSEDÜREL EL HAREKETLERİNİN BİLGİSAYARLI GÖRMEYE DAYALI YORUMLANMASI

Rüstem ÖZAKAR

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Eyüp GEDİKLİ
2024, 155 Sayfa

Dünya Sağlık Örgütü (DSÖ), enfeksiyonlara önlem olarak el hijyeni için belirli süreler boyunca uygulanan çeşitli hareketler içeren bir el yıkama prosedürü önermektedir. Özellikle sağlık alanında, bu prosedüre göre el yıkama işlemi, yetkili kişilerce aralıklı bir şekilde denetlenmektedir. Bu çalışmada, DSÖ'nün önerdiği el yıkama hareketlerine uygun olarak belirlenen 8 yıkama hareketinin ve yıkama sürelerinin asgari düzeyde uygulanmasını otonom denetleyebilmek için, bilgisayarlı görmeye dayalı bir sistem geliştirilmiştir. Çalışmada, kameradan elde edilen renkli (RGB), derinlik kamera görüntüleri ve render görüntülerinden elde edilen sentetik verilerle 2 özgün veri seti oluşturulmuştur. Veri setlerinden, renkli görüntüler, projeksiyon görüntüleri, noktalar kümesi ve birleştirilmiş noktalar kümesi örüntüleri üretilmiştir. El yıkama hareketleri, bu örüntülerden çerçeve bazlı olarak, zamansal boyut da içeren 16 farklı yapay sinir ağı modeli ile sınıflandırılmıştır. Ayrıca sentetik veriler üzerinde sabun miktarına göre 4 ve 32 sınıf belirlenerek sınıflandırma yapılmıştır. Önerilen sistemde, kameradan alınan gerçek zamanlı görüntüleri işlemek için çoklu iş parçacığı yaklaşımı kullanılarak maliyet azaltılmıştır. Sınıflandırmalarda kamera veri setinde %100, sentetik veri setinde %56,5 ve %48,5 doğruluk oranları elde edilmiştir.

Anahtar Kelimeler: DSÖ el yıkama prosedürü, Bilgisayarlı görme, El hareketi tanıma, El hijyeni, Nokta kümesi tabanlı veri modelleri, RGB-D, Kinect sensör, Örüntü tanıma

PhD. Thesis

SUMMARY

INTERPRETATION OF PROCEDURAL HAND GESTURES USING COMPUTER VISION

Rüstem ÖZAKAR

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Computer Engineering Graduate Program
Supervisor: Assistant Prof. Dr. Eyüp GEDİKLİ
2024, 155 Pages

World Health Organization (WHO) recommends a hand washing procedure for hand hygiene against infections consisting of various hand gestures performed for certain periods of time. Especially in the health sector, hand washing according to this procedure is inspected by authorized personnel periodically. In this work, a computer vision-based system was developed to inspect performing of 8 hand washing gestures derived from WHO's recommendations and their washing times autonomously. Two new datasets were created using color, depth data from camera and synthetic data from rendered images. Color (RGB) images, projection images from depth data, point cloud and merged point cloud data models were obtained from datasets. With these frame-based data models, hand washing gestures were classified using 16 different neural network models, including models which are spatio-temporal. Also, on synthetic data, classification with 4 and 32 classes according to soap amount was performed. The proposed system uses multi thread-based pre-processing for camera images in real time for efficiency. In classifications, %100 accuracy was achieved with camera dataset, %56,5 and %48,5 accuracies were achieved with render dataset.

Keywords: WHO hand washing procedure, Computer vision, Hand gesture recognition, Hand hygiene, Point cloud based data models, RGB-D, Kinect sensor, Pattern recognition

ŞEKİLLER DİZİNİ

	<u>Sayfa No</u>
Şekil 1. OpenCV’de Gaussian Karışım Modelleri kullanılarak yapılan arka plan çıkarma örneği (URL-6, 2023)	37
Şekil 2. Bir görüntünün RGB, HSV ve YCbCr renk uzaylarında üç boyutlu histogramları	39
Şekil 3. 0-255 değerleri arasında bir görüntünün Kırmızı, Yeşil, Mavi kanallar ve Yoğunluk (Intensity) piksel miktarlarını gösteren bir histogram örneği	39
Şekil 4. Gri düzey bir görüntü (sol üst) ve eşikleme sonrası oluşan ikili görüntüler (sağda)	40
Şekil 5. Orijinal görüntü (solda), ikili görüntü (ortada), maskelenmiş görüntü (sağda)	41
Şekil 6. Morfolojik operatörlerin ikili görüntülerde oluşturduğu etkiye örnekler. Genleşme (dilation) (üstte), aşındırma (erosion) (altta)	42
Şekil 7. Bağlı Bileşenler Analizinin ikili görüntülerdeki algoritma aşamaları. İlk etiketleme (solda) ve ikinci geçişten sonra (sağda) tespit edilen cisimler	43
Şekil 8. Normal görüntü (solda), integral görüntü (sağda)	44
Şekil 9. Viola-Jones algoritmasında öznitelik çıkarmak için kullanılan çekirdeklerden örnekler (Viola & Jones, 2001)	45
Şekil 10. SIFT ile anahtar nokta tespiti	46
Şekil 11. FAST ile belirlenen köşe örneği (Rosten & Drummond, 2006)	47
Şekil 12. ORB ile anahtar nokta tespiti	47
Şekil 13. BRISK ile anahtar nokta tespiti	48
Şekil 14. HOG ile elde edilen özniteliklerin örneği	49
Şekil 15. Perceptron yapısı: Girdiler (x), ağırlıklar (w), aktivasyon fonksiyonu (o)	50
Şekil 16. Çok katmanlı perceptron ağı	51
Şekil 17. Konvolüsyon işlemi: Solda orijinal görüntü, ortada konvolüsyon çekirdeği, sağda üretilen sonuç	53
Şekil 18. Aktivasyon fonksiyonu örnekleri	54
Şekil 19. Inception-V3 ağında kullanılan konvolüsyon filtreleri (Szegedy vd., 2016)	56
Şekil 20. MobileNet-V1 ağında kullanılan depthwise (üstte) ve pointwise (altta) konvolüsyon filtreleri (Howard vd., 2017)	56

Şekil 21.	ResNet-34 (solda) ve ResNet-50 (sağda) ağları için residual fonksiyonların yapısı (He vd., 2016)	57
Şekil 22.	PointNet ağında bulunan modüllerin yapısı (Qi vd. 2017).....	57
Şekil 23.	LSTM hücre yapısı	59
Şekil 24.	R-CNN çalışma prensibi (Girshick vd., 2014)	60
Şekil 25.	SSD ağının yapısı (Liu vd., 2016)	60
Şekil 26.	SSD ağının farklı boyutlarda öznetelik arama ve çevreleyen kutu mekanizması (Liu vd., 2016)	61
Şekil 27.	Çok sınıflı sınıflandırmalarda doğru negatif (DN), yanlış negatif (YN), doğru pozitif (DP), yanlış pozitif (YP) bölgelerinin hata matrisindeki temsili	63
Şekil 28.	Çerçeve farkı ile arka plan çıkarımı örneği	67
Şekil 29.	MOG2 yöntemi ile arka plan çıkarımı.....	67
Şekil 30.	Deri rengi aralığıyla el tanıma	68
Şekil 31.	Şablon eşleştirme örneği.....	69
Şekil 32.	Haar cascade ile avuç bölgesi belirleme	70
Şekil 33.	Kayan pencere yöntemiyle Hausdorff mesafesi ile el tespiti.....	70
Şekil 34.	Öznetelik çıkarma algoritmalarıyla gri düzey görüntülerde el tanıma. Soldan sağa; BRISK, ORB, FAST, AKAZE.....	71
Şekil 35.	EgoHands (Bambach vd., 2015) veri seti ile eğitilmiş SSD ağıyla el tanıma örneği	72
Şekil 36.	Kinect 2 kamerası (URL-22, 2023)	72
Şekil 37.	Time-of-Flight kameralarının çalışma prensibi	73
Şekil 38.	Kinect 2 kamerasının iskelet takibi sisteminde kullandığı eklem noktaları	73
Şekil 39.	Kinect 2 ile tespit edilen avuç bölgesi yörüngesinin OpenGL ile çizdirilmesi (solda), Kinect 2 ile el ve parmak tespiti (sağda)	74
Şekil 40.	DSÖ önerilerine uygun olarak belirlenen sekiz el yıkama hareketi	76
Şekil 41.	Veri setinde kullanılan lavaboların arka plan görüntüleri. Derinlik görüntüleri (solda), RGB görüntüleri (ortada), renkli-derinlik eşleştirilmiş görüntü (sağda).....	78
Şekil 42.	Gerçek zamanlı el bölgesi tespiti ve veri seti oluşturma sisteminin genel çalışma şekli.....	80
Şekil 43.	Sekiz yıkama hareketi için RGB görüntülerinden elde edilen el bölgesi örnekleri (ROI) (üstte), derinlik görüntülerinden elde edilen el bölgesi örnekleri (altta). Soldan sağa hareketler: 1-8.....	83
Şekil 44.	Sekiz yıkama hareketi için projeksiyon görüntülerinden örnekler. Soldan sağa hareketler: 1-8.....	83

Şekil 45.	Sekiz yıkama hareketi için, noktalar kümesi tipinde elde edilen verilerden örnekler. Üst sıra soldan sağa: hareketler 1-4. Alt sıra soldan sağa: hareketler 5-8.....	84
Şekil 46.	Sekiz yıkama hareketi için, birleştirilmiş noktalar kümesi tipinde elde edilen verilerden örnekler. Üst sıra soldan sağa: hareketler 1-4. Alt sıra soldan sağa: hareketler 5-8	84
Şekil 47.	Modellenen dört farklı lavabonun genel görünüşü.....	87
Şekil 48.	Lavabolara yerleştirilen kameraların ürettiği render görüntüleri.....	87
Şekil 49.	Makehuman ile modellenip çeşitli modifikasyon yapılan dört farklı karakter ve Blender'daki iskelet sistemleri.....	88
Şekil 50.	Blender'da Node sistemi ile bir karakterin deri dokusunun nitelikleri.....	89
Şekil 51.	Pseudo-Blur işleminin aşamalarından çerçeveler. (a) 4. adım sonrası oluşan görüntü. (b) 5. adım sonrası oluşan görüntü. (c) 6. adım sonrası oluşan görüntü. (d) Gaussian gürültüsü. (e) Görüntünün orijinali. (f) Pseudo-Blur işleminden sonra oluşan görüntü	91
Şekil 52.	Sekiz yıkama hareketinin animasyonlarından örnek çerçeveler. Soldan sağa: hareket 1-8	91
Şekil 53.	Bir yıkama hareketinin dört farklı sabun seviyesiyle animasyonundan çerçeveler. Soldan sağa, sabunsuzdan çok sabunluya	91
Şekil 54.	Derinlik görüntülerinin ön işleminden geçirilmesi.....	94
Şekil 55.	Inception-V3 ağının yapısı	95
Şekil 56.	MobileNet-V1 ağının yapısı	95
Şekil 57.	ResNet-50 ağının yapısı.....	95
Şekil 58.	PointNet ağının yapısı.....	96
Şekil 59.	PointNet – LSTM ağının yapısı.....	96
Şekil 60.	VGG-16 – LSTM ağının yapısı	96
Şekil 61.	Öznitelik füzyonu için kullanılan üç farklı ağın yapısı	96
Şekil 62.	Kamera veri setinin eğitim – doğrulama – test dağılımı.....	101
Şekil 63.	Inception-V3 RGB çerçeve bazlı modelin eğitim grafikleri.....	104
Şekil 64.	Inception-V3 RGB çerçeve bazlı sınıflandırmanın hata matrisi.....	104
Şekil 65.	MobileNet-V1 RGB çerçeve bazlı modelin eğitim grafikleri	105
Şekil 66.	MobileNet-V1 RGB çerçeve bazlı sınıflandırmanın hata matrisi	105
Şekil 67.	ResNet-50 RGB çerçeve bazlı modelin eğitim grafikleri.....	106
Şekil 68.	ResNet-50 RGB çerçeve bazlı sınıflandırmanın hata matrisi.....	106
Şekil 69.	Inception-V3 projeksiyon çerçeve bazlı modelin eğitim grafikleri.....	107
Şekil 70.	Inception-V3 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi	107
Şekil 71.	MobileNet-V1 projeksiyon çerçeve bazlı modelin eğitim grafikleri.....	108

Şekil 72.	MobileNet-V1 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi.....	108
Şekil 73.	ResNet-50 projeksiyon çerçeve bazlı modelin eğitim grafikleri	109
Şekil 74.	ResNet-50 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi	109
Şekil 75.	PointNet çerçeve bazlı modelin eğitim grafikleri	110
Şekil 76.	PointNet çerçeve bazlı sınıflandırmanın hata matrisi	110
Şekil 77.	VGG-16 - LSTM RGB modelinin eğitim grafikleri.....	111
Şekil 78.	VGG-16 - LSTM RGB sınıflandırmanın hata matrisi.....	111
Şekil 79.	VGG-16 - LSTM projeksiyon modelinin eğitim grafikleri	112
Şekil 80.	VGG-16 - LSTM projeksiyon sınıflandırmanın hata matrisi	112
Şekil 81.	PointNet - LSTM modelinin eğitim grafikleri.....	113
Şekil 82.	PointNet - LSTM sınıflandırmanın hata matrisi	113
Şekil 83.	PointNet noktalar kümesi toplamı modelinin eğitim grafikleri.....	114
Şekil 84.	PointNet noktalar kümesi toplamı sınıflandırmasının hata matrisi	114
Şekil 85.	Öznitelik füzyonu modelinin eğitim grafikleri	115
Şekil 86.	Öznitelik füzyonu sınıflandırmasının hata matrisi	115
Şekil 87.	Kamera veri seti modelleri için Mikro-ROC eğrisi grafiği	117
Şekil 88.	Kamera veri seti modelleri için Makro-ROC eğrisi grafiği.....	118
Şekil 89.	PointNet-LSTM yeniden değerlendirme modellerinden birinci modelin eğitim grafikleri	119
Şekil 90.	PointNet-LSTM yeniden değerlendirme modellerinden ikinci modelin eğitim grafikleri	119
Şekil 91.	PointNet-LSTM ikinci değerlendirme sınıflandırmasının hata matrisi	120
Şekil 92.	Kamera veri setiyle eğitilmiş Inception-V3 ağının sentetik veri seti sınıflandırmasının hata matrisi	121
Şekil 93.	Inception-V3 4 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri	122
Şekil 94.	Inception-V3 4 sınıflı modelin sentetik veri seti üzerinde hata matrisi.....	123
Şekil 95.	Inception-V3 ağı 4 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)	123
Şekil 96.	YOLO-8n 4 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri	125
Şekil 97.	YOLO-8n 4 sınıflı modelin sentetik veri seti üzerindeki hata matrisi.....	125
Şekil 98.	YOLO-8n 4 sınıflı modelin eğitim verileri ve etiketlerinden örnekler.....	126
Şekil 99.	YOLO-8n ağının 4 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)	126
Şekil 100.	Inception-V3 8 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri	127

Şekil 101. Inception-V3 8 sınıflı modelin sentetik veri seti üzerindeki hata matrisi	128
Şekil 102. Inception-V3 8 sınıflı modelin kamera veri seti üzerindeki hata matrisi	128
Şekil 103. YOLO-8n 8 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikler,.....	129
Şekil 104. YOLO-8n 8 sınıflı modelin sentetik veri seti üzerindeki hata matrisi.....	130
Şekil 105. YOLO-8n 8 sınıflı modelin eğitim verileri ve etiketlerinden örnekler.....	130
Şekil 106. YOLO-8n ağının 8 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)	131
Şekil 107. YOLO-8n 8 sınıflı modelin kamera veri seti üzerindeki hata matrisi	131
Şekil 108. Inception-V3 sentetik veri seti hareket sınıflandırması alıcı işletim karakteristiği grafiği.....	132
Şekil 109. Yolo-8n sentetik veri seti hareket sınıflandırması alıcı işletim karakteristiği grafiği.....	133
Şekil 110. Inception-V3 32 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri	134
Şekil 111. Inception-V3 32 sınıflı modelin sentetik veri seti üzerinde hata matrisi.....	134
Şekil 112. Inception-V3 ağı 32 sınıflı modelinin sınıflandırmasından örnekler (birleşik görüntü)	135
Şekil 113. YOLO-8n 32 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri	136
Şekil 114. YOLO-8n 32 sınıflı modelin sentetik veri seti üzerinde hata matrisi.....	136
Şekil 115. YOLO-8n 32 sınıflı modelin eğitim verileri ve etiketlerinden örnekler.....	137
Şekil 116. YOLO-8n ağı 32 sınıflı modelinin sınıflandırmasından örnekler (birleşik görüntü)	137

TABLolar DİZİNİ

	<u>Sayfa No</u>
Tablo 1. Kamera veri setindeki çerçevelerin miktarı ve dağılımı.....	83
Tablo 2. Sentetik veri seti için oluşturulan modellerin veri tablosu	92
Tablo 3. Sınıflandırmada kullanılan ağlar ve parametreleri	102
Tablo 4. Kamera veri seti modellerinin doğruluk oranları.....	116
Tablo 5. Kamera veri seti modellerinin değerlendirme metrikleri.....	116
Tablo 6. Sentetik veri seti 8 sınıflı modellerin değerlendirme metrikleri	132
Tablo 7. Literatürde bulunan el yıkama tanıma çalışmalarının genel özeti	138

SEMBOLLER VE KISALTMALAR DİZİNİ

ANN	: Yapay Sinir Ağları (Artificial Neural Network)
API	: Uygulama Programlama Arayüzü (Application Programming Interface)
CNN	: Konvolüsyonel Sinir Ağı (Convolutional Neural Network)
CGI	: Bilgisayarda Üretilmiş Görüntü (Computer Generated Imagery)
CCA	: Bağlı Bileşenler Analizi (Connected Component Analysis)
HSV	: Renk Tonu Doygunluk Değer (Hue Saturation Value)
HMM	: Saklı Markov Modeli (Hidden Markov Model)
HOG	: Yönlü Gradyanlar Histogramı (Histogram of Oriented Gradients)
GPU	: Grafik İşleme Birimi (Graphics Processing Unit)
LDA	: Lineer Diskriminant Analizi (Linear Discriminant Analysis)
LSTM	: Uzun Kısa Süreli Bellek (Long Short Term Memory)
MOG	: Gaussian Karışım Modeli (Mixture of Gaussians)
PSO	: Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)
PCD	: Noktalar Kümesi (Point Cloud)
PCA	: Temel Bileşenler Analizi (Principal Component Analysis)
RGB	: Kırmızı Yeşil Mavi (Red Green Blue)
RNN	: Yinelemeli Sinir Ağı (Recurrent Neural Network)
ROI	: İlgi Bölgesi (Region of Interest)
RF	: Rastgele Orman (Random Forest)
ReLU	: Doğrultulmuş Lineer Birim (Rectified Linear Unit)
R-CNN	: Bölge Tabanlı Konvolüsyonel Sinir Ağı (Region-Based CNN)
ROC-AUC	: Alıcı İşletim Karakteristiği – Eğri Altı Alan (Receiver Operating Characteristic – Area Under Curve)
SSD	: Tek Atış Dedektörü (Single Shot Detector)
SVM	: Destek Vektör Makinesi (Support Vector Machine)
YOLO	: Sadece Bir Kez Bak (You Only Look Once)
k-NN	: k-En Yakın Komşu (k-Nearest Neighbours)
ToF	: Uçuş Süresi Sensörü (Time of Flight)

1. GENEL BİLGİLER

Son yarım yüzyılda bilgisayar bilimleri alanında yaşanan büyük gelişmeler, insanların çalışma anlayışlarını, iş yapma biçimlerini oldukça değiştirmiştir. Bu nedenle, teknolojik cihazları kontrol etmede sağlanan arayüzler de giderek gelişme göstermiş, sesle, vücut hareketleriyle, beyin dalgalarıyla yönetilebilen, kullanıcıya rahat ve kapsamlı kontrol sunan arayüzler geliştirilmiştir.

Her türlü cihaz dahil olmak üzere, nesnelerle etkileşim yoğunlukla eller sayesinde olmaktadır. Üç boyutlu uzayda nesneleri hareket ettirmek, dönüştürmek, iletişim kurmak için eller kullanılır. Gelişen teknolojinin sunduğu hemen hemen tüm cihazlar da el yordamıyla bir kontrol imkânı sunar. Hatta, bu alanda özel bir bilim dalı kurulmuş, Human-Computer Interaction (İnsan-Bilgisayar Etkileşimi) adı altında cihazların en etkili kontrolü nasıl yapılır sorusuna bilimsel araştırmalarla cevap aranmıştır.

El tanıma ve el hareketleri tanıma, bu gelişmelerin sonucunda yoğun bir araştırma alanı haline gelmiş, herhangi bir fiziksel temas olmadan, sadece el hareketleriyle etkili ve hatasız biçimde çeşitli etkileşimler yaratmak bu alanın temel hedefi olmuştur. Makine öğrenmesi yöntemlerinin gelişmesi sonucunda, el hareketi tanımanın birçok uygulaması hayata geçmiştir. İşaret dili hareketlerinin çözümlenmesi (Buehler vd., 2008), araç içi mekanizma ve ayarlarının kontrolü (Molchanov vd., 2015), oyun konsolları ve sanal gerçeklik uygulamaları el hareketi tanıma uygulamalarına örnekler olarak sayılabilir.

Doğal bir etkileşim sunan arayüzler günümüzde Doğal Kullanıcı Arayüzleri (Natural User Interfaces) adını almıştır. Yakın gelecekte, interaktif kullanım sunan hemen her cihazda (medikal, otomotiv, eğlence vb.) insan-bilgisayar etkileşiminin iki boyutlu düzlemde üç boyutlu düzleme taşınması, el hareketi tanıma alanındaki gelişmelere bağlı olacaktır.

1.1. Bilgisayarlı Görme ve Makine Öğrenmesi

Bilgisayarlı Görme kavramı (Computer Vision), kamera yardımıyla elde edilmiş görüntüler üzerinde uygulanan görüntü işleme ve makine öğrenmesi yöntemleri sayesinde, görüntüden anlamlı bilgiler çıkarmak, bu bilgilere dayalı olarak bir probleme çözüm olanakları geliştirmeyi tanımlamaktadır. Görüntü işleme ve makine öğrenmesi yöntemlerindeki gelişmeler ve işlemci gücünün artması gibi etmenler sayesinde, her geçen

gün daha karmaşık problemler bilgisayarlı görmenin konusu olmakta ve çözüm geliştirilmektedir.

Zeka, bilim tarihinde çokça tartışılan ve birçok tanımı olan bir kavramdır. Ortak tanımlardan biri şu şekildedir; Zeka; sebeplendirme, planlama, problem çözme, soyut düşünme, kompleks fikirleri anlama, hızlıca öğrenme ve tecrübeye dayalı öğrenmeyi içeren temel bir zihinsel kabiliyettir (Leg & Hutter, 2007). Çoklu zeka kuramını ortaya atan Amerikan psikolog Howard Gardner’a göre zeka; “Problem çözme, bir veya birden fazla kültürde değeri olan bir ürün üretebilme yeteneğidir” (Gardner, 2000).

Bilgisayar bilimleri alanında yapay zeka kavramıyla ilgilenen araştırmacıların zeka tanımlarından bazıları şu şekildedir (Leg & Hutter, 2007);

- “Başarının ölçeğinin sistemin ana hedefine ulaştıracak alt hedeflerin gerçekleştirilmesi olduğu önceden belirlenmemiş bir ortamda, başarı oranını artıracak davranışların seçilmesidir.” (J. S. Albus)
- “Bir dizi çevrede hedeflere ulaşmak için uyum sağlar davranış gösteren herhangi bir sisteme akıllı adı verilebilir.” (D. Fogel)
- “Kompleks çevrelerde kompleks hedeflere ulaşmaktır.” (B. Goertzel)
- “Akıllı sistemlerin çok farklı çevrelerde iyi çalışması beklenir. Çevre hakkında bilgileri yeterli olmasa bile zeka özellikleri başarı olasılıklarını maksimize eder. Akıllı sistemler çevreden ve hedefe ulaşma durumundan farklı değerlendirilemezler.” (R. R. Gudwin)
- “Zeka, bir sistemin kompleks bir çevrede başarılı performansdır (hedefe ulaşmasıdır).” (J. A. Horst)
- “Zeka, hedeflere ulaşmak için sınırlı kaynakları (zaman dahil) optimal olarak kullanma kabiliyetidir.” (R. Kurzweil)
- “Zeka, olası çok büyük bir arama uzayında hızlıca uygun bir çözüm bulma gücüdür.” (D. Lenat ve E. Feigenbaum)
- “Zeka, bir ajanın geniş bir çevre aralığında hedeflere ulaşma yeteneğini ölçer.” (S. Legg ve M. Hutter)
- “Geniş bir dizi görevde başarı gösterilmesi olarak tanımlanabilir.” (H. Masum)
- “Zeka, dünyada hedeflere ulaşma kabiliyetinin hesaba dayalı kısmıdır. İnsanlarda, hayvanlarda ve makinelerde çeşitli türde ve seviyede zeka görülür.” (J. McCarthy)
- “Zor problemleri çözme yeteneğidir.” (M. Minsky)

- “Zeka, kompleks bir çevrede bilgiyi etkili bir biçimde işleme yeteneğidir. Etkili olma kriterleri önceden belirlenmemiştir ve erişilebilir değildir. Bilgiyi işleme sonucunda elde edilirler.” (H. Nakashima)
- “Çevrenin doğacak taleplerine ve sistemin imkanlarına göre, belirli bir karmaşıklık ve hız sınırı içerisindeki uygun davranıştır.” (A. Newell ve H. A. Simon)
- “Akıllı bir ajan kendi koşullarına ve hedeflerine uygun davranışı gerçekleştirir, çevresel değişimlere ve hedeflere karşı esnektir, tecrübeden öğrenir, sınırlı hesaplama ve sınırlı algı çerçevesinde doğru seçimler yapar.” (D. Poole)
- “Zeka, zamanla daha iyiye gitmektir.” (Schank)
- “Zeka, bir bilgi işleme sisteminin, kısıtlı bilgi ve kaynaklarla çevreye uyum sağlama yeteneğidir.” (P. Wang)
- “Başarılı bir hayat sürebilmek için gereken mental kabiliyettir.” (K. Warwick)

Makinelerin (ve bilgisayarların) insan zekasına benzer davranışlar sergileyebilmesi noktasında geçtiğimiz yüzyılda büyük gelişmeler yaşanmıştır. Modern makine öğrenmesi yöntemlerinin geçmişi nispeten yakın da olsa, bu alanın temelleri 20. yüzyılın başlarına kadar gitmektedir. Bu alandaki birtakım önemli buluşlardan kronolojik olarak bahsetmek gerekirse; 1936 yılında, matematikçi, istatistikçi ve genetikçi Sir Ronald Aylmer Fisher tarafından, halen bilgisayar bilimleri dahil birçok alanda yaygınlıkla kullanılmakta olan Lineer Diskriminant Analizi (LDA) yönteminin temeli atılmıştır. Bu yöntem, ilk bölütleme (segmentation) algoritması olarak kabul edilebilir.

1950’de Alan Turing, daha sonra oldukça popüler olacak olan “Turing Testi” fikrini ortaya atmıştır (Turing, 2009). Bu teste göre, bir insan, karşısında bilgisayarla mı ya da bir insanla mı muhatap olduğunu bilmeden bir bilgisayarla etkileşim kuracak, etkileşimin sonunda kişi karşıdakinin insan olduğuna kanaat getirirse, bilgisayar Turing Testini geçmiş olacaktı.

1951 yılında Evelyn Fix ve Joesph Hodges tarafından k-En Yakın Komşu (k-Nearest Neighbours, k-NN) yönteminin temeli atılmıştır (Fix & Hodges, 1951). Bu yöntemle bir örneğin hangi sınıfa ait olduğu, diğer örneklerden komşularına olan mesafelerine bakarak karar veriliyordu.

Yapay sinir ağlarının temelleri, genel olarak Cornell Üniversitesinden psikolog Frank Rosenblatt’a dayandırılır. İnsan merkezi sinir sisteminden örnek alarak fikirlerini geliştiren Rosenblatt ve ekibi, 1950’li yılların sonunda alfabenin harflerini tanıyan bir sistem

geliştirmiştir. Bu makineye, hala makine öğrenmesi alanında kullanılan “Perceptron” adı verilmiştir (Rosenblatt, 1958).

1966 yılında, Marvin Minsky MIT üniversitesinde, asistan öğrencisine bir kamerayı bilgisayara bağlamasını ve bilgisayarın bu verileri anlamasını sağlamasını istemiştir (Szelisky, 2022).

1967 yılında MacQueen tarafından k-Means algoritması geliştirilmiştir (MacQueen, 1967). Yoğun bir kullanıma sahip olan bu algoritmada temel mantık, veri kümelerinin rastgele seçilen k adet merkeze olan uzaklıklarının hesaplanarak sınıflandırılmasıdır. Bu k adet merkez noktası, sınıflandırılan verilerin ortalaması alınarak yeniden hesaplanmakta ve değerlerinde değişim olmayana dek bu işlem tekrarlanmaktadır.

1974’te, basılı karakterleri tanınması için ilk Optik Karakter Okuma (Optical Character Recognition) cihazı geliştirilmiştir (URL-1, 2023).

1979 yılında Coleman ve Andrews tarafından k-Means algoritması ile görüntülerin bölütlemesi yöntemi önerilmiştir (Coleman & Andrews, 1979).

1981 yılında Horn ve Schunk tarafından geliştirilen optik akış yöntemi, bilgisayarlı görme alanında sıklıkla kullanılan temel bir başka yöntemdir (Horn & Schunk, 1981). Bu yöntemde, hareket eden cisimlerin piksellerinin parlaklık değerlerinin benzer olacağı varsayımıyla yola çıkılmış, pikseller arasındaki konumsal ve zamansal değişim bir vektör olarak formüle edilmiştir.

Gabor filtreleri, adını Dennis Gabor’dan almış özel bir fonksiyondur (Gabor, 1946). Daugman tarafından yapılan çalışmada (Daugman, 1985), memelilerin görüntü hücrelerinin bu fonksiyonlarla modellenebileceği gösterilmiştir. Zamanla görüntü işleme alanında temel bir yöntem haline gelen Gabor filtreleri, kenar bulma, köşe tespiti, yuvarlak (blob) tespiti için yoğun olarak kullanılmaktadır.

1986 yılında Rummelhart vd. tarafından zamansal boyuta sahip veriler üzerinden öğrenme yapabilen bir yapay sinir ağları modeli geliştirilmiştir (Rummelhart vd., 1986). Yinelemeli Yapay Sinir Ağları (Recurrent Neural Networks) adı verilen bu ağ modeli, daha sonra Sepp Hochreiter ve Jürgen Schmidhuber tarafından ilerletilerek 1997 yılında, kendi içinde hafıza barındırabilen hücre yapısı sayesinde daha uzun zamansal veriler üzerinden öğrenme yapabilen Long-Short Term Memory (LSTM) ağı olarak sunulmuştur (Hochreiter & Schmidhuber, 1997).

1986 yılında John F. Canny tarafından, görüntüler üzerinde etkili bir kenar bulma algoritması geliştirilmiştir (Canny, 1986). Bu algoritmanın temelinde, filtreden geçirilen bir

görüntüdeki piksel değerlerinin x ve y yönündeki değişimlerinin birinci türevi yardımıyla hesaplanan gradyanlar bulunmaktadır.

1986 yılında, Rina Dechter ilk kez Derin Öğrenme (Deep Learning) tanımını kullanmıştır (Dechter, 1986). LeCun ve arkadaşlarının yaptığı çalışmayla beraber (LeCun vd., 2015), bugün de yapay zekada en çok kullanılan mimarilerin temeli olan Derin Sinir Ağları (Deep Neural Networks) popülerleşmiştir.

Robert Schapire tarafından 1990 yılında, Boosting olarak kullanılmaya başlanacak olan “Zayıf Öğreniciler” (Weak Learning) yöntemi geliştirilmiştir (Schapire, 1990). Bu yöntemdeki temel mantık, birçok zayıf öğrenicinin bir araya geldiğinde güçlü bir sınıflandırıcı oluşturabileceği fikrine dayanıyordu.

Turk ve Pentland tarafından (Turk & Pentland, 1991), 1991 yılında öz değerler kullanılarak (Eigenfaces) yüz tanıma sistemi geliştirilmiştir. Yüz görüntülerinden elde edilen vektör üzerinde Temel Bileşenler Analizi (Principal Component Analysis, PCA) kullanılarak, bir yüze ait temel hatlar belirlenmekteydi.

Cortes ve Vapnik tarafından 1995’te, bugün makine öğrenmesinde Destek Vektör Makineleri (Support Vector Machine, SVM) olarak yaygınlıkla kullanılmakta olan yöntem geliştirilmiştir (Cortes & Vapnik, 1995). Bu yöntem, farklı çekirdek fonksiyonları sayesinde doğrusal yöntemle ayrıştırılamayan veriler için de kullanılabilir.

2001 yılında Viola ve Jones tarafından gerçek zamanlı bir yüz tespit algoritması geliştirilmiştir (Viola & Jones, 2001). Yüz tanıma için özel tasarlanmış bazı çekirdekler görüntü üzerinde dolaştırılarak yüz öznitelikleri (feature) bulunmuştur. Hesaplamayı kolaylaştırmak için, her bir pikselin kendinden solda ve üstte kalan piksellerin değerlerinin toplamı olarak ifade edildiği integral görüntü yöntemi sunulmuştur. Özniteliklerin en etkili olanları boosting algoritmaları yardımıyla seçilmiştir.

Yönlendirilmiş Gradyanların Histogramı (Histogram of Oriented Gradients, HOG) yöntemi, ilk olarak 1986 yılında konsept olarak McConnell tarafından ortaya konmuşsa da (McConnell, 1986), 2005 yılında Dalal ve Triggs tarafından insan tanıma yöntemi olarak sunulmuş ve kısa sürede temel görüntü işleme yöntemlerinden biri olmuştur (Dalal & Triggs, 2005).

Alex Krizhevsky ve arkadaşları tarafından geliştirilen AlexNet (Krizhevsky vd. 2017), çok katmanlı yapay sinir ağlarının ekran kartları kullanılarak eğitilmesine öncülük eden çalışmalardan biri olmuştur. 2012 yılında ImageNet (Deng vd., 2009) veri seti üzerinde

yüksek başarı sağlayan ağı, bugün hala yoğun olarak kullanılan konvolüsyonel sinir ağlarının (Convolutional Neural Network) önemli örneklerinden biri olarak sayılmaktadır.

2014'te Ian Goodfellow ve arkadaşları tarafından (Goodfellow vd., 2014) Çekişmeli Üretici Ağlar (Generational Adversarial Network) geliştirilmiştir. Bu ağların yapısı sayesinde yapay zekanın yeni görüntüler üretmesi mümkün olmuştur. Son yıllarda bir başka teknik olarak otokodlayıcıların (autoencoder) derin ağların içerisinde kullanılmasıyla beraber üretici (generative) yapay zeka modelleri oldukça başarılı ve popüler olmaya başlamıştır.

2018'li yıllardan itibaren çok büyük metin verileriyle eğitilen ve çok fazla parametreye sahip olan Büyük Dil Modelleri (Large Language Model) yaygınlaşmaya başlamıştır. Yazıları anlamlandırıp insana benzer şekilde cevaplar verebilen bu modellerin başarısı, yapay zekanın insana benzer şekilde düşünüp düşünemeyeceği tartışmalarını başlatmış ve genel bir yapay zekanın (artificial general intelligence) yakın gelecekte mümkün olabileceği kanısını oluşturmuştur.

1.2. Amaç ve Problem Tanımlaması

Hareket tanıma, görüntü işleme ve makine öğrenmesi alanlarının yoğun çalışılan konularından biridir. Bu tanıma, herhangi bir cismin hareketi üzerine yoğunlaşabileceği gibi, insana has mimik ve hareketler üzerinde de kullanılabilir. Literatürde yaygın olarak çalışılan çeşitli hareket tanımlar, el hareketi tanıma, yüz hareketi tanıma, vücut hareketi tanıma başlıklarında ele alınabilir (Mitra & Acharya, 2007).

Bir hareket tanıma hareketin niteliğine göre ikiye ayrılabilir; dinamik hareket (dynamic gesture) ve statik hareket (static gesture). Bazı hareketler bu ikisinin kombinasyonunu içerebilir. Statik hareket tanımada, genelde hareketsiz bir görüntü üzerindeki hareketler algılanmaktadır. Dinamik hareket tanımada ise, uzay-zaman düzleminde gerçekleşen hareketli görüntüler üzerinden tanıma yapılmaktadır. Burada hareketlerin başlangıç ve bitiş noktaları, hareket süresi, hareket sıralamaları önem taşımaktadır.

Hareket tanıma, hareket tanınan yöntem üzerinden de ikiye ayrılabilir; temas veya sensör bazlı (contact / sensor based) ve görüntü bazlı (vision based). Sensör bazlı yöntemde kullanıcının bir tür cihazla beraber ilgili hareketi gerçekleştirmesi gerekir. Bu cihaz ayırt edici bir renge sahip giyilebilir bir nesne olabileceği gibi, sensörlere sahip bir elektronik de

olabilir. Görüntü bazlı tanıma yönteminde ise bir veya birden fazla kameranın aldığı görüntülerdeki hareketin çeşitli görüntü işleme ve makine öğrenmesi prensipleriyle yorumlanmasına çalışılır.

Bu tezde, dinamik el hareketi tanıma konusu üzerinde çalışılmıştır. Tezin amacı, bir prosedüre bağlanmış el yıkama hareketlerinin algılanması ve tanınmasıdır. Bu sayede özellikle el hijyeninin olmazsa olmaz olduğu alanlarda (sağlık, gıda sektörü gibi) hijyen takibi otomatik olarak yapılabilir. Günümüzde bunun denetlemesi, özellikle sağlık sektöründe yoğun bakım alanlarında insan gücü ile yapılmaktadır. Mevcut teknoloji ile bu denetleme otomatik bir sistem haline getirilebilir.

Son yıllarda salgın hastalıkların artmasıyla birlikte, çeşitli mikropların insan hayatını ne denli etkilediği herkesçe tecrübe edilmiştir. Kişisel hijyenin hiç olmadığı kadar önem kazandığı bu dönemlerde, enfeksiyon riski en yoğun alanlardan biri de hastanelerdir. Nozokomiyal enfeksiyon, yani hastaneden kapılan enfeksiyon, en önemli hasta ölümleri arasında gelmektedir. Yapılan bir çalışmada (Zhou vd., 2020), Covid-19 pandemisi sırasında, nozokomiyal enfeksiyonların oranı %44 olarak belirlenmiştir. Yine bir başka çalışmada, hastane enfeksiyonlarının yayılmasının başlıca sebepleri hemşireler ve doktorlar olarak görülmektedir (Wiener-Well vd., 2011). Birden fazla hastayla ilgilenmek durumunda olan doktorlar ve hemşireler, çeşitli hijyen kurallarına uymakla yükümlüdürler. Özellikle yoğun bakımlarda bu kurallar hayati öneme sahiptir. Bu hijyen kurallarından biri de doğru bir biçimde el yıkaması gerçekleştirmektir. Bir hastanın bakımından diğerine geçtikleri sırada, birtakım özel prosedürlerle süre ve hareket olarak belirlenmiş bir el yıkaması yapmak zorundadırlar. Bu yıkama prosedürleri, Dünya Sağlık Örgütü tarafından belirlenmiş (URL-2, 2023) hareketlere uygun olarak gerçekleşmektedir.

Günümüzde el yıkama prosedürünü kontrol eden yaygın bir teknoloji ürünü bulunmamaktadır. Bu çalışmadaki amaç ise, bu el yıkama prosedürünü otomatikleştirmek ve teknoloji ile kontrol edilebilir hale getirmektir. Bu bağlamda, bilgisayar bilimleri alanında görüntü işleme ve el hareketi tanıma metotları kullanılarak, bir el yıkama prosedürü takip edici sistem tasarlanması tezin amaçları arasındadır. Tasarlanan bu sistem, el yıkama prosedürlerini takip ederek, hijyen gereken her yerde el yıkamanın belirlenen prosedüre uygun olup olmadığını belirleyecektir. Böylece, toplumun genel sağlığına ve hijyenine olumlu katkı sağlayacak, salgın hastalıkların önlenmesine yardımcı olacaktır.

1.3. Literatür Taraması

El tanıma ve el hareketi tanıma, üç asırdan fazla bir süredir literatürde yoğun olarak çalışılmakta ve sürekli geliştirilmekte olan bir konudur. Her probleme yönelik, yüzde yüz kesin çözüm sunan standart bir yöntem olmamakla birlikte, farklı durumlar için uygun düşen metotlar veya metot kombinasyonları denenmiş ve çözümler üretilmiştir. Neredeyse tüm görüntü işleme ve makine öğrenmesi metotları bu konuda kullanılmış ve başarıları ölçülmüştür. Geniş kapsamlı literatür taraması sonucu, bu alanda kullanılan metotlardan ve teknolojilerden kısaca bahsedilecektir. Literatür taraması, el tanıma, el hareketi tanıma ve el yıkama hareketlerini tanıma konusunda yapılmış çalışmalar olarak farklı başlıklarda incelenecektir.

1.3.1. Görüntü Bazlı El Tanıma Çalışmaları

El hareketi tanımak için gereken ilk adım, etkili bir biçimde görüntüde elin bulunduğu bölgeyi tespit etmektir. Sensör bazlı hareket tanımalarda bu aşamaya gerek kalmadan hareket tanıma yapılabilmektedir. Görüntü bazlı hareket tanımalarda bu aşama mecburidir.

İnsan elleri, bileklerle beraber toplamda 27 kemik ve 27 eklem bulunduran gelişmiş organlardır. Toplamda 27 serbestlik derecesi (degrees of freedom) içerdiği için, herhangi bir anda herhangi bir durumda birçok farklı pozisyon ve şekilde bulunabilir. Bu, el tanımayı zorlaştıran öğelerden birisidir. El tanımada bir diğer zorluk, deri rengi ve ışık yansımalarının çok çeşitli olabilmesidir (Mohamed vd., 2021). Bir başka zorluk da ellerin hareketten veya etkileşimden dolayı, cisimlerin arkasına düşmesi, kısmen veya tamamen görüntüden çıkması olabilir (Mohamed vd., 2021). Bu durum anlık veya bir süreliğine gerçekleşebilir. El tanıma için geliştirilecek yöntemlerin, yukarıda saydığımız zorlukları aşması gerekmektedir.

Bilgisayarlı görme bazlı bütün el hareketi tanıma çalışmaları, mutlaka el tanıma işlemini gerçekleştirmesi gerektiği için, bu çalışmalar incelendiğinde el tanıma için kullanılan yaygın yöntemler de incelenmiş olacaktır. Sadece el tanıma ve takibine yoğunlaşan bazı önemli çalışmalar şu şekilde sıralanabilir;

Black ve Jepson'ın yaptığı çalışmada (Black & Jepson, 1998), obje tanıma ve takibi için EigenTracking metodu geliştirilmiştir. Eigenspace gösterimi, büyük bir görüntü setini az sayıda ortogonal temel (orthogonal basis) görüntülerle temsil edebilmeyi sağlar. Bu yöntem en küçük kareler (least squares) metodu üzerine kurulmuş bir metottur. Girdi görüntülerine dönüşüm uygulayarak, bir objenin tüm açılardan çekilmiş olası tüm

görüntülerini kullanmak yerine daha az sayıda görüntüyle işlem yapılabilir. Eigen uzayında tüm olasılıkları temsil yerine, belirli obje temsilleri kullanarak, görüntülerde bunları aramak üzerine kurulu bir yöntem önerilmiştir. Her bir görüntü için tek boyutlu bir vektör oluşturulup, bunlardan oluşan bir matris de Tekil Değer Ayrışması (Singular Value Decomposition, SVD) yöntemiyle özvektörlere ayrılmıştır. Bu yöntemle de el hareketlerini tanımayı gerçekleştirmişlerdir.

MacCormick ve Isard'ın yaptığı çalışmada (MacCormick & Isard, 2000), parçacık filtreleri bölünmüş örnekleme (partitioned sampling) yöntemi ile kullanılarak el takibi yapılmıştır. Başparmak, işaret parmağı ve yumruk şeklindeki el pozu, b-spline eğrisi olarak temsil edilmiş ve kamera görüntülerinden el bölgesi kontörleri ile el tanıma yapılmıştır.

Stenger ve arkadaşlarının yaptığı çalışmada (Stenger vd., 2001), Kokusuz Kalman Filtresi'yle (Unscented Kalman Filter) görüntülerdeki el bölgelerinin kontör bilgisi elde edilmiş, kuadrik (quadric) geometrik parçalardan oluşan bir üç boyutlu el modeli kullanılarak elin pozu belirlenmeye çalışılmıştır.

Bretzner ve arkadaşlarının yaptığı çalışmada (Bretzner vd., 2002), görüntü IUV renk uzayına çevrildikten sonra, her bir renk kanalı için Gaussian çekirdekleri kullanılarak boyut-uzay (scale-space) temsilleri hesaplanmıştır. Bu boyut-uzayın içinde lokal maksimumlardan hareketle damlalar (blob) oluşturulmuştur. Hareket takibi için parçacık filtresi (particle filtering) kullanılmıştır. Oluşturulan damlalar Gaussian Karışım Modelleri (Gaussian Mixture Model) olarak temsil edilmiştir. Hareketlerin tanınması da bu modeller arasındaki farkın karesiyle ölçülmektedir.

Ong ve Bowden'ın yaptığı çalışmada (Ong & Bowden, 2004), deri rengi kullanılarak elde edilmiş el şekillerinden bir eğitim seti oluşturulmuştur. k-Medoid yöntemi ile şekiller gruplandırılmış ve el kontör bilgileri kullanılarak sınıflandırıcı ağaçlar ile el pozları sınıflandırılmıştır.

Stenger ve arkadaşlarının yaptığı çalışmada (Stenger vd., 2006) el tanıma, kenar bilgisi ve deri rengi kullanılarak hiyerarşik Bayes filtre yardımıyla sağlanmıştır.

Buehler ve arkadaşlarının yaptığı çalışmada (Buehler vd., 2008), sürekli görüntüde işaret dili tanınması yapılmıştır. Baş, kol ve el pozisyonunu belirleyen bu çalışmada, HOG yöntemi ile önce kolların yeri tespit edilmiştir. Elle işaretlenmiş veri üzerinden, görüntüdeki arka planlar devre dışı bırakılmak üzere, RGB (Red, Green, Blue) uzayında Gaussian karışımları belirlenmiştir. Model yerleştirme (model fitting) yöntemiyle, iteratif olarak kol modeli bulunmaya çalışılmıştır.

de La Gorce ve arkadaşlarının yaptığı çalışmada (de La Gorce vd., 2008), tek bir kamera görüntüsünden elde edilen el görüntülerinden üç boyutlu el pozu sentezleyebilen bir sistem üretilmiştir. Bu sistem elin aç, doku ve ışıklandırma parametrelerini çözmek için bir objektif fonksiyon içermektedir.

Wang ve Popovic'in yaptığı çalışmada (Wang & Popovic, 2009), içerisinde farklı renkler bulunduran bir eldivenden elde edilen görüntüler, en yakın komşu (nearest neighbour) yöntemi ile poz veri setinden en yakın olan el pozunu bulmak için kullanılmıştır.

Tang ve diğerlerinin yaptığı çalışmada (Tang vd., 2014), derinlik görüntülerinden elde edilen el topolojisi latent tree olarak temsil edilmiştir. Çalışma kendi veri setleri üzerinde yapılmıştır. Latent Regression Forest metodu kullanılarak, eldeki veriden elin üç boyutlu pozunu oluşturmaya çalışmışlardır.

Qian ve arkadaşlarının yaptığı çalışmada (Qian vd., 2014), derinlik görüntülerinden elde edilen el görüntüleri Parçacık Sürü Optimizasyonu (Particle Swarm Optimization, PSO) yöntemi ile takip edilmiştir. X - Y düzleminde parmak uçları belirlenmiş ve el pozuna uygun bir üç boyutlu iskelet oluşturulmuştur.

Sefat ve Shahjahan'ın yaptığı çalışmada (Sefat & Shahjahan, 2015), $YCbCr$ renk uzayında deri rengi kullanılarak gerçek zamanlı el tanıma yapılmıştır. Daha sonra HOG yöntemi ile öznitelik çıkarımı yapıp, SVM ile de bölütleme yapılmıştır. Kendi özgün veri setleri üzerinde çalışılmıştır.

Sharp ve arkadaşlarının yaptığı çalışmada (Sharp vd., 2015), derinlik görüntüleri kullanılmıştır. Daha önceden eğitilmiş bir piksel bazlı sınıflandırıcıyla el bölgesi bulunmuştur. Sentetik bir veri setiyle eğitilmiş bir poz bulucu ile poz eşleştirmesi yapılmıştır. Tespit edilen el pozu üç boyutlu bir modelle temsil edilmiştir.

Urooj ve Borji'nin yaptığı çalışmada (Urooj & Borji, 2018), hem hazır veri setleri hem de yeni veriler üzerinde yapay sinir ağları ile el tespiti yapılmış ve başarıları ölçülmüştür.

Mueller ve arkadaşlarının yaptığı çalışmada (Mueller vd., 2019), tek bir derinlik kamerası ile iki elin de takibi yapılmış ve el pozlarının üç boyutlu modeli çıkarılmıştır. Sentetik derinlik görüntüleriyle eğitim yapılan bir Correspondence Regression ağıyla derinlik görüntülerinden el pozu tespit edilmeye çalışılmıştır. Derinlik görüntülerinden elde edilen noktalar kümesi (point cloud) ile üç boyutlu modeller karşılaştırarak ellerin üç boyutlu modeli bulunmaya çalışılmıştır.

1.3.2. Görüntü Bazlı El Hareketi Tanıma Çalışmaları

Görüntü bazlı çalışmalar, uzaktan tek veya birden fazla kamera ile (Starner vd., 1998, Molchanov vd., 2015) elin belirlenmesi ve izlenmesine dayanır. El belirleme işlemi için arka plan çıkarımına dayalı (Kao & Fahn, 2011), deri rengine dayalı (Cheng vd., 2013), özniteliğe bağlı sınıflandırma (Boulahia vd., 2017), derinliğe dayalı (Kapuscinski vd., 2015), optik akış (Kishore vd., 2016) gibi yöntemler kullanılmıştır. Hareket tanıma için Saklı Markov Modeli (Hidden Markov Model, HMM) (Elmezain vd., 2009), SVM (Benmoussa vd., 2018), CNN (Li vd., 2017) gibi yöntemler kullanılmıştır. Literatürde yer edinmiş bazı görüntü bazlı el hareketi tanıma çalışmaları şöyle özetlenebilir;

Starner ve arkadaşlarının çalışmasında (Starner vd., 1998), Amerikan işaret dilinin hareketlerini tanımak için HMM yöntemi Baum Welch algoritmasıyla kullanılmıştır. 16 öznitelik içeren bir vektör kullanılmıştır. Eller gerçek zamanlı olarak deri rengi kullanarak takip edilmiştir. İki farklı kamera pozisyonu kullanılarak kendi veri setlerini oluşturmuşlardır.

Chen ve arkadaşlarının çalışmasında (Chen vd., 2003), el bölgesi eşikleme, deri rengi ve kenar bulma yöntemleriyle tespit edilmiştir. Öznitelik çıkarımı için Fourier tanımlayıcılar kullanılmış ve HMM ile hareket tanınması yapılmıştır. Kendi veri setleri ile çalışılmış olup 20 farklı hareket tanımlanmıştır.

Liu ve Fujimura'nın çalışmasında (Liu & Fujimura, 2004), el hareketlerini tanımak için Time-of-Flight (ToF) derinlik kamerası kullanılmıştır. Elleri takip etmek için önce histogram yardımıyla kafa tespiti yapılmıştır. Chamfer uzaklığı kullanarak takip edilen ellerin şekli tanınmaya çalışılmıştır. Ayrıca, ellerin izlediği yörünge kullanılarak da hareket tanıma yapılmaya çalışılmıştır.

Patwardhan ve Roy'un yaptığı çalışmada (Patwardhan & Roy, 2004), Eigentracker yönteminin geliştirildiği bir çalışma yapılmıştır. K boyutlu bir vektör, el hareketini temsil için seçilmiştir. Bu vektörü, el şeklini ve yörüngesini temsil eden parametreler olarak belirlenmiştir. Her bir hareket için, o hareketi temsil eden ortalama vektör kullanılarak bir korelasyon matrisi hesaplanmıştır. Bu hareketlerin sınıflandırılabilmesi için Mahalanobis uzaklığı kullanılmıştır. Hareketler, el şeklinin değişmesine göre belirlenmiştir.

Coogan ve arkadaşlarının yaptığı çalışmada (Coogan vd., 2006) renk, hareket ve konum bilgisi kullanılarak el tespiti yapılmıştır. Tasarlanan sistem boyut, konum ve şekil varyasyonlarından etkilenmeyen yapıdadır. Bunu da her bir el şekli için, bir dönüşüm alt

uzayını önceden oluşturarak gerçekleştirmişlerdir. Bu alt uzaylar bilgisayar animasyonu kullanılarak oluşturulan görüntülere PCA uygulanarak oluşturulmuştur. Hareket algılama için iki aşama kullanılmıştır. İlki, çerçeve farkı (frame differencing) yöntemi, ikincisi de her piksele bir olasılık değeri atayarak, ele ait olup olmadığının belirlenmesidir. Bir Kalman filtresi yardımıyla elin yüz ile girişim yapıp yapmadığı da belirlenmiştir. POSER isimli yazılımla 3d el transformasyonları oluşturulmuş, daha sonra HMM ile hareket tanınmaya çalışılmıştır.

Fang ve arkadaşlarının yaptığı çalışmada (Fang vd., 2007), AdaBoost ile bazı öznitelikler kullanılarak el tanıma gerçekleştirilmiştir. Optik akış ve renk bilgisiyle de el izleme gerçekleştirilmiştir. Flock of features isimli bir izleme algoritması da kullanılmıştır. El rengini algılamak için, HSV (Hue, Saturation, Value) renk uzayında bir Gaussian modeli kullanılmıştır. Damlalar (blob), normalize edilmiş Laplas (Laplace) operatörünün karesinin boyut-uzayındaki (scale-space) lokal maksimumlarla elde edilmiştir. Görüntülerde Fourier dönüşümü kullanılarak tekrar AdaBoost ile hareket algılama yapılmıştır. Altı tane hareket tanıma çalışmışlardır.

Birdal ve Hassanpour'un yaptığı çalışmada (Birdal & Hassanpour, 2008), arka plan modellemesiyle görüntülerden arka plan çıkarma yöntemi uygulanmıştır. Görüntüde nesne olmadığı zaman arka plan görüntüsü alınıp, buradan piksellerin ortalama ve standart sapmasına göre arka plan belirlenmiştir. Bu işlem HSV renk uzayında uygulanmıştır. Arka plan çıkarıldıktan sonra, el ve yüz tanıma renk aralığı ile gerçekleştirilmiştir. Hareket tanıma içinse Mean Shift algoritması kullanılmıştır. Eller eğer tanımlanmış hareket tanıma bölgelerine ulaşmışsa, orada öznitelik çıkarma işlemine başlanmıştır. Pozisyon ve öznitelik bilgisine göre de hareket tanımlanmıştır.

Elmezain ve arkadaşlarının yaptığı çalışmada (Elmezain vd., 2009), Arapça 0'dan 9'a kadar numaraları temsil eden hareketleri tanıyan bir sistem oluşturulmuştur. $YCbCr$ renk uzayı kullanılan bu çalışmada, bir pikselin deri olup olmadığı eğitilmiş Gaussian modeliyle belirlenmiştir. Epanechnikov çekirdeği ile renk histogramını çıkardıkları ele, Mean Shift yöntemiyle hareket algılaması uygulamışlardır. El yörüngelerinde konum, yön ve hız gibi öznitelik kombinasyonları kullanılarak en iyi sonuç veren kombinasyon aranmıştır. Bu öznitelik uzayı da k-Means yöntemi ile gruplanıp, HMM'ye girdi olarak sunulmuştur.

Zabulis ve arkadaşlarının yaptığı çalışmada (Zabulis vd., 2009), el tanıma Bayesian ağı (Bayesian Network) kullanılarak gerçekleştirilmiştir. YUV renk uzayında çalışılmıştır. Buradan piksel olasılıkları damlalara (blob) dönüştürülmüştür. Bağlı Bileşenler Analizi

(Connected Component Analysis, CCA) yönteminden sonra bu oluşturulan damlalar kovaryans matrisleriyle belirlenen el şekillerinin temelini oluşturmuştur. El şekillerinden de parmak uçları algılanarak, hareketler sınıflandırılmaya çalışılmıştır. Bu da kural (rule) tabanlı bir sınıflandırıcıyla yapılmıştır.

Suk ve arkadaşlarının yaptığı çalışmada (Suk vd., 2010), deri rengi kullanılarak el takibi yapılmış ve el yörüngesinden öznitelikler çıkarılmıştır. Dinamik Bayes Ağları kullanılarak 10 farklı hareket tanınmıştır.

Kao ve Fahn'ın yaptığı çalışmada (Kao & Fahn, 2011), Bresenham's Midpoint Circle Scan-Conversion algoritması kullanılarak deri rengi üzerinden avuç içi tanıma yapılmıştır. Sabit bir arama penceresiyle her çerçevede el takip edilmiş ve orta nokta üzerinden bir yörünge çıkarılmıştır. Elin yön özelliği ana öznitelik olarak belirlenmiştir. Tek elle 4 hareket, iki elle toplamda 24 hareket tanıyabilecek bir sistem geliştirilmiştir. Eğitilmiş sistemin parametreleri, Viterbi algoritmasına sunulurken, en iyi hareket yolu bulmaya çalışılmıştır. Hareketleri tanımak için HMM kullanılmış, HMM'yi eğitmek için Baum-Welch algoritması kullanılmıştır.

Song ve arkadaşlarının yaptığı çalışmada (Song vd., 2012), havacılıkta kullanılan el hareketlerini tanıyan bir sistem geliştirilmiştir. RGBD kameralardan alınan görüntüler üzerinde çalışılmıştır. Codebook ve derinlik kesme yaklaşımları beraber kullanılarak arka plan çıkarma yapılmıştır. Arka plan çıkarma işleminden sonra, HOG ile öznitelikler elde edilmiştir. Üst vücudu ve elleri temsil eden bir üç boyutlu model oluşturulup, bu modelde poz tahmini PSO ile optimize edilmiştir. Latent-Dynamic Conditional Random Fields (LDCRF) yöntemi ile beraber SVM kullanılarak el şekli sınıflandırılmıştır.

Yun ve arkadaşlarının çalışmasında (Yun vd., 2012), bir ön işleme sonrasında, çeşitli öznitelikler kullanılarak (deri renginin açısı, deri olmayan rengin açısı, deri rengi, ve Hu momentleri) el tanıma yapılmıştır. Öklid uzaklığıyla özdeğerler (eigenvalue) öznitelik değerleriyle (feature value) karşılaştırılmıştır. Çalışma sabit görüntüler üzerinde yapılmıştır. Elin önce merkezini tespit ederek, buradan hareketle farklı çaplarda çemberler hesaplayıp, bu çemberlerden de yola çıkarak parmaklar arasındaki açısını bulmuşlardır. Renk uzayı olarak HSV kullanmışlardır.

Cheng ve arkadaşlarının yaptığı çalışmada (Cheng vd., 2013), konuşma algılama sistemiyle beraber bir el tanıma sistemi geliştirilmiştir. İkisi beraber kullanılarak sisteme çeşitli komutlar verilmiştir. Kalman filtresi ile izleme ve deri rengi kullanılarak el tanınması yapılmıştır. Hareket eden objeleri tanımak için, diamond search algoritması kullanılmıştır.

Block match algoritmasıyla beraber ortalama karesel hata (mean square error) yöntemi kullanılmıştır. Çalışmada YC_bC_r renk uzayı kullanılmıştır. El hareketinin açısı hesaplanıp, 12 aralıktan birine nicelenmiştir.

Wang ve arkadaşlarının yaptığı çalışmada (Wang vd., 2014), Kinect kamerayla RGB ve derinlik görüntüleri kullanılarak el hareketi tanıma yapılmıştır. Eller, Kinect kameranın iskelet yerleştirme sistemi ile takip edilmiştir. El hareketleri arasındaki farklılığı ölçmek için, yeni bir uzaklık metriği olan Superpixel Earth Mover's Distance (SP-EMD) yöntemi geliştirilmiştir. Hem mevcut hem de kendi oluşturdukları veri setiyle yaklaşımlarını test etmişlerdir.

Ma ve Wu'nun yaptıkları çalışmada (Ma & Wu, 2014), ışık değişiminden etkilenmeyen bir sistem geliştirilmiştir. Derinlik kamerasından görüntülerle çalışılmıştır. Algılanan el üç boyutlu modele dönüştürülmüştür. El, ekrana en yakın obje olarak kabul edilmiş ve el tanıma böyle gerçekleştirilmiştir. Eli temsil eden, uzaydaki üç boyutlu noktalar kümesi, PCA ve CCA ile hesaplanmıştır. Tanınan elin Sobel operatörüyle kenarları bulunmuş, elin merkezi ise uzaklık dönüşümü ile bulunmuştur. İki türlü öznelik tanımlanmıştır; birisi çıkıntılar, diğeri de el şeklinden oluşturulan maskenin, ikili koda dönüştürülmüş halidir. Modifiye edilmiş bir PSO yöntemi ile de kullandıkları objektif fonksiyonu optimize etmişlerdir. Çalışma GPU üzerinde gerçekleştirilmiştir.

Ohn-Bar ve Trivedi'nin yaptığı çalışmada (Ohn-Bar & Trivedi, 2014), araçlarda konsolu elle kontrol etmek için bir çalışma yapılmıştır. RGB ve derinlik görüntüleri kullanılarak hareket tanıma yapılmıştır. İlgi bölgesi (Region Of Interest, ROI) olarak, panelin bulunduğu kısım dikkate alınmıştır. HOG ve SVM kullanılarak el bölütlemesi yapılmıştır. Bundan sonra dört farklı yöntem denenerek (HOG-2D, PCA ile birlikte kullanılan HOG, HOG-3D (k-Means ile beraber) ve Dense Trejectory Motion Map) başarıları karşılaştırılmıştır.

Tompson ve arkadaşlarının yaptığı çalışmada (Tompson vd., 2014), gerçek zamanlı video görüntüleriyle Random Forest yöntemiyle bölütleme yapıp, Convolutional Network (ConvNet) kullanılarak öznelik çıkarılmıştır. Aynı zamanda gerçek zamanlı bir ters kinematik çözücü algoritma önerilmiştir. Derinlik kamerası ve RGB görüntüleri bir arada kullanılmıştır ve özgün veri setlerini oluşturmuşlardır. Elde edilen poz değerlerini iyileştirmek için PSO ve Nelder-Mead optimizasyonu kullanmışlardır. Eğittikleri ağlardan ısı haritası (heat-map) görüntüleri alıp, bu görüntülerden de 3d modeller oluşturmuşlardır.

Chen ve arkadaşlarının yaptığı çalışmada (Chen vd. 2014), arka plan çıkarma yöntemiyle el tanıma yapılmıştır. Çalışmada HSV renk uzayı kullanılmıştır. İkili görüntüye dönüştürülen el görüntülerinde uzaklık dönüşümü ile avuç noktası belirlenmiştir. Bu avuç noktasından bir çember çizilerek, parmaklara ve bileğe kadar olan kısım belirlenmiştir. Parmaklar ayrıştırıldıktan sonra, hareketler parmakların duruma göre bir kural sınıflandırıcı (rule classifier) ile sınıflandırılmıştır.

Molchanov ve arkadaşlarının yaptığı çalışmada (Molchanov vd., 2015), Time-of-Flight derinlik kamerası, renkli (RGB) kamera ve bir de kısa mesafe radar kullanılarak araç içi el hareketi tanıma yapılmıştır. Radar hareket algıladığı anda hareketin başladığı kabul edilip kameralar aktif hale getirilmektedir. El, derinlik kamerasında ekrana en yakın cisim olarak kabul edilip ayrıştırılmıştır. Radar ve görüntülerden öznitelik çıkarımı yapılmış ve Konvolüsyonel Sinir Ağı (Convolutional Neural Network, CNN) ile hareket sınıflandırılması yapılmıştır.

Kapuscinski ve arkadaşlarının yaptığı çalışmada (Kapuscinski vd., 2015), derinlik görüntüleri üzerinde çalışılmıştır. Bir noktalar kümesi üzerinde Görüşaçısı Öznitelik Histogramı (Viewpoint Feature Histogram, VFH) adı verilen yöntemle el tanıma gerçekleştirilmiştir. VFH noktalar kümesinin global tanımlayıcısıdır. Dinamik zaman bükmesi (dynamic time warping) ile birlikte k-NN ve HMM sınıflandırıcıları kullanılmıştır.

Ben Jmaa ve arkadaşlarının yaptığı çalışmada (Ben Jmaa vd., 2016) statik görüntüler kullanılmıştır. Fransız alfabesinin 23 el hareketiyle temsili tanınmaya çalışılmıştır. Kinect ile elde edilen RGBD görüntüler üzerinde gürültü filtrelemeden sonra, kenar bulma yapılmış, kenarlardaki açıklıklar kapatılmış, el bölgesi doldurulmuştur. Öznitelikler 2d ve 3d olarak seçilmiştir. 2d olanlar yön, boyut, Euler rakamı iken, 3d öznitelikler varyans derinlik değeri, derinlik ortalaması olarak seçilmiştir. Sınıflandırma Random Forest (RF) ile yapılmıştır.

Kishore ve arkadaşlarının yaptığı çalışmada (Kishore vd., 2016), Hintçe işaret dili hareketleri tanınmaya çalışılmıştır. El hareketi takibi için Horn Schunk Optik Akış (Horn Schunk Optical Flow, HSOF) yöntemi kullanılmıştır. Şekil öznitelikleri aktif kontör modeli (active contour model) yöntemiyle çıkarılmıştır. Ayrıca bir hız vektörü de tanımlanmıştır. Bu öznitelikler bir Artificial Neural Network (ANN) modeline girdi olarak tanımlanmıştır.

Xu'nun çalışmasında (Xu, 2017), gerçek zamanlı görüntülerde hareket tanıma çalışılmıştır. Görüntü, arka plan çıkarma, renk filtresi, Gaussian yumuşatması, eşikleme (thresholding) ve morfolojik operasyonlardan geçirilip ön işleme yapıldıktan sonra, CNN sınıflandırıcıya iletilmiştir. Avuç belirleme uzaklık dönüşümü ile yapılmıştır. Avuç

belirlendikten sonra, el kontörleri, poligonal şekillere Ramer-Douglas-Peucker algoritması ile dönüştürülerek kullanılmıştır.

Boulahia ve arkadaşlarının çalışmasında (Boulahia vd., 2017), HIF3D isimli iskelet bazlı bir öznitelik seti kullanılmıştır. Leap Motion cihazı ile yeni bir dinamik el hareketi kütüphanesi oluşturulmuştur. Hareket sınıflandırma için SVM kullanılmıştır.

Li ve arkadaşlarının çalışmasında (Li vd., 2017), Log Path Signature adını verdikleri bir öznitelik üzerinden el hareketi tanıma gerçekleştirilmiştir. Hem derinlik hem RGB (Red Green Blue) görüntüleri kullanılmıştır. El tanıma için Bölge Bazlı Konvolüsyonel Sinir Ağı (Region Based CNN, R-CNN) kullanılmıştır. Sınıflandırma için CNN kullanılmıştır.

Liu ve arkadaşlarının yaptığı çalışmada (Liu vd., 2017), RGB ve derinlik görüntüleri üzerinde Fast R-CNN ile el tanıma yapılmış, 3d Convolutional Network (C3D) ile öznitelik çıkarımı yapılmış ve sınıflandırma için SVM kullanılmıştır. Continuous Gesture Dataset (ConGD) üzerinde çalışılmıştır.

Azad ve arkadaşlarının yaptığı çalışmada (Azad vd., 2018), el hareketi içeren derinlik görüntüleri üzerinde, Multilevel Temporal Sampling (MTS) adı verilen yaklaşımla öznitelik çıkarılmıştır. Bu yaklaşımda, Depth Motion Maps (WDMM), HOG, VLAD kodlamasıyla Local Binary Pattern (LBP) yöntemleri kullanılmıştır. Bu yöntemler dört açık veri setine uygulanmış, Single Hidden Layer Feedforward Network (SLFN) yöntemiyle sınıflandırma yapılmıştır.

Benmoussa ve arkadaşlarının çalışmasında (Benmoussa vd., 2018), Kinect kamerası ile elde edilen derinlik görüntüleri kullanılmıştır. SIFT ve SURF algoritmalarının sunduğu noktalar üzerinden el takibi gerçekleştirilmiştir. Bag-of-Words yaklaşımıyla, çıkarılan noktalardan codeword sözlükleri oluşturulmuş, bu sözlüklerden hesaplanan histogramlar ise SVM sınıflandırıcıyla sınıflandırılmıştır.

Owoyemi ve Hashimoto'nun yaptığı çalışmada (Owoyemi & Hashimoto, 2018), Kinect kameradan elde edilen noktalar kümesi ile 10 el hareketi tanınmıştır. 3D-CNN yöntemiyle sınıflandırma gerçekleştirilmiştir.

Abavisani ve arkadaşlarının yaptığı çalışmada (Abavisani vd., 2019), farklı modalitelerden öğrenen ağlar arasında bağlantı kuran Spatiotemporal Semantic Alignment (SSA) hata hesaplama metodu geliştirilmiştir. Bu yaklaşım el hareketi tanımada üç farklı açık veri setinde test edilmiştir.

Chen ve arkadaşlarının çalışmasında (Chen vd., 2019), iskelet bazlı el verisinden parmakların öznitelikleri varyasyonel otokodlayıcı (Variational Autoencoder, VAE)

yöntemiyle çıkarılmıştır. Hem elin global hareket öznitelikleri hem de parmakların hareket öznitelikleri kullanılarak hareket tanıma gerçekleştirilmiştir. Sınıflandırma için LSTM ağı kullanılmıştır. Hazır bazı iskelet veri setlerini kullanılmıştır.

1.3.3. Sensör Bazlı El Hareketi Tanıma Çalışmaları

Sensör bazlı el hareketi tanıma çalışmalarında, ellere veya bileklere takılan sensörlü çeşitli cihazlar, radar benzeri prensipte çalışan sensörler kullanılmaktadır. Literatürdeki bazı sensör bazlı çalışmalar şöyle özetlenebilir;

Zhang ve arkadaşlarının yaptığı çalışmada (Zhang vd., 2011), Çince işaret dilinde bulunan 72 el hareketinin tanınması yapılmıştır. El yörüngesi, ortalama mutlak değer (mean absolute value) ve dördüncü dereceden özbağlanımlı (autoregressive) katsayılar (coefficients) öznitelikler olarak kullanılmıştır. Karar ağaçları ve HMM sınıflandırma için kullanılmıştır.

Kim ve Toomajian'ın yaptığı çalışmada (Kim & Toomajian, 2016), el hareketi tanıma için micro-Doppler imzalarının spektrogramları kullanılmıştır. 10 farklı hareketin sınıflandırılması için Derin Konvolüsyonel Sinir Ağı (Deep CNN, DCNN) kullanılmıştır.

Kim ve diğerlerinin yaptığı çalışmada (Kim vd., 2017), ellerden yansıyan Impulse Radio (IR) sinyalleri el hareketi tanıma için kullanılmıştır. CNN ve SVM kullanılarak Amerikan işaret dilindeki 6 hareket tanınmaya çalışılmıştır.

Zhang ve arkadaşlarının yaptığı çalışmada (Zhang vd., 2018), FMCW radar sensörüyle el hareketlerinin spektrogramı elde edilmiştir. Connectionist Temporal Classification algoritmasıyla beraber LSTM ve 3D-CNN ağları sınıflandırma için kullanılmıştır.

Amin ve arkadaşlarının yaptığı çalışmada (Amin vd., 2019) micro-Doppler spektrogram görüntüleriyle 15 el hareketinin tanınması yapılmıştır. El yordamıyla belirlenen özniteliklerin yanı sıra spektrogramların zarfları (envelope) da öznitelik olarak kullanılmıştır. k-NN ve SVM sınıflandırıcılar dört farklı uzaklık metriğiyle beraber kullanılmıştır.

1.3.4. Görüntü Bazlı El Yıkama Hareketi Tanıma Çalışmaları

Literatürde bulunan görüntü bazlı el yıkama hareketi tanıma çalışmalarında, önceki el hareketi tanıma çalışmalarına benzer yöntemler yer almaktadır. Çalışmalarda hareket tanımları ve sayıları değişken olup, kullanılan veri setlerinde de ortaklık bulunmamaktadır.

Literatürde yer alan görüntü bazlı el yıkama hareketi tanıma çalışmaları kısaca şöyle özetlenebilir;

Llorca ve arkadaşlarının yaptığı çalışmada (Llorca vd., 2011), deri rengi ve hareket analizi kullanılarak el tespiti yapılmıştır. Parçacık filtresi ve k-Means algoritmaları ile el takibi yapılmış, HOG ve Histogram of Oriented Optical Flow (HOF) ile öznitelik çıkarımı yapılmıştır. SVM kullanılarak 6 hareketi sınıflandırılmıştır.

Xia ve arkadaşlarının yaptığı çalışmada (Xia vd., 2015), RGBD görüntüleriyle birlikte Gaussian Expectation Maximization yöntemi kullanılarak el bölgesi belirlenmeye çalışılmıştır. HOG ve PCA kullanılarak öznitelik çıkarımı yapılmıştır. SVM, Lineer Diskriminant Analizi (Linear Discriminant Analysis, LDA) ve RF yöntemleriyle 12 tane yıkama hareketi sınıflandırılmıştır.

Ivanovs ve arkadaşlarının yaptığı çalışmada (Ivanovs vd., 2020), el yıkama değerlendirmesi için video formatında özgün veri seti oluşturulmuştur. MobileNet-V2 ve Xception ağları 7 farklı hareketin sınıflandırılması için kullanılmıştır.

Prakasa ve Sugiarto'nun yaptığı çalışmada (Prakasa & Sugiarto, 2020), deri rengi kullanılarak el tespiti yapılmıştır. CNN kullanılarak 6 hareket sınıflandırılmıştır.

Vo ve arkadaşlarının yaptığı çalışmada (Vo vd., 2021), 7 hareketten oluşan ve 6 farklı görüntü açısından alınmış RGB görüntülerinden oluşan bir özgün veri setiyle çalışılmıştır. MobilNet-V2, ResNet-18, Inception-V3 ve Augmented Multiscale Deep Infomax (AmDim) ağları sınıflandırma için kullanılmıştır.

Zhong ve arkadaşlarının yaptığı çalışmada (Zhong vd., 2021), farklı kamera açılarıyla özgün veri seti oluşturulmuştur. İlgi bölgeleri (ROI, Region of Interest) el ile işaretlenmiştir. RGB görüntüleri ve Optik Akış (Optical Flow) verileri ResNet-50, ResNet-50-LSTM ve Temporal Relational Network (TRN) ağları ile kullanılmış, 4 farklı el hijyeni hareketi sınıflandırılmıştır.

Xie ve arkadaşlarının yaptığı çalışmada (Xie vd., 2022) özgün video veri seti üzerinde CNN, öznitelik biriktirme modülü (feature aggregation) ile birlikte kullanılarak 7 hareket sınıflandırılmıştır.

1.3.5. Sensör Bazlı El Yıkama Hareketi Tanıma Çalışmaları

El yıkama hareketi tanıma için sensör bazlı yaklaşımlar da mevcuttur. Bu çalışmalar kısaca şöyle özetlenebilir;

Mondol ve Stankovic'in yaptığı çalışmada (Mondol & Stankovic, 2015), akıllı saatler kullanılarak el yıkama kalitesi ölçülmüştür. Sensörlerin x , y , z eksen bilgilerinden çıkarılan farklı öznitelikler kullanılmıştır. Karar ağaçlarıyla iki ayrı özgün veri setinden 7 ve 10 yıkama hareketi sınıflandırılmıştır.

Wang ve arkadaşlarının yaptığı çalışmada (Wang vd., 2020), Myo kolbantları kullanılarak DSÖ'nün önerdiği 10 farklı el yıkama hareketinin verileri toplanmıştır. IMU ve sEMG verilerinden birçok öznitelik çıkarılmış ve Extreme Gradient Boosting (XGBoost) yöntemi ile sınıflandırma yapılmıştır.

Samyoun ve arkadaşlarının yaptığı çalışmada (Samyoun vd., 2021), 10 adıma sahip el yıkama sürecini akıllı saat ile sınıflandıran bir sistem tasarlanmıştır. Sensörlerden elde edilen istatistiksel öznitelikler ve CNN ile elde edilen öznitelikler LSTM ağlarına kombine edilerek sınıflandırılma yapılmıştır.

Santos-Gago ve arkadaşlarının yaptığı çalışmada (Santos-Gago vd., 2021), akıllı saatler SVM, RF, Logistic Regression ve ANN yöntemleriyle kullanılarak el yıkama tanıma yapılmıştır.

Wang ve arkadaşlarının yaptığı çalışmada (Wang vd., 2021), el yıkama takibi için Uwash isminde bir sistem önerilmiştir. Akıllı saatlerden elde edilen ivmeölçer (accelerometer) ve jiroskop verileriyle, özelleşmiş bir U-Net ağı mimarisi kullanılarak dokuz farklı hareket sınıflandırılmıştır.

Zhang ve arkadaşlarının yaptığı çalışmada (Zhang vd., 2021), bileklerde Byteflies sensörleri kullanılarak el yıkama verileri toplanmıştır. Öznitelikler zaman ve frekans bilgi alanlarında çıkarılmıştır. k-NN, XGBoost, SVM ve Feed Forward Neural Network (FFN) yöntemleriyle el yıkamanın 11 adımı sınıflandırılmıştır.

1.3.6. El Hareketi Tanımada Kullanılan Kamera Türleri

El tanıma için kullanılan görüntü bazlı donanımlar incelendiğinde, çeşitli kamera türleri göze çarpmaktadır. Kamera çeşidi ve alınan görüntülerin niteliği, çalışmanın ne şekilde ilerleyeceğine doğrudan etki etmektedir. Günümüzde çalışmalarda kullanılan öne çıkan kamera tipleri ve sundukları avantajlar şu şekilde sıralanabilir;

- Derinlik Kameraları: Bir sahneye ait görüntüdeki cisimlerin uzaklığını hesaplayabilen kameralar derinlik kameralarıdır. Derinlik bilgisi hesaplamanın birden fazla yöntemi vardır. Yöntemlerden biri, belirli bir desene sahip kızılötesi

ışığı bir kaynaktan yansıtıp, cisimler üzerindeki iz düşümüne göre derinlik bilgisi hesaplamaktır. Desen bilgisi önceden bilindiği için, desende meydana gelen değişiklikler mesafeyi bildirir. Bu yöntemi kullanan kameraya Kinect örnek verilebilir. Ortamda birden fazla bu kameralardan varsa hatalı görüntü elde edilebilir. Işık yansımalarının fazla olduğu yüzeyler de görüntüde hataya sebebiyet verebilir. Time-of-Flight (ToF) adı verilen kamera türlerinde farklı ışık kaynakları tarafından yayılan ışık bilgisinin, kameraya geri yansımaya kadar geçen süre hesaplanarak cisimlerin uzaklığı bulunabilir. Devamlı ışık kullanan bir yapıda ise, yansıyan ışıktaki faz değişimi bilgisi de yine derinliği hesaplamakta kullanılır. Ortamdaki köşeler ışık bilgisinin birden fazla kere yansımaya, yanlış görüntüler elde edilmesine yol açabilir. Aynı şekilde yansıtıcı materyaller, güneş veya ortamdaki başka ToF kameralar yanlış görüntülere sebebiyet verebilir. LiDAR kameralar bu yöntemle çalışmaktadır. Stereo kameralar ise, 1950’li yıllardan beri kullanılmakta olan bir teknoloji olup, görüntüyü çift lens kamera kullanarak elde eder. İki kamera arasındaki kalibrasyon bilgileri kullanılarak, alınan çift görüntüden bir derinlik bilgisi oluşturulur. Bu derinliği hesaplamak için kullanılan en basit algoritmalarından biri SSD (Sum of Squared Differences) algoritmasıdır. Bu algoritmada, kameraların odak uzunluğu (focal length), kameraların açısı, kameralar arası mesafe, disparite (üç boyutlu uzaydaki bir noktanın iki kameradaki izdüşümlerinin arasındaki fark) kullanılarak block-matching algoritmasıyla beraber derinlik hesaplanabilir.

- RGB kameralar: Standart dijital kameralardır. Ekranı yansıyan ışık bilgisi sensör tarafından üç kanal renk bilgisine (Red, Green, Blue) dönüştürülüp iki boyutlu düzleme yansıtılır. Bu kameralar ışık ve arka plan değişiminden oldukça etkilenirler ve obje tanıma için kullanılmadan önce görüntünün çeşitli ön işlemlerden geçmesi gerekebilir.

Bu tez çalışmasında, RGBD görüntü sunabilen Kinect 2 (URL-3, 2023) kamerasıyla çalışılmıştır. Kinect 2 kameranin sunduğu derinlik görüntülerinden elde edilen noktalar kümesi, benzer amaçlı diğer kameralara göre daha başarılıdır (URL-4, 2023). Kinect, ilk olarak Microsoft tarafından 2010 yılında piyasaya sürülmüş olan, hareket algılamaya yönelik, RGB kamerası ve kızılötesi sensörleri bulunan bir cihazdır. Amacı, Xbox oyun konsolunda oyunları hareketlerle kontrol imkanı sunmak olsa da, birçok yapay zeka uygulamasında tercih edilmektedir. Birinci ve ikinci versiyonlarının yanı sıra, 2020 Mart

ayında Azure Kinect olarak son modeli kullanıma sunulmuştur. İlk Kinect kamerası kızılötesi ışık deseni prensibiyle çalışırken, 2. versiyonu ToF prensibiyle çalışmaktadır. Kinect sensörü, aynı zamanda görüntülediği alanda önceden eğitilmiş sınıflandırıcılarıyla insan vücudunu algılayıp, basit bir iskelet modeliyle de gerçek zamanlı iskelet bilgisi sunabilmektedir.

Farklı üreticilerin derinlik kameralarıyla ilgili detaylı ve güncel bir liste (URL-5, 2023) kaynağında görülebilir.

1.3.7. Literatürden Çıkarılan Sonuçlar

Literatürde el ve el hareketi tanıma için birçok yöntem kullanılmaktadır. Gün geçtikçe gelişen ve kullanım alanları genişleyen yapay sinir ağları, en çok ön plana çıkan yöntem olmaktadır.

El hareketi tanımanın ilk adımı eli doğru tanıyıp zaman akışı boyunca doğru takip edebilmektir. El tanıma obje tanıma araştırma alanının bir parçasıdır. Obje tanımada genel olarak öznitelikler belirlenerek benzer objeler sınıflandırılmaya çalışılmaktadır. Örnek verilerden çeşitli algoritmalar kullanılarak elde edilen özniteliklerden gerçek verilerde obje bulunmaya çalışılır. Yapay sinir ağları modelleri, diğer öznitelik çıkarma yöntemleriyle beraber kullanılabildiği gibi hem öznitelik çıkarma hem de sınıflandırma işlemleri yapabilmektedir.

Tez konusu çalışma için, literatürdeki birçok standart yöntem denenmiş ve sonuçları bir sonraki bölüm olan yapılan çalışmalar bölümünde belirtilmiştir.

1.4. Görüntü İşleme ile İlgili Temel Bilgiler

1.4.1. Arka Plan Çıkarımı

Görüntü işlemede, kameranın sabit olduğu ve arka planın değişmediği durumlarda, arka plan bilgisi basitçe görüntüden arındırılarak sadece sahneye sonradan giren veya hareketli objeler tespit edilebilir. Bu tespit, kullanım senaryosuna göre basitçe iki görüntünün farkına bakılarak ya da birden fazla çerçevenin ortalamasına bakılarak yapılabileceği gibi, kompleks makine öğrenmesi yöntemleri ile de yapılabilir. Bu alandaki en yaygın yöntemlerden biri Gaussian Karışım Modelleri (Mixture of Gaussians, MOG) kullanılarak yapılan arka plan çıkarımıdır (Stauffer & Grimson, 1999). Son yıllarda

geliştirilen birçok farklı yöntemle kompleks durumlar için de arka plan çıkarımı yapılabilmektedir (Garcia vd. 2020). Şekil 1’de arka plan çıkarımına bir örnek görülmektedir.



Şekil 1. OpenCV’de Gaussian Karışım Modelleri kullanılarak yapılan arka plan çıkarma örneği (URL-6, 2023)

Gaussian Karışım Modelleri, görüntüdeki her bir pikselin sahip olduğu değerlere (RGB veya parlaklık gibi) bakarak, pikselin birden fazla Gaussian dağılımı üzerinde olasılığını bulan ve görüntü bölütlemeye (segmentasyona) yarayan gözetimsiz öğrenme yöntemidir. Eğer kümeleme değeri K iki seçilirse, arka planı ve objeleri temsil eden iki Gaussian dağılımı ile arka plan çıkarımı için de kullanılabilir.

Bu yöntemin adımları şu şekildedir; başlangıçta verilen veri kümelerinin dağılımını temsil için olasılık yoğunluk fonksiyonları tanımlanır. Her bir piksel (x) için, Formül 1’deki gibi bir olasılık belirlenir;

$$P(x|b) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_b|}} e^{\frac{-1}{2}(x-\mu_b)^T \Sigma_b^{-1} (x-\mu_b)} \quad (1)$$

Burada b , ilgili veri kümesini, Σ kovaryans matrisini, $|\Sigma_b|$ ise bu matrisin determinantını temsil eder. Euler sayısı üssünde ise Mahalanobis uzaklığı formülü bulunmaktadır. μ , veri kümesinin ortalamasını (mean) temsil eder.

$$P(b|x) = \frac{P(x|b)P(b)}{\sum_{i=1}^n P(x|i)P(i)} \quad (2)$$

Bir x noktasının, b veri kümesine ait olasılığını bulmak için yukarıdaki Bayes formülü kullanılır. $P(b)$, önsel olasılığı yani b veri kümesinin toplam veriler içerisinde tuttuğu miktarı temsil eder. Paydada ise, tüm veri kümelerinin olasılık dağılım fonksiyonları ve önsel olasılıklarının çarpımlarının toplamı bulunur.

Bu aşamalar hesaplandıktan sonra, yeni ortalamalar (μ), önsel olasılıklar $P(b)$ ve kovaryans (Σ) matrisi hesaplaması için aşağıdaki (3), (4), (5) formülleri kullanılır.

$$P(b) = \frac{1}{n} \sum_{x=1}^n P(b \vee x) \quad (3)$$

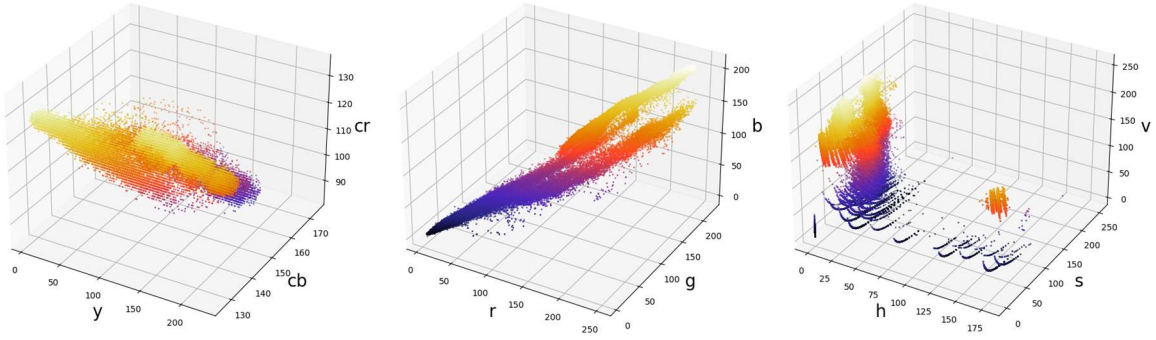
$$\Sigma_b = \frac{\sum_{x=1}^n P(b \vee x)(x - \mu_b)(x - \mu_b)^T}{\sum_{x=1}^n P(b \vee x)} \quad (4)$$

$$\mu_b = \frac{\sum_{x=1}^n P(b \vee x)x}{\sum_{x=1}^n P(b \vee x)} \quad (5)$$

1.4.2. Eşikleme

Dijital görüntülerde piksellerin her biri görüntüyü temsil eden bir nokta gibi düşünülebilir. Bir pikselin rengi yaygın olarak üç renk kanalının karışımından elde edilir; kırmızı, yeşil ve mavi (RGB). Her bir kanalın renk değerini temsil eden bir sayısal değer bulunur. Bu değer, 0-255 arasında 8 bit kullanılarak temsil edilir. Böylece üç kanaldan 24 bit renk derinliğine sahip bir görüntü oluşur.

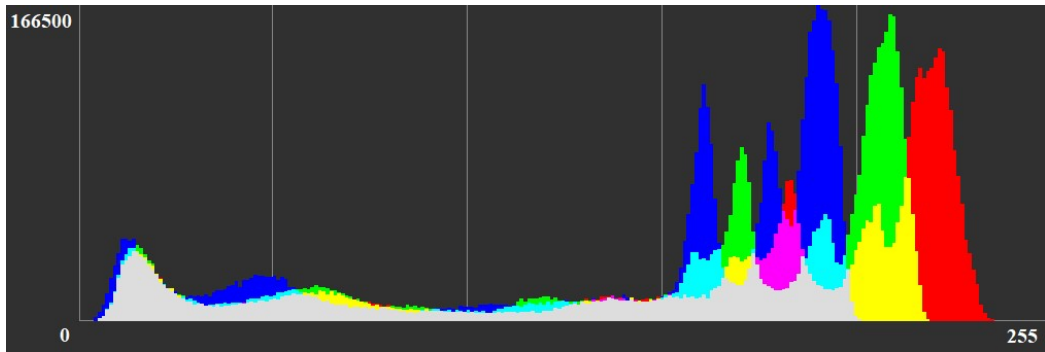
Renklerin bilgisayar ortamında temsil edilmesi birden fazla yolla olabilir. Renkler arası geçiş formülleri ve temsil edildikleri sayısal değerlere göre farklı renk uzayları (color space) geliştirilmiştir. RGB, HSV, YUV, YCbCr renk uzayları görüntü işlemede yaygın olarak kullanılan bazılarıdır. Bu renk uzayları, çeşitli formüller yardımıyla birbirlerine dönüştürülebilmektedir. Şekil 2’de, aynı görüntünün farklı renk uzaylarındaki histogramları gösterilmiştir.



Şekil 2. Bir görüntünün RGB, HSV ve YCbCr renk uzaylarında üç boyutlu histogramları

Görüntü işleme uygulamalarında, her kanala ait renk değerlerine bağlı olarak eşikleme işlemleri gerçekleştirilebilir. Bir obje veya görüntüdeki bir parça, istenilen renk aralıklarına denk düşüyorsa, o renk aralığındaki pikseller ayırt edilecek biçimde bir işlemde geçirilerek asıl görüntüden ayrılıp tespit edilebilir. Bu işleme eşikleme (thresholding) denmektedir (Gonzalez & Woods, 2018).

Bir görüntüdeki her bir parlaklık ya da renk seviyesine sahip olan piksellerin sayısının grafik halinde gösterimine histogram denir. Histogramlar, eşikleme işleminde çok kullanışlı olabilmektedir. Bir görüntüdeki piksellerin renk veya parlaklık bilgileri, belirli değerlerde gruplanıyorsa, bu görüntüde eşikleme yapmak daha kolay olacaktır. Şekil 3'te, yatayda piksel değerleri ve düşeyde piksel sayısı olacak şekilde gösterilmiş bir histogram örneği bulunmaktadır.

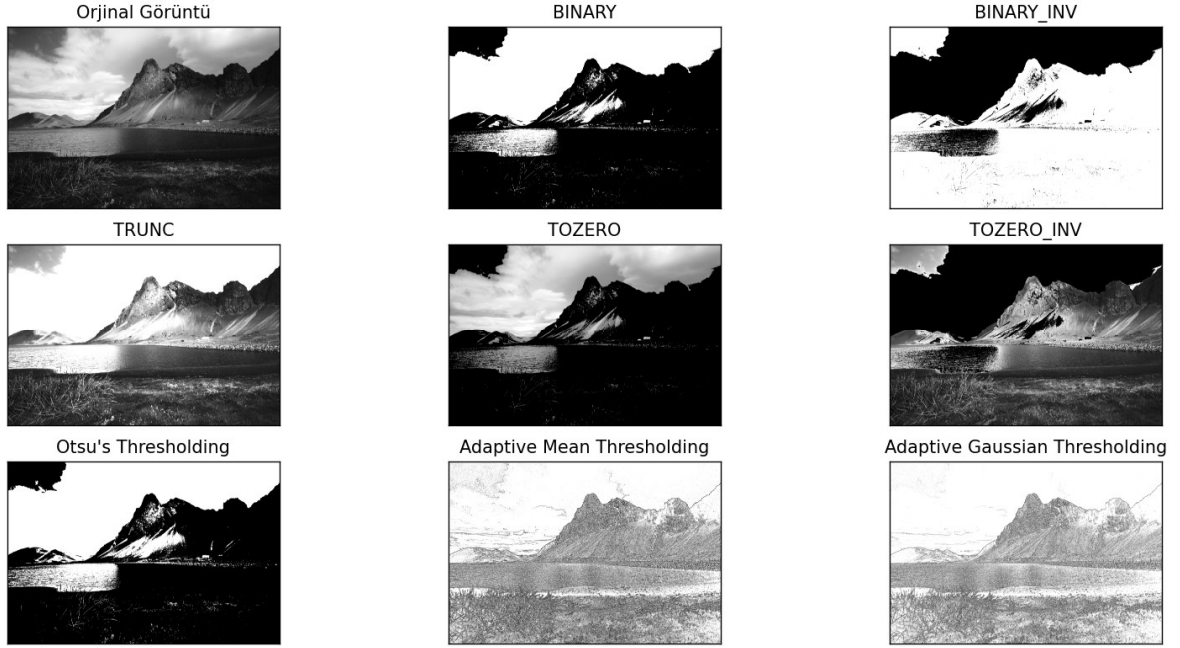


Şekil 3. 0-255 değerleri arasında bir görüntünün Kırmızı, Yeşil, Mavi kanallar ve Yoğunluk (Intensity) piksel miktarlarını gösteren bir histogram örneği

Eşikleme yapılan görüntü sonucunda ikili (binary) bir görüntü elde edilir. İstenen eşik değerinin altında kalan kısımlar siyah, eşik değerine erişen ve üzerinde kalan kısımlar beyaz piksellerle gösterilir. Eşiklemenin genel ifadesi Formül 6'da verilmiştir. Eşikleme ile ilgili kapsamlı bir çalışma için Sezgin ve Sankur'un çalışmasından yararlanılabilir (Sezgin &

Sankur, 2004). İnsan derisine ait renk aralığını etkin biçimde tespit etmek için farklı renk uzaylarında yapılmış çalışmalar mevcuttur (Kolkur vd., 2017). Şekil 4’te farklı eşikleme yöntemlerine örnekler gösterilmiştir.

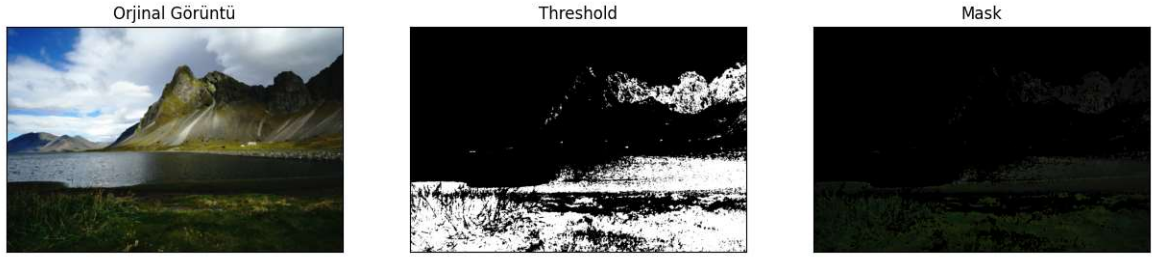
$$\begin{pmatrix} \text{pikseldeğeri} > \text{eşik} \rightarrow \text{değer: 255} \\ \text{pikseldeğeri} < \text{eşik} \rightarrow \text{değer: 0} \end{pmatrix} \quad (6)$$



Şekil 4. Gri düzey bir görüntü (sol üst) ve eşikleme sonrası oluşan ikili görüntüler (sağda)

1.4.3. Maskeleye

İkili görüntülerin sunduğu olanaklardan birisi de maskeleye yöntemidir. Orijinal görüntüden elde edilen ikili görüntülerdeki beyaz pikseller, orijinal görüntüde denk düştükleri kısımlarla eşleştirilip ilgili bölgenin görüntünün geri kalanından ayırt edilmesini sağlarlar. Bu operasyon, programlamada da bulunan “ve (and, &)” operatörü mantığıyla çalışmaktadır. Orijinal görüntü ve ikili görüntü bir tür ve işlemine sokulur. Aynı koordinatlardaki pikseller eğer ikili görüntüde beyaz ise maskelenmiş görüntüye aktarılır, aksi durumda aktarılmaz. Şekil 5’te maskeleye işlemine bir örnek gösterilmiştir.

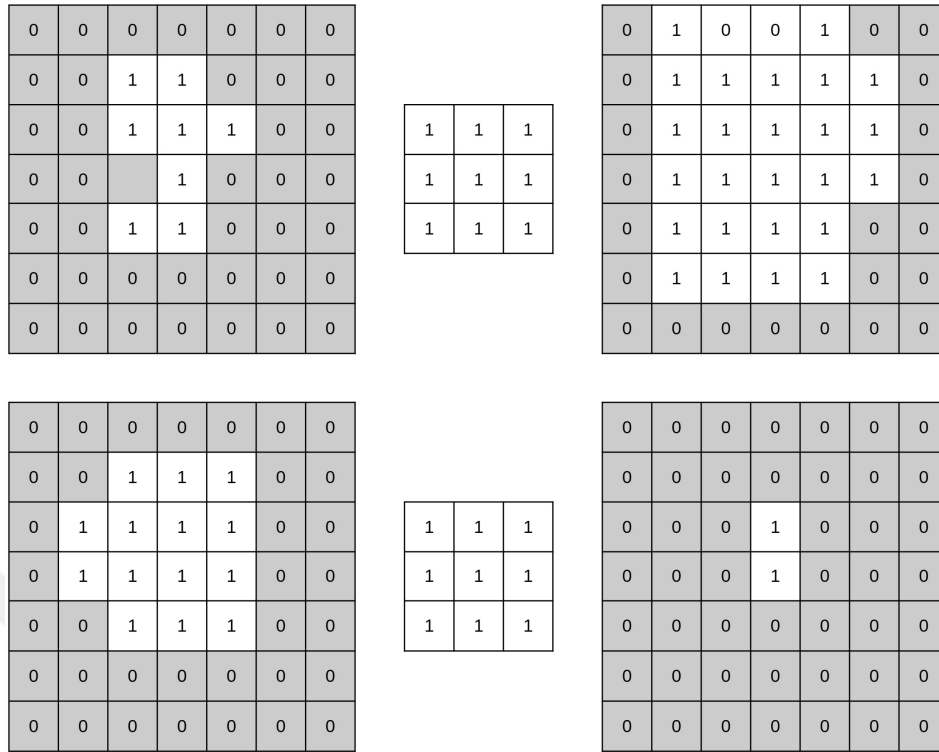


Şekil 5. Orijinal görüntü (solda), ikili görüntü (ortada), maskelenmiş görüntü (sağda)

1.4.4. Morfolojik Operatörler

Görüntü işlemede, ikili görüntüler elde edildikten sonra, beyaz piksellerde bazı istenmeyen durumlar oluşabilir. Örneğin, bir cismi temsil eden piksellerde boşluklar meydana gelmiş olabilir. Ya da cismin boyutu değiştirilmek isteniyor olabilir. Bu durumlarda bazı özel çekirdek filtreler görüntü üzerinde konvolüsyon yöntemiyle uygulanarak istenen sonuçlar elde edilir. Bu filtreler istenen boyutta ve değerlerde tasarlanabilir.

Morfolojik operatörler dört adettir. Bunlar aşındırma (erosion), genleşme (dilation), açma (opening), kapamadır (closing) (Gonzalez & Woods, 2018). Şekil 6'da, 3x3'lük bir çekirdeğin ikili görüntülerde genleşme ve aşındırmasına örnek verilmiştir. İlk örnekte çekirdek, arka plan (0 değerli pikseller) üzerinde dolaştırılmış ve herhangi bir matris elemanı bir objeyle (1 değerli pikseller) kesişiyorsa o piksel beyaza dönüştürülmüştür. Genleşme işlemi bu şekilde sağlanmıştır. İkinci örnekte ise, çekirdek obje üzerinde dolaştırılmış ve herhangi bir matris elemanı arka plan pikseliyle kesişiyorsa o piksel siyaha dönüştürülmüştür. Aşındırma işlemi bu şekilde sağlanmıştır. Genleşme işlemi $\oplus$ sembolüyle, aşındırma işlemi $\ominus$ sembolüyle, ifade edilir.



Şekil 6. Morfolojik operatörlerin ikili görüntülerde oluşturduğu etkiye örnekler. Genleşme (dilation) (üstte), aşındırma (erosion) (altta)

1.4.5. Bağlı Bileşenler Analizi

İkili görüntülerde elde edilen bölgelerin sayısı bilinmek istendiğinde, şekilleri birbirinden bağımsız olarak etiketleyip sayısını bulabilecek bir yöntem olan Bağlı Bileşenler Analizi (Connected Component Analysis) kullanılmaktadır. Bu yöntemde, tüm görüntünün pikselleri ele alınarak, piksel eğer arka plan (background) değil de cisme ait ise (foreground) etiketleme işlemi gerçekleştirilir. Bu etiketleme işlemi gerçekleştirilirken, soldan sağa doğru görüntünün matrisi taranır. Bir piksel eğer etiketlenilmemişse, komşu olduğu diğer 8 piksele bakılır. Komşu piksellerde etiket varsa en düşük değerli etiket ile etiketlenir. Daha önce etiketlenmemiş ve komşuluğu da etiketlenmemiş bir pikselle karşılaşıldığında bir etiket değeri verilir. Böylece tüm beyaz pikseller komşuluk bilgisine göre etiketlenmiş olur. Ancak, ilk seferde bu işlem tamamlandığında, en az etiket sayısı ile bitmeyebilir. Bu yüzden etiketleme bittiğinde tekrar komşuluklar değerlendirilerek tüm komşu piksellerin aynı etiketi alması sağlanır. Bu işleme iki geçiş (two-pass) adı verilir.

Bağlı bileşen analizi, görüntüdeki cisimlerin sınırlarını belirlemeye yardımcı olduğu gibi, cisimlerden kontör bilgisi elde edip boyutlarını bulmaya, kırpmaya, takip etmeye yarar. Şekil 7’de nasıl çalıştığına dair bir örnek verilmiştir. Beyaz piksellerle ifade edilen cisimler ilk seferde soldan sağa doğru taranarak etiket numaraları verilmiştir. İkinci geçişte ise, pikselin komşularına bakılarak en düşük değerli etiket numarası seçilir. Bu işlem tüm cisimler ayrılana kadar tekrarlanır.

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	2	2	0	0
0	1	1	0	0	2	2	2	2	0
0	1	1	1	1	1	0	0	2	0
0	0	0	0	0	0	0	0	2	0
0	0	3	3	3	0	0	0	2	0
0	0	3	3	3	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	0	0
0	1	1	0	0	1	1	1	1	0
0	1	1	1	1	1	1	0	0	1
0	0	0	0	0	0	0	0	1	0
0	0	2	2	2	0	0	0	1	0
0	0	2	2	2	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Şekil 7. Bağlı Bileşenler Analizinin ikili görüntülerdeki algoritma aşamaları. İlk etiketleme (solda) ve ikinci geçişten sonra (sağda) tespit edilen cisimler

1.4.6. Şablon Eşleştirme

Elde bulunan sabit bir görüntü parçasının, başka görüntülerde de bulunup bulunmadığını tespit edebilmek için kullanılan şablon eşleştirme (template matching) yöntemi, basit düzeylerde obje tanıma için kullanılabilir. OpenCV kütüphanesi yardımıyla, altı farklı matematiksel formülle kullanılabilen bu yöntem, en iyi sonuçları sabit görüntüler üzerinde vermektedir. Yöntemin çalışması şu şekildedir; aranmak istenen objenin bir görüntüsündeki tüm pikseller, görüntü üzerinde denk düştüğü piksellerde gezdirilip, matematiksel formüllerle benzerliğe bakılır (URL-7, 2023). Yüksek değerlere (lokal maksimuma) ulaşıldığı yerlerde nesnenin var olduğuna dair çıkarımda bulunulur. Formül 7’de örnek olarak normalize edilmiş correlation coefficient (korelasyon katsayısı) metodu verilmiştir. Kaynak görüntü I , şablon görüntü T ile temsil edilmektedir.

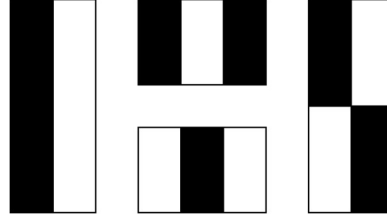
$$R(x, y) = \frac{\sum_{x', y'} (T(x' y') \cdot I(x + x', y + y'))}{\sqrt{\sum_{x', y'} (T(x' y')^2 \cdot \sum I(x + x', y + y')^2)}} \quad (7)$$

1.4.7. Viola-Jones Algoritması

Viola-Jones algoritması, 2001 yılında geliştirilmiş bir obje tanıma yöntemidir (Viola & Jones, 2001). Temel amaç, Haar çekirdeklerinden faydalanıp yüz tanıma için gerekli özellikleri tespit edebilmektir. Dikdörtgen şeklinde tanımlanan bu çekirdekler, görüntü üzerinde dolaştırılarak, çekirdeğin ilgili bölgeleri altındaki piksellerin toplamalarının farkları alınır. Birden fazla çekirdek için ve farklı boyutlardaki çekirdekler için, pratikte çok fazla hesaplama gerektirecek bu yöntemin kolaylaşması adına iki yöntem sunulmuştur. Bunlardan biri integral görüntüdür. İntegral görüntü, görüntüdeki her bir pikselin kendinden üstte ve solda kalan piksel değerlerinin toplamını temsil etmesidir. Böylece öznitelik çıkarma çok daha kolaylaşmaktadır. Diğer yöntemde ise, en kesin sonuçları veren çekirdeklerin önce uygulanıp, vereceği sonuca göre diğer çekirdeklerin uygulanıp uygulanmayacağına kararı alınır. Eğer bu çekirdek pozitif sonuç üretirse arama yapılmaya devam edilir, yoksa vazgeçilir. AdaBoost gibi bir sınıflandırıcı yardımıyla bu çıkarılan öznitelikler üzerinden ilgilenilen obje aranır. Şekil 8’de normal bir görüntünün integral görüntüsü verilmiştir. Şekil 9’da ise Viola-Jones algoritmasında kullanılan çekirdeklerden örnekler görülmektedir.

4	7	3	5	20	4	11	14	19	39
6	5	18	4	2	10	22	43	52	74
1	8	2	10	5	11	31	54	73	100
12	3	4	6	7	23	46	73	98	132
16	1	9	2	3	39	63	99	126	163

Şekil 8. Normal görüntü (solda), integral görüntü (sağda)



Şekil 9. Viola-Jones algoritmasında öznitelik çıkarmak için kullanılan çekirdeklerden örnekler (Viola & Jones, 2001)

1.4.8. Hausdorff Mesafesi

İkili görüntülerden elde edilen bölgelerde kontör bilgisiyle istenen bir şeklin araması gerçekleştirilebilir. Referans bir ikili görüntü kontörüyle, görüntü üzerinde kaydırılacak bir pencere içerisindeki kontörlerin karşılaştırılıp, tüm ekran tarandığında en kısa mesafenin bulunduğu pencere istenen bölgeyi buldurabilir. Hausdorff mesafesi (8)'deki formülle ifade edilir. İki farklı kapalı şekli temsil eden noktalar kümesinden, bir şekilden diğerine her bir nokta için diğer şekildeki noktalara olan uzaklık hesaplanır. Bu uzaklıklardan en kısa olanları arasındaki en uzun mesafe $h(A, B)$ olarak ifade edilir. Aynı işlem $h(B, A)$ için yapılır.

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (8)$$

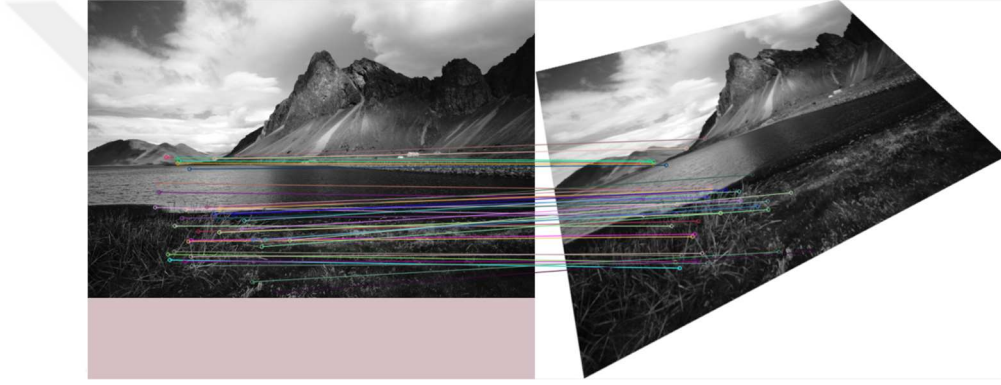
1.4.9. Öznitelik Çıkarma Algoritmaları

Öznitelikler, bilgisayarlı görmede görüntüde tespit edilmek istenen obje veya dokuya ait, onu genel olarak tanımlayabilecek sayısal verilerdir. Bu öznitelikler, kenar, şekil, köşe vb. gibi özelliklerden elde edilebileceği gibi, birtakım özel çekirdeklerin konvolüsyon yöntemiyle görüntü üzerinde dolaştırılmasıyla da elde edilebilir. Elde edilen öznitelikler sınıflandırıcıları eğitmekte kullanılır ve yeni görüntülerde ilgilenilen kısmı tespit etmeye yarar.

Öznitelik çıkarıcı (feature extractor), öznitelik tanımlayıcı (feature descriptor) ve öznitelik eşleştirici (feature matching) algoritmalar birbirinden farklıdır. Öznitelik çıkarıcı algoritmalar, görüntüde anahtar nokta (keypoint) tespiti için yöntemler sunarlar. Bu noktalar genellikle köşelerden seçilir. Öznitelik tanımlayıcı algoritmalar ise, bu anahtar noktaların öznitelik özelliği taşıyabilmesi için mantıksal olarak nasıl tanımlanacağı ile ilgilenirler. Öznitelik eşleştirme yöntemleri ise, tanımlanan öznitelikleri optimize yollarla farklı

görüntülerde birbiriyle eşleştirmek için kullanılır. Literatürde yer edinmiş yaygın kullanılan anahtar nokta bulma, öznitelik tanımlayıcı ve eşleştirici algoritmalar şöyle özetlenebilir;

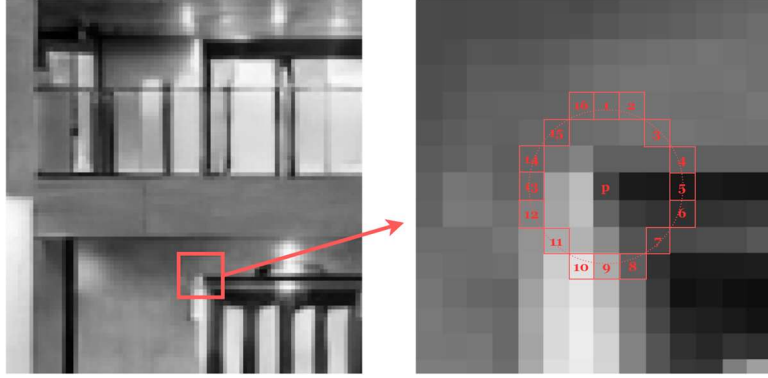
SIFT (Lowe, 2004), 2004 yılında geliştirilmiş bir öznitelik eşleştirme algoritmasıdır. Temel olarak, görüntünün farklı boyutlardan oluşan bir uzayı (scale-space) oluşturulur. Gaussianların farkı (Difference of Gaussians) yöntemiyle öznitelik çıkarılacak noktalar keşfedilir. Bu noktalar etrafındaki piksellerin gradyenlerinden bir histogram üretilir. Bu histogram yardımıyla noktaların döndürme bağımsız (rotation invariant) olması sağlanır. En son olarak da anahtar nokta etrafında öznitelik tanımlamak için 16x16'lık bir pencere içerisinde gradyen şiddetleri ve yönleri hesaplanır. Şekil 10'da SIFT algoritmasının OpenCV ile bir uygulaması görülmektedir.



Şekil 10. SIFT ile anahtar nokta tespiti

SURF (Bay, 2006), 2006 yılında SIFT üzerine yapılan gelişmelerden doğmuş bir öznitelik eşleştirme algoritmasıdır. Hessian matrisleri ve integral görüntüler kullanılarak öznitelik çıkarılacak pikseller tespit edilir. Bu noktalar etrafındaki piksellerde Haar tipinde dalgacıklar (wavelet) uygulanarak, en baskın yön belirlenir. Anahtar noktalar, etrafındaki piksellerin dalgacık sonuçlarının tutulduğu vektörlerle tanımlanır.

FAST (Rosten & Drummond, 2006), 2006 yılında geliştirilmiş bir öznitelik çıkarma algoritmasıdır. Temel mantık olarak, bir pikselin etrafında çember şeklinde yer alan 16 adet piksel değerlendirilir. Bu çember üzerindeki peş peşe piksellerin seçilen noktadan daha aydınlık ya da daha karanlık olmasına göre bu nokta öznitelik çıkarmak için köşe seçilir. Şekil 11'de FAST algoritmasının köşe noktası örneği görülmektedir.

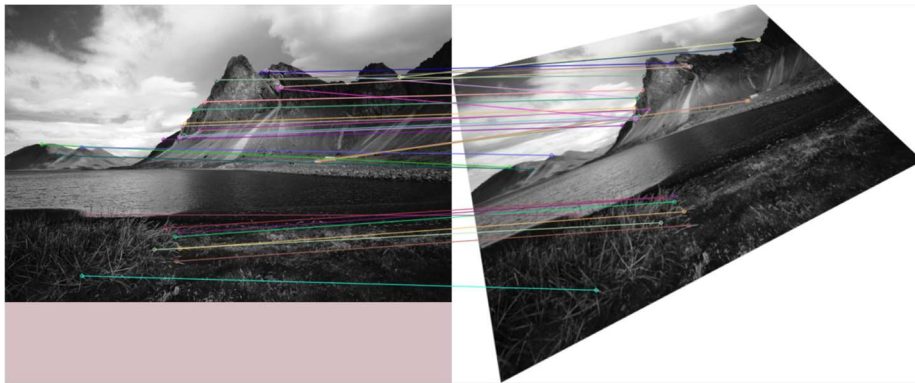


Şekil 11. FAST ile belirlenen köşe örneği (Rosten & Drummond, 2006)

FLANN (Muja & Lowe, 2009), 2009 yılında geliştirilmiş, öznitelik uzayında daha hızlı arama ve anahtar noktaları daha doğru eşleştirme için ağaç yapılarını kullanan bir algoritmalar bütünüdür. SIFT, SURF veya ORB gibi öznitelik çıkarıcı algoritmalar ile beraber kullanılır.

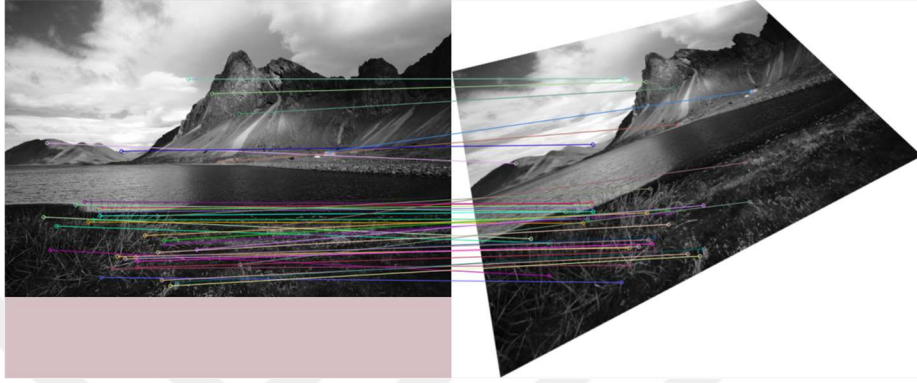
BRIEF (Calonder vd., 2010), 2010 yılında geliştirilmiş bir öznitelik tanımlayıcıdır. Öznitelikler, bellekte kapladıkları alan ve gerektirdikleri hesaplama maliyeti sebebiyle özellikle gömülü sistemlerde yük oluşturabilmektedir. Bu amaçla BRIEF, daha pratik bir yöntem sunmuştur. Bu yöntemde anahtar noktalar parlaklık (intensity) değerine göre birbiriyle karşılaştırılıp, ikili bir öznitelik vektörü elde edilir. Hamming uzaklıklarıyla benzer noktalar bulunmaya çalışılır.

ORB (Rublee vd., 2011), 2011 yılında geliştirilmiş, FAST ve BRIEF algoritmalarının birleşmiş ve iyileştirilmiş halidir. FAST algoritmasına momentler yardımıyla döndürmeden bağımsızlık özelliği eklenmiş, BRIEF algoritmasının oluşturduğu öznitelik vektörlerine de yön uygulanmıştır. Şekil 12’de ORB algoritmasının OpenCV ile bir uygulaması görülmektedir.



Şekil 12. ORB ile anahtar nokta tespiti

BRISK (Leutenegger vd., 2011), 2011 yılında geliştirilmiş, BRIEF gibi ikili (binary) bir öznitelik tanımlayıcıdır. Döndürmeden bağımsızdır. Kendine özgü bir örnekleme deseniyle anahtar noktalar etrafında kısa ve uzun mesafedeki piksellere bakılarak öznitelik oluşturulur ve Hamming uzaklığıyla karşılaştırma yapılır. Şekil 13'te BRISK yönteminin OpenCV ile bir uygulaması görülmektedir.



Şekil 13. BRISK ile anahtar nokta tespiti

KAZE (Alcantarilla vd., 2012), 2012 yılında geliştirilmiş bir öznitelik çıkarma ve tanımlama algoritmasıdır. Boyutlandırmadan bağımsızdır. Görüntülerde Hessian matrislerinin determinantlarıyla öznitelik çıkarımı yapıp, SURF benzeri bir öznitelik tanımlayıcısı kullanır.

AKAZE (Alcantarilla vd., 2013), 2013 yılında geliştirilmiş, KAZE algoritmasının öznitelik çıkarma ve öznitelik tanımlama yöntemlerini modifiye eden bir öznitelik eşleştirme algoritmasıdır.

HOG (Dalal & Triggs, 2005), görüntü işlemede sıklıkla kullanılan bir öznitelik çıkarma algoritmasıdır. Çalışma prensibi basitçe şu şekildedir; görüntünün ilgilenilen kısmında, yüksekliği genişliğinin iki katı olacak şekilde bir alan seçilir. Bu alanda gradyen görüntü hesaplanır. Gradyen görüntü, pikseller arasındaki geçişin yönünü ve şiddetini belirtmeye yarar. Bir pikselin etrafındaki piksel değerlerinin, dikey ve yatay açı yönlerine göre farkları hesaplanır. Bu farkların kareleri alınarak toplanır ve toplam değerın karekökü bulunur. Bu, gradyenin şiddetini (magnitude) belirler. Farkların arctanjantı ise gradyenin açısı olarak bulunur. Her bir piksel, bu şekilde açı ve şiddet şeklinde temsil edildikten sonra, görüntü 8x8'lik alanlara bölünür. Bu 8x8'lik hücreler üzerinde gezilerek, açı ve şiddete göre 9'a bölünmüş bir histogram elde edilir. Tüm görüntüde elde edilen bu histogramlar bir

sınıflandırıcıya verilerek tespit gerçekleştirilir. Şekil 14'te HOG yöntemiyle elde edilen özniteliklerin bir görüntüsü görülmektedir.



Şekil 14. HOG ile elde edilen özniteliklerin örneği

1.5. Bilgisayarlı Görmede Yapay Sinir Ağları ve Değerlendirme Metrikleri

Yapay sinir ağları, temelleri 1950'li yıllara uzanan ve günümüzde en çok kullanılan makine öğrenmesi metodlarından biridir. Temel olarak, beyin yapısında bulunan nöronlardan ilham alınarak, benzer bir yapının matematiksel olarak ifade edilebilen girdiler ve çıktılar olarak tanımlanmasından oluşur. Bu yapının en küçük birimi bir nöronu temsil eden perceptron yapısıdır. Her bir perceptronda, bir ağırlık (weight) değeriyle bir aktivasyon fonksiyonu bulunur. Girdi değerleri bu ağırlık değeriyle çarpılır, toplamı alınır ve aktivasyon fonksiyonuna iletilir. Eşik değerinden yüksek bir değer elde edilmişse, nöron ateşlenmiş olur ve bir sonraki nörona bu bilgiyi iletir. Perceptronlar, eldeki verileri ikili olarak sınıflandırabilirler (0 veya 1 değerlerini üretirler). Birden fazla perceptronun bir araya gelmesiyle bir katman oluşur. Bu katmanların sayısı birden fazla olabilir ve önceki katmanlar sonraki katmanlara bağlanır. Bu şekilde çoklu katman (mutli-layer) perceptron ağları oluşur. Bu ağların özelliği, tek katmanla ayrıştırılamayan verileri sınıflandırabilmesidir. Bu işlem için, geri yayılım (backpropagation) adı verilen, geriye dönük olarak tüm katmanlardaki nöronların ağırlık değerlerinin güncellendiği bir matematiksel formül uygulanır.

Tek katmanlı perceptronda, her bir nörona gelen girdiler (x) nöronun ağırlık değeriyle çarpılıp, bir bias (b) değeri ile toplanır (Formül 9). Tüm x verileri için yapılan bu işlem, bir aktivasyon fonksiyona gönderilir (Formül 10). Bu fonksiyon, sigmoid fonksiyonunda olduğu gibi, genellikle türevi alınabilen bir fonksiyon olur. Bunun sebebi, çok katmanlı ağ yapılarında geri yayılım işleminde ağırlıklar güncellenirken, hata fonksiyonunun türevi

kullanıldığı içindir. Bu aktivasyon fonksiyonundaki sonuç, nöronun ürettiği sonuç olacaktır. Nöronun ağırlık değeri (Formül 11) formülünde olduğu gibi, x verisi, veriye ait hata miktarı ile öğrenme oranının (learning rate, l) çarpımının önceki ağırlık değerine eklenmesiyle bulunur. Bu çıktı değerlerinin hedeflenen değerden ne kadar uzakta olduğunu ölçmek için bir hata fonksiyonu kullanılır (Formül 12). Bu fonksiyonda, çıktıda beklenen değer ile elde edilen değerin farkının karesi alınır, bu ortalama hata karesi (mean squared error) formülüdür. Formül 13, ağırlık değişiminin genel ifadesi, η ise öğrenme oranıdır. Şekil 15'te perceptronun görsel bir temsili bulunmaktadır.

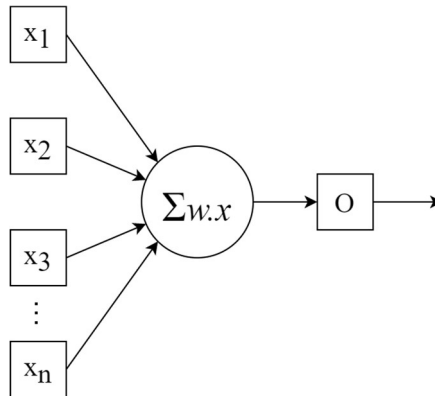
$$z = \sum_{i=1}^n x_i w_i + b \quad (9)$$

$$y = \frac{1}{1 + e^{-z}} \quad (10)$$

$$w' = w + l(d_i - y_i)x_i \quad (11)$$

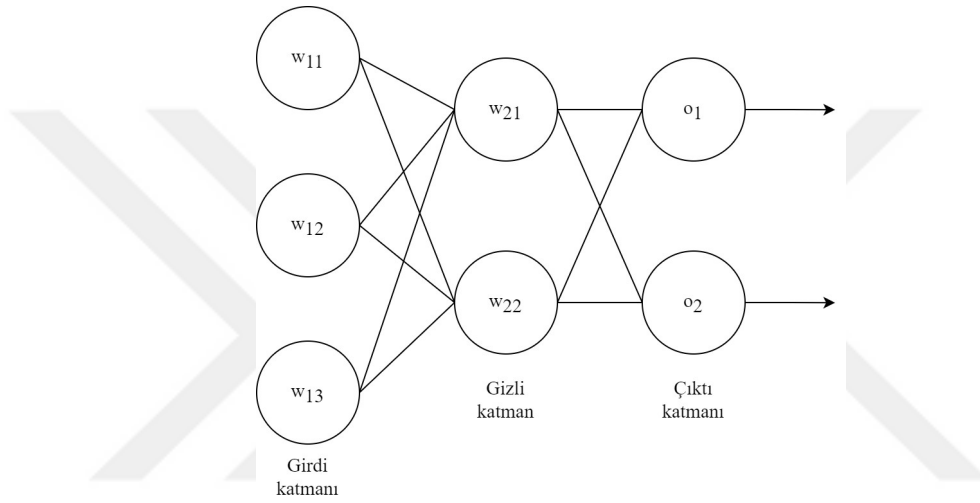
$$E = \frac{1}{2} \sum_{i=1}^m (d_i - y_i)^2 \quad (12)$$

$$\Delta w = \eta(d_i - y_i)f'(y)x_i \quad (13)$$



Şekil 15. Perceptron yapısı: Girdiler (x), ağırlıklar (w), aktivasyon fonksiyonu (o)

Çok katmanlı perceptron ağlarında, bir nöron katmanının ürettiği sonuçlar, diğer nöron katmanına girdi olarak iletilir. Böylece katmanlar istenildiği sayıda artırılabilir. Bu durumda, ağdaki ağırlık değerlerini güncellemek için, gradyan iniş (gradient descent) yöntemi kullanılır. Bu yöntem, hata fonksiyonunun türevinde, minimum noktayı tespit etmek amacıyla kullanılır. Hata fonksiyonunun türevinde bulunan minimum nokta, hatanın en az olduğu nokta olacaktır ve ağdaki ağırlık değerlerinin bu noktaya göre güncellenmesi ağın başarılı olmasını sağlar. Şekil 16’da çok katmanlı bir perceptron ağı yapısı görülmektedir.



Şekil 16. Çok katmanlı perceptron ağı

Ağdaki ağırlıkları güncellemek için, geri yayılım yöntemi kullanılır. Bu yöntem, türevdeki zincir kuralına bağlıdır. Bir nöronun ağırlık matrisinin güncellenmesi, Formül 14’te olduğu gibi hata fonksiyonunun, nöronun ürettiği değere bağlı kısmi türevi ile bulunabilir. (η) burada adım miktarı (step size) olarak sabit bir değerdir. Zincir kuralına göre, Formül 14’teki türev kısmı Formül 15’teki gibi ifade edilebilir. Formül 16’da, Formül 15’teki $\delta E / \delta y_k$ kısmı ifade edilmiştir. Bir i verisi için, hata fonksiyonunun türevi, hatanın karesinin baştaki katsayı olan $\frac{1}{2}$ ile sadeleşmesiyle $(d_k - y_k)$ olacaktır. Sigmoid fonksiyonunun türevi ise $y_k (1 - y_k)$ olacaktır. Formül 16 ve Formül 17’ye göre, ağırlık güncellemesi için gerekli olan değişim, Formül 18’deki gibi özetlenebilir. Çoklu katmanlara sahip bir ağdaki hata fonksiyonunun genel ifadesi ise Formül 19’daki gibi olacaktır.

$$w_i' = w_i - \eta \frac{\delta E}{\delta w_i} \quad (14)$$

$$\frac{\delta E}{\delta w_i} = \frac{\delta E}{\delta y_k} \frac{\delta y_k}{\delta w_i} \quad (15)$$

$$\frac{\delta E}{\delta y_k} = (d_k - y_k)y_k(1 - y_k) \quad (16)$$

$$\frac{\delta y_k}{\delta w_i} = y_k(1 - y_k) \sum_i w_i \delta y_i \quad (17)$$

$$\Delta_i = \eta \delta y_k x_i \quad (18)$$

$$E = \frac{1}{2} \sum_m \sum_i (t_i - o_i)^2 \quad (19)$$

Bilgisayarlı görme alanında, konvolüsyonel yapay sinir ağları (CNN), görüntülerde ilgilenilen bölgenin veya görüntünün tamamının özniteliklerini elde etmeye yarayan mekanizmalar bulunduran, görüntüleri ya da bölgeleri sınıflandırabilmek için oluşturulan, çok katmanlı kompleks yapay sinir ağlarıdır. Oldukça yoğun kullanımı olan ve yeni özellikler eklenerek geliştirilmeye devam eden bu ağlarda bulunan temel kavramlar aşağıdaki gibi özetlenebilir;

Girdi katmanı (Input layer): Görüntünün yatay ve dikeyde yer alan tüm piksellerinin ayrı ayrı olarak birer nörona bağlanarak ağın içerisine sayısal veri olarak ileten katmandır.

Konvolüsyon katmanı (Convolution layer): Görüntü boyutu arttıkça, sonraki katmanlar için artacak nöron sayısı ve işlem gücü, aynı zamanda performans ve doğruluk oranında düşüş yaşatacaktır. Bu yüzden, doğrudan görüntünün pikselleri üzerinde çalışmak yerine, konvolüsyon çekirdekleri sayesinde görüntüden çıkarılacak özniteliklerin sonraki katmanlara iletilmesi tercih edilir. Bu katman, içerisinde bulundurduğu çekirdeklerin girdi katmanındaki sayısal değerlere konvolüsyon yöntemiyle uygulanması sayesinde öznitelik çıkarımı sağlar. Çekirdekler birden fazla sayıda olabilir. RGB görüntüler için, her bir kanala ait özel bir çekirdek bulunabilir. Konvolüsyon işlemi sonrasında, K adet filtre ve N adet renk kanalı için, $K \times N$ kadar iki boyutlu görüntü oluşur. Şekil 17’de konvolüsyon işlemine örnek verilmiştir.

8	20	4	9
10	5	15	14
12	8	3	6
19	7	10	12

0	1	0
1	2	1
0	1	0

63	56
43	45

Şekil 17. Konvolüsyon işlemi: Solda orijinal görüntü, ortada konvolüsyon çekirdeği, sağda üretilen sonuç

Stride: Konvolüsyon katmanında bulunan filtrelerin, girdi görüntüsünün üzerinde kaç piksel farkla kaydırıldığını belirleyen bir parametredir.

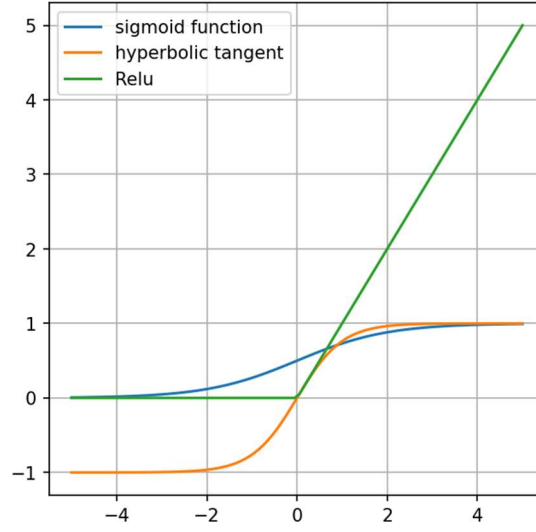
Zero Padding: Görüntünün kenarlarında kalan piksellere konvolüsyon filtrelerinin uygulanabilmesi için, görüntüyü temsil eden iki boyutlu matrisin kenarlarına $N \times N$ kadarlık 0 değerlerinin eklenmesidir. Bu sayede görüntü boyutu küçülmeden konvolüsyon işlemi mümkün olabilir.

Havuz katmanı (Pooling layer): Konvolüsyon sonrası elde edilen iki boyutlu matrisin boyutunu azaltmak için kullanılan bir yöntemdir. Genellikle $N \times N$ boyutlarındaki bir filtre bu matrisin üzerinde gezdirilerek, denk düştüğü tüm hücrelerdeki en büyük sayısal değerler çıktı matrisini oluşturur.

Sıkı katman (Dense layer): Genellikle yapay sinir ağlarının son kısımlarında yer alan bir katmandır. Tüm nöronları, önceki katmanın tüm nöronları ile bağlanmış haldedir. Özniteliklerin sınıflandırılması için gereklidir. Çoklu sınıf sınıflandırması için sınıf sayısı kadar nörona sahip olan bu katman kullanılır.

Aktivasyon fonksiyonu: Bir katmanda bulunan nöronların, kendilerine ulaşan girdi sinyalini hangi eşikten itibaren bir sonraki katmana aktaracağını belirleyen matematiksel fonksiyonlardır. Sınıf sayısına göre farklı türlerde seçilir. Yaygın olarak kullanılanlardan fonksiyonlardan ReLU (Rectified Linear Unit), Sigmoid, Softmax örnek olarak verilebilir. Şekil 18'de yaygın aktivasyon fonksiyonlarından örnekler verilmiştir. Formül 20'de Softmax aktivasyon fonksiyonu gösterilmiştir.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (20)$$



Şekil 18. Aktivasyon fonksiyonu örnekleri

Dropout (Seyreltme): Bazı durumlarda, eğitim yapılan veri setindeki veriler yapay sinir ağını istenmeyen yönde eğitebilir. Eğitim veri setindeki bireysel örnekler üzerinde yüksek başarı sağlayan bir ağ, yeni verilerde başarısız olabilir. Bu da ağı sınıflandırma başarısını düşürür. Bu duruma overfitting denir. Ağ yapısının gereğinden fazla eğitilmemesi, gürültü ve detaylardan daha az etkilenmesi için bilinçli olarak katmanlarda bulunan belirli orandaki rastgele nöronlar hesaplamalarda devre dışı bırakılır. Bu yöntemle Dropout adı verilir. Amacı overfitting durumunu önlemektir.

Validation (Doğrulama): Eğitim boyunca, ağ modelinin test verilerine benzer verilerle test edilmesi işlemidir. Ağı verilerle uygunluğunu, öğrenme durumunu erkenden gözlemleyebilmek için kullanılır.

Batch Normalization (Küme Normalizasyonu): Modern yapay sinir ağlarında bir katman şeklinde bulunan normalizasyon katmanı, ağı daha hızlı eğitilmesini ve daha stabil olmasını sağlar. Bunu da verilere aktivasyondan önce normalizasyon uygulayarak yapar.

Loss Function (Kayıp Fonksiyonu): Ağlardaki ağırlık matrislerinin güncellenmesi için, ağı istenen çıktıdan ne kadar uzakta olduğunu ölçen matematiksel fonksiyonlardır. En çok kullanılanlar arasında, ortalama hata karesi, çapraz entropi (cross entropy) gibi fonksiyonlar bulunur. Yaygın olarak, çoklu sınıflara ait veriler sınıflandırıldığı takdirde kategorik çapraz entropi (categorical cross entropy) kayıp fonksiyonu kullanılır. Formül 21’de kategorik çapraz entropi kayıp fonksiyonu gösterilmiştir.

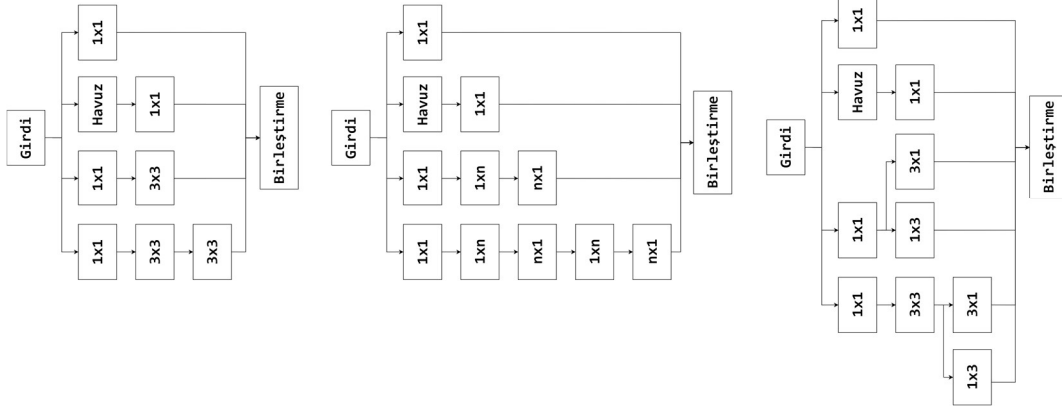
$$L = \sum_{i=1}^n t_i \log(p_i) \quad (21)$$

Optimization (Optimizasyon): Ağın eğitimini yapmak için nöron ağırlıkları eğitim boyunca güncellenir. Bu güncellemeyi yaparken hata fonksiyonu değerini minimuma indirmek için çeşitli optimizasyon yöntemleri uygulanır. Yaygın bazı yöntemlere, Stokastik Gradyan İniş, Adam (Kingma & Ba, 2014) örnek olarak verilebilir. Adam, değişken bir öğrenme oranı (learning rate) sunan, hata fonksiyonunu optimize edici bir algoritmadır. Gradyenlerin momentlerini kullanarak ağı ağırlık matrislerini güncellemeye yarar. Algoritmanın adımları Formül 22’de verilmiştir. Varsayılan olarak parametreler α 0.001, β_1 0.9, β_2 0.999, ϵ ise 10^{-8} olarak önerilmektedir. θ parametre vektörü, α öğrenme oranı, β_1 ve β_2 momentlerin azalma oranı (decay rate), ϵ ise sabit değerdir. m_t ve v_t gradyenlerin birinci ve ikinci momentleridir. Birinci moment ortalama (mean), ikinci moment ise varyans anlamına gelmektedir.

$$\begin{aligned} g_t &= \nabla_{\theta} f_t(\theta_{t-1}) \\ m_t &= \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \\ v_t &= \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \\ \hat{m}_t &= m_t / (1 - \beta_1^t) \\ \hat{v}_t &= v_t / (1 - \beta_2^t) \\ \theta_t &= \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) \end{aligned} \quad (22)$$

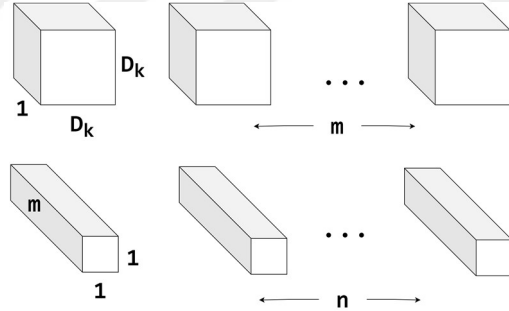
Yaygın olarak kullanılan ve tez çalışmasında da kullanılmış olan bazı konvolüsyonel yapay sinir ağlarına şunlar örnek olarak verilebilir;

Inception-V3 ağı 2016 yılında geliştirilmiş bir CNN’dir (Szegedy vd., 2016). Temel dikkat çeken noktaları şöyle özetlenebilir; konvolüsyon filtrelerin boyutu daha küçük ve asimetrik yapıdadır, bu dizaynı sayesinde daha verimli çalışmaktadır. Örneğin, 5x5’lik bir konvolüsyon yerine 3x3’lük iki adet konvolüsyon tercih edilebilir. Bu 3x3’lük konvolüsyonlar da 3x1 ve 1x3 olarak asimetrik seçilebilir. Böylece özellik çıkarımı çok fazla etkilenmeden parametre sayısı düşürülerek iyi bir performans alınabilir. Şekil 19’da Inception-V3 ağındaki filtrelerden örnekler görülmektedir.



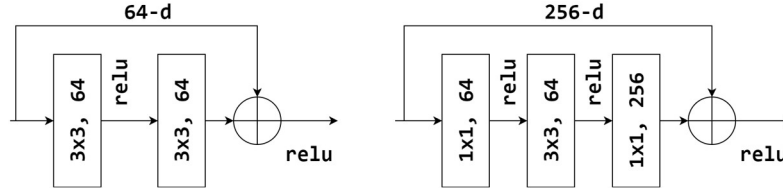
Şekil 19. Inception-V3 ağında kullanılan konvolüsyon filtreleri (Szegedy vd., 2016)

MobileNet 2017 yılında geliştirilmiş bir CNN ağıdır (Howard vd., 2017). Temel özellikleri şöyle sıralanabilir; MobileNet'te her bir kanal için uygulanan tek bir konvolüsyon filtresi bulunmaktadır (Depthwise convolution). Bu konvolüsyon sonucuna, 1x1'lik bir konvolüsyon filtresi daha uygulanmaktadır (Pointwise convolution). Bu katmanlar ağ boyunca tekrarlanarak öznetelik çıkarımı yapmaktadır. MobileNet ağının avantajı, nispeten daha kompakt ve düşük maliyetli olmalarıdır. Dolayısıyla mobil cihazlar için kullanılma imkanları vardır. Şekil 20'de konvolüsyon filtrelerinin örnekleri görülmektedir.



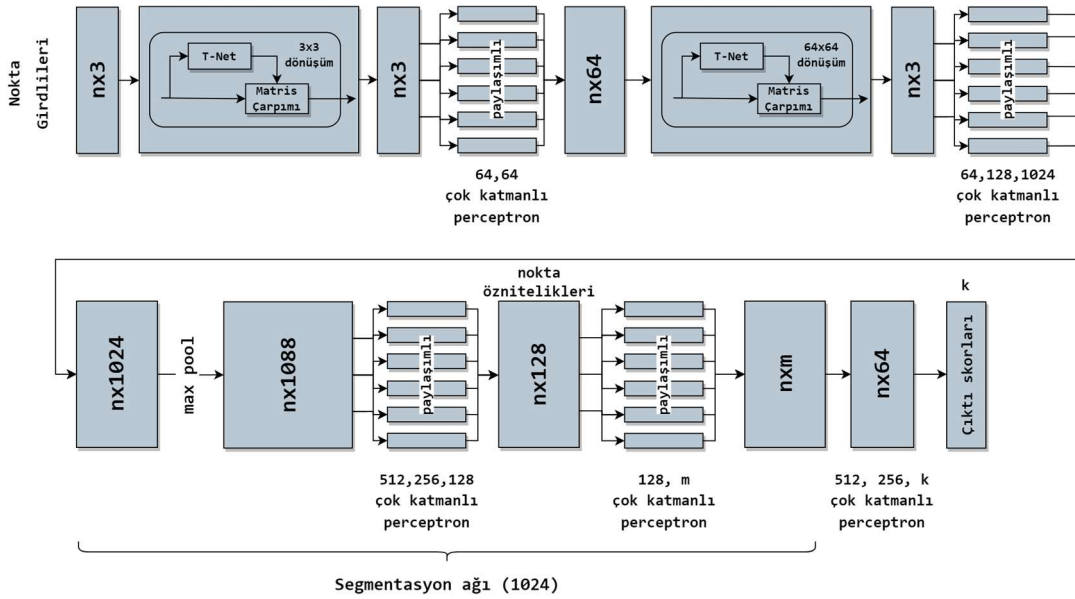
Şekil 20. MobileNet-V1 ağında kullanılan depthwise (üstte) ve pointwise (altta) konvolüsyon filtreleri (Howard vd., 2017)

ResNet, 2016 yılında geliştirilmiş bir CNN ağıdır (He vd., 2016). Temel özellikleri şöyle sıralanabilir; CNN ağlarında görülen temel problemlerden biri, katman sayısı arttıkça ağın parametrelerinin güncellenmesinin zorlaşmasıdır (vanishing gradient). Residual Network (ResNet) ile, birbirine bağlı olmayan katmanlar arasında kestirme yollar (shortcut) tanımlanarak, bu katmanlar arasında etkileşim sağlanmıştır (residual block). Hem girdiler ağa verildiğinde hem de geri yayılım uygulandığında bu kestirme yollar sayesinde katmanlar güncellenmektedir. Böylece katman sayısının artması ağın performansını düşürmemiş olmaktadır. Şekil 21'de ResNet-50 ağlarındaki residual fonksiyon yapısı görülmektedir.



Şekil 21. ResNet-34 (solda) ve ResNet-50 (sağda) ağırları için residual fonksiyonların yapısı (He vd., 2016)

PointNet (Qi vd. 2017), noktalar kümesi verilerini kullanarak sınıflandırma yapabilen, 2017 yılında geliştirilmiş bir yapay sinir ağı modelidir. Derin öğrenme yapısıyla noktalar kümesi verilerinin birlikte kullanılması, giderek yaygınlaşan bir yöntemdir. Derinlik kameralarının bilgisayarlı görmede etkili biçimde kullanılmasıyla beraber, üç boyutlu noktalar kümesi verilerinin işlenmesi, bu verilerin tek başına ya da farklı veri tipleriyle beraber kullanılması söz konusu olmuştur. Noktalar kümesinin özelliği, sıralanmamış, üç boyutlu uzayda bir konuma göre konumlandırılmamış veya geometrik transformasyonu yapılmamış veriler olmasıdır. Dolayısıyla özel bir mimari gerektirmektedir. PointNet ağı da girdi verilerinin tüm olası kombinasyonlarını elde eden bir modül içermektedir. Böylece sıralanmamış veri sorununu çözmektedir. Ayrıca, verinin tüm geometrik dönüşümlerini (transformation) elde etmek için de bir modül içermektedir. PointNet ağı, üç boyutlu noktalar kümesiyle cisimlerin sınıflandırmasını ve segmentasyonunu yapabilmektedir. Şekil 22’de PointNet ağının genel yapısı görülmektedir.

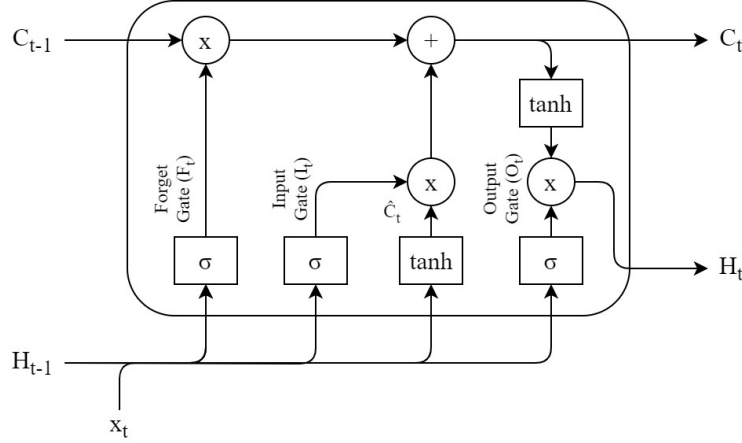


Şekil 22. PointNet ağında bulunan modüllerin yapısı (Qi vd. 2017)

VGG-16 (Visual Geometry Group, VGG) ağı, 2014 yılında geliştirilmiş 16 katmanlı oldukça popüler bir CNN ağıdır (Simonyan & Zisserman, 2014). Konvolüsyon filtreleri olarak 3x3 veya 1x1'lik filtreler kullanılmaktadır. Oldukça büyük olması sebebiyle (138 milyon parametre içermektedir) eğitimi yavaştır. VGG-16'nın iki farklı kullanım türü bulunmaktadır, önceden eğitilmiş bir VGG-16 ağı ile görüntü sınıflandırma yapılabilir, aktarmalı öğrenme (transfer learning) için kullanılabilir veya sıfırdan bir ağ eğitilip görüntü tanıma yapılabilir. Eğitilmiş ağ ile ayrıca, son katmanlar dahil edilmeden öznetelik çıkarımı yapılabilmektedir.

LSTM ağı 1997 yılında geliştirilmiş bir tür RNN (Recurrent Neural Network) ağıdır (Hochreiter & Schmidhuber, 1997). Temel amacı, zamansal bir boyutu bulunan verilerin sınıflandırılmasıdır. LSTM ağlarında hücre deneni yapılar, gelen bilginin mevcut durumu (state) güncelleyip güncellemeyeceğini, bu bilginin ağı çıktısını etkileyip etkilemeyeceğini kararlaştırmak için kapı mekanizmaları bulunur. Temel olarak bir hücre üç bölüme ayrılabilir, unutma kapısı (forget gate), girdi kapısı (input gate), çıktı kapısı (output gate). LSTM hücre yapısı ve aktivasyon fonksiyonları Şekil 23'te görülmektedir. LSTM ağları katman olarak bulunduğu diğer CNN'lerle birleştirilip kullanılabilir. Doğrudan görüntülerin girdi olarak verilmesi uygulanan bir yöntem olmamakla beraber, LSTM ağları belirli bir zaman aralığındaki özneteliği çıkarılmış verileri sınıflandırmak için oldukça etkili bir yöntemdir.

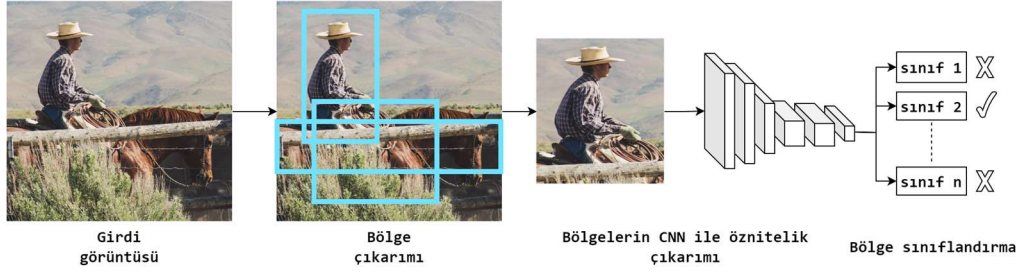
Bir LSTM hücresinin matematiksel olarak ifadesi Formül 23'te görülmektedir. x_t gelen veriyi, H_{t-1} ise bir önceki hücrenin gizli durumunu (hidden state) ifade eder. W ifadeleri ağırlık matrisleridir. C_t ise hücrenin hafıza (memory) durumudur. Girdi kapısı (input gate) I_t , unutma kapısı (forget gate) F_t , çıktı kapısı da O_t olarak ifade edilmiştir. Unutma kapısı, mevcut verinin tutulup tutulmayacağına karar veren yapıdır. Çıktı kapısı ise mevcut verinin bulunulan zaman noktasında çıktıya etkisine karar veren yapıdır. Girdi kapısı ise mevcut verinin hücrenin iç yapısındaki durumuna ne kadar katkı vereceğine karar veren yapıdır. Bu kapılar sigmoid fonksiyonuyla beraber kullanılır. Hücrenin hafıza durumu (C_t) ve gizli durumu (H_t), Hadamart çarpımı adı verilen matris işlemiyle ($\odot$) belirlenir.



Şekil 23. LSTM hücre yapısı

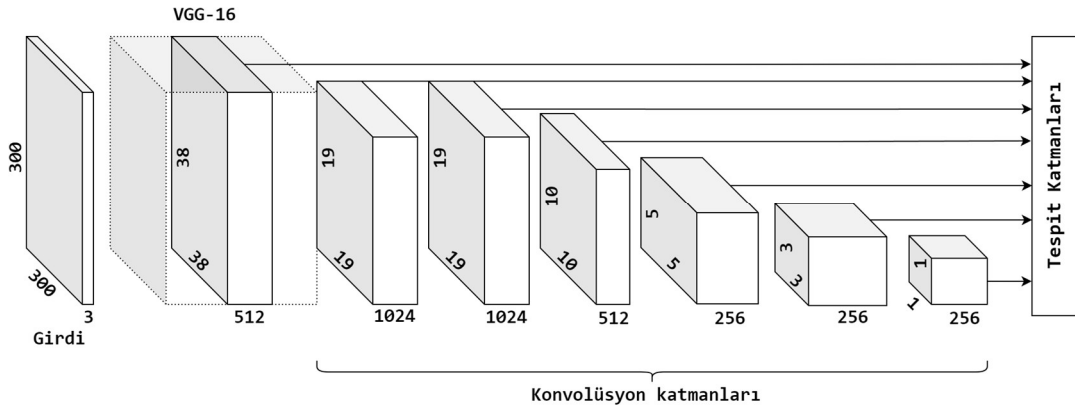
$$\begin{aligned}
 I_t &= \sigma(x_t W_{xi} + H_{t-1} W_{hi}) \\
 F_t &= \sigma(x_t W_{xf} + H_{t-1} W_{hf}) \\
 O_t &= \sigma(x_t W_{xo} + H_{t-1} W_{ho}) \\
 \hat{C}_t &= \tanh(x_t W_{xc} + H_{t-1} W_{hc}) \\
 C_t &= F_t \odot C_{t-1} + I_t \odot \hat{C}_t \\
 H_t &= O_t \odot \tanh(C_t)
 \end{aligned} \tag{23}$$

Hemen her türlü makine öğrenmesi işlemi için kullanılan yapay sinir ağı modelleri, nesne tanıma için de yaygın olarak kullanılmaktadır. Nesne tanımda en çok tercih edilen derin öğrenme ağları Region-Based Convolutional Neural Network (R-CNN) (Girshick vd., 2014) ve Single Shot Detector (SSD) (Liu vd., 2016) olmaktadır. R-CNN’de, görüntüler üzerinde ilgilenilen nesnenin olabileceği kısımları işaretleyip, daha önceden eğitilmiş bir CNN’e sunulur. CNN, çıktı olarak bir öznitelik vektörü üretir ve bu vektör sınıflandırılmak üzere SVM’e sunulur. Bölgelerin bulunması için, seçici arama (selective search) (Felzenszwalb & Huttenlocher, 2004) algoritmasından faydalanılır. Bu algorithmada, bölütlenmiş bir görüntüde tüm kısımlar renk, doku, boyut ve dolgu özelliklerine bakılarak birleştirilir. Ancak, bu bölgeler sayıca fazla olduğunda işleyiş oldukça yavaş olmaktadır. Bununla baş etmek için Fast R-CNN (Girshick, 2015) ve Faster R-CNN (Ren vd., 2015) geliştirilmiştir. Fast R-CNN’de bölgeler ayrı ayrı CNN’e sunulmak yerine tüm görüntü sunulur, bu görüntünün konvolüsyon yardımıyla öznitelik haritası çıkarılır. Bu harita üzerinde bölge araması yapılır. Faster R-CNN’de ise, maliyetli seçici arama algoritması yerine bölge seçme işini de yapan Region Proposal Network (RPN) adı verilen ayrı bir ağ bulunur. Şekil 24’te, R-CNN ağının çalışma mantığı özetlenmiştir.

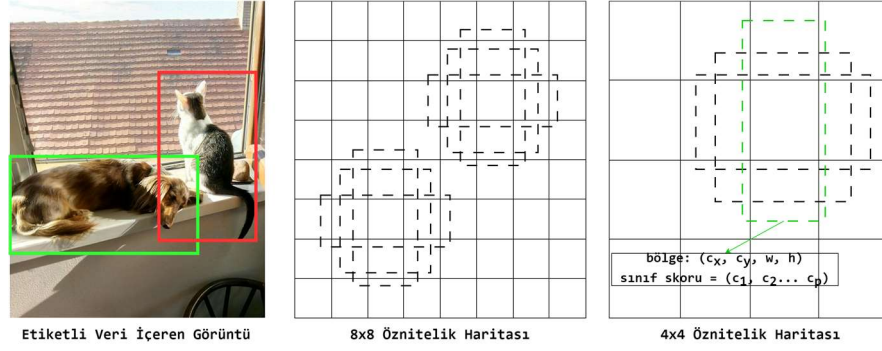


Şekil 24. R-CNN çalışma prensibi (Girshick vd., 2014)

SSD ağlarında, görüntüye farklı boyutta konvolüsyon filtreleri uygulanarak öz nitelik çıkarımı işlemi yapılır. Başlangıçta öz nitelik uzayı çıkarmak için VGG-16 ağından faydalanılır. Bu işlem sonucunda, görüntü farklı boyutlarda hücrelerden oluşan ızgara şeklinde bölünmüş öz nitelik uzayları elde edilir. Bu öz nitelik uzayında, tekrar konvolüsyon çekirdekleri uygulanır ve her bir bölge için farklı boyutlarda önceden tanımlanmış kutular yerleştirilerek, kutunun denk düştüğü alanda sınıf olasılıkları hesaplanır. Bu kutuların boyutları varsayılan olarak seçilir. Dolayısıyla bu kutuların seçimi ağın başarısında önem taşır. Kutuların ne kadar doğru yerleştirildiği ise IoU isimli (Intersection over Union) bir metrik ile ölçülür. Kutulardan en düşük skorlu olanları, negatif örnekler olarak Hard Negative Mining adı verilen yöntemle ağı eğitmek için kullanılır. Şekil 25'te SSD ağının genel yapısı görülmektedir. Şekil 26'da SSD ağının çalışma mantığı özetlenmiştir.



Şekil 25. SSD ağının yapısı (Liu vd., 2016)



Şekil 26. SSD ağının farklı boyutlarda öznitelik arama ve çevreleyen kutu mekanizması (Liu vd., 2016)

YOLO (You Only Look Once) ağları, obje tanıma yöntemleri içerisinde en ileri teknoloji sayılabilecek bir yöntem haline gelmiştir. Hem gerçek zamanlı olarak performansının üst düzey olması, hem de doğruluk oranının yüksek olması sebebiyle yaygınlıkla kullanılmaktadır. İlk versiyonu (Redmon vd., 2016) 2016 yılında geliştirilmiş olan bu ağ, 8. versiyonuna (URL-8, 2023) gelmiş bulunmaktadır.

YOLO ağlarının çalışma prensibi SSD ağlarının çalışma prensibi gibidir. Hücrelere bölünmüş bir görüntü üzerinde, her bir hücrede bulunan objelerin olasılıkları ve bu objelere atanan çevreleyen kutuların (bounding box) merkezleri ve boyutları bulunur. Daha sonra ağ, bulunan obje için çizdirilen farklı boyutlardaki kutulardan en yüksek IoU (objeyle en yüksek kesişimi olan) skorlu olanını tespit etmeye çalışır. YOLO ağının 8. versiyonu, obje tanımanın yanı sıra, segmentasyon ve poz tahmini de gerçekleştirebilmektedir.

YOLO ve benzeri obje tanıma ağlarında genellikle üç yapı bulunur. Bunlar backbone (omurga), neck (boyun) ve head (kafa) isimlendirmelerini alır. Omurga kısmı, konvolüsyonel bir yapay sinir ağı olup, farklı hücre boyutları için öznitelikleri çıkaran kısımdır. Kafa kısmı ise, elde bulunan tüm bilgilerden yola çıkarak objenin konumunu tespit eden özel bir ağ kısmıdır. Boyun kısmı ise, omurga ve kafa arasında bağlantı kuran, seviyelerdeki öznitelikleri birleştiren ağ kısmıdır. YOLO-8 ağının yapısı da bu katmanlardan içermektedir. (URL-9, 2023)'te YOLO ağının detaylı yapısı görülebilir.

Görüntü işlemede obje tespiti ya da görüntü sınıflandırmada, kullanılan metodun başarısını ölçmek için kullanılan matrislere hata matrisi (confusion matrix) adı verilir. Bu iki boyutlu matrislerin eksenleri, sınıflandırılan test verilerine ait sınıf etiketlerini ve metodun ilgili veri örneğine karşı tahmin ettiği sınıfı gösterir. Hata matrislerinden elde edilebilen kesinlik (precision), duyarlılık (recall / sensitivity), özgüllük (specificity), F1 ve ROC-AUC (Alıcı İşletim Karakteristiği – Eğri Altı Alan, Receiver Operating Characteristic – Area

Under Curve) istatistikleri modelin başarısını anlamak için kullanılır. Bu değerlerin her biri 0-1 aralığındadır. Her bir değer için, değer 1'e ne kadar yakınsa ideal modele o kadar yakın kabul edilebilir. Bu değerlerin ifade ettiği kavramların anlaşılması için, doğru negatif (DN), doğru pozitif (DP), yanlış negatif (YN), yanlış pozitif (YP) kavramlarının anlaşılması gerekir. Doğru negatif, modelin verilen örneğin ait olmadığı sınıfları doğru tahmin etmesidir. Doğru pozitif, modelin verilen örneğe ait sınıfı doğru tahmin etmesidir. Yanlış negatif, modelin verilen örneğe ait olmadığını tahmin ettiği ve yanlış olduğu sınıflardır. Yanlış pozitif, modelin verilen örneğe ait sınıfı yanlış tahmin etmesidir. Bu kavramların hata matrisinde ifade ettiği alanlar Şekil 27'de gösterilmiştir.

- Kesinlik: Modelde doğru pozitif tahmin edilen örnek sayısının, tüm pozitif tahmin edilen örneklere oranıdır. Precision olarak adlandırılır. Formül 24'teki gibi hesaplanır.

$$Kesinlik = \frac{Doğru Pozitif}{Doğru Pozitif + Yanlış Pozitif} \quad (24)$$

- Duyarlılık: Modelde doğru pozitif tahmin edilen örnek sayısının, tüm pozitif örneklere oranıdır. Recall olarak da adlandırılır. Formül 25'teki gibi hesaplanır.

$$Duyarlılık = \frac{Doğru Pozitif}{Doğru Pozitif + Yanlış Negatif} \quad (25)$$

- Özgüllük: Modelde doğru negatif tahmin edilen örnek sayısının, tüm negatif örneklere oranıdır. Formül 26'daki gibi hesaplanır.

$$Özgüllük = \frac{Doğru Negatif}{Doğru Negatif + Yanlış Pozitif} \quad (26)$$

- F1: Kesinlik ve Duyarlılık değerlerinin harmonik ortalamasıdır. Formül 27'deki gibi hesaplanır. Makro ve Mikro F1 skoru, çok sınıflı sınıflandırmada her sınıf için elde edilen değerlerin ortalama hesaplanma biçimini ifade eder.

$$F1 = \frac{2 \times \text{Kesinlik} \times \text{Duyarlılık}}{\text{Kesinlik} + \text{Duyarlılık}} \quad (27)$$

- ROC-AUC: Alıcı İşletim Karakteristiği (ROC) eğrisi, doğru pozitif oranının (True Positive Rate, TPR), yanlış pozitif oranına (False Positive Rate, FPR) oranının çizildiği eğridir. Doğru pozitif oranı duyarlılık ifadesiyle aynıdır. Yanlış pozitif oranı, Formül 28 ile elde edilir. ROC eğrisi altında kalan bölgenin alanı AUC skoru (Area Under Curve) olarak adlandırılır. Bu değer ne kadar yüksekse, model ideale o kadar yakın kabul edilir.

$$\text{Yanlış Pozitif Oranı} = \frac{\text{Yanlış Pozitif}}{\text{Yanlış Pozitif} + \text{Doğru Negatif}} \quad (28)$$

	DN	YP	DN
Veri etiketi	YN	DP	YN
	DN	YP	DN
	Tahmin		

Şekil 27. Çok sınıflı sınıflandırmalarda doğru negatif (DN), yanlış negatif (YN), doğru pozitif (DP), yanlış pozitif (YP) bölgelerinin hata matrisindeki temsili

2. YAPILAN ÇALIŞMALAR

Bu çalışmada, bilgisayarla görmeye dayalı olarak DSÖ tarafından da önerilen prosedürel el yıkama hareketlerinin tespiti ve denetimi amaçlanmıştır. Bu amaçla ilk aşama olarak elin tespit edilmesi gerekmektedir. El tespiti için her duruma uygun düşen ve yüksek doğrulukta çözüm sağlayan kesin bir yöntem bulunmamaktadır. Literatürdeki birçok çalışmada el hareketleri algılanırken, el ile ilgili birtakım ön kabuller yapılarak sonuçlar elde edilmektedir. Örneğin, kimi çalışmalar renkli görüntülerde deri renginden el bölgesi tespiti yapmaktadır. Ancak bu yaklaşım, ışıklandırmanın değiştiği ortamlarda veya kişinin deri renginin farklı olduğu durumlarda hatalı sonuç üretmektedir. Bir başka yaklaşım olarak, derinlik görüntülerinde kameraya en yakın cismin eller olduğu kabul edilmektedir. Bu yöntemde de kamera karşısında el hareketleri algılanırken, herhangi bir cismin kameraya ellerden daha çok yaklaşması, uygulanan algoritmaların hatalı sonuç üretmesine sebep olabilmektedir. El tespitinde makine öğrenmesi algoritmaları ile başarılı sonuçlar elde edilmektedir. Elleri, yüz tanımda kullanılan Viola-Jones algoritması gibi önceden belirlenmiş sınırlı öznelilikler kullanarak tespit etmek zordur. Çünkü, el üzerinde bulunan hareketli parmaklar, farklı eksenlerde hareket eden bilekler, elleri birçok farklı şekle sokmaktadır. Ayrıca, ışıklandırma ve deri rengi farklılıkları da durumu daha karmaşık hale getirmektedir. Yüksek miktarda el görüntüsü içeren çeşitli veri setleriyle eğitilen farklı makine öğrenmesi sınıflandırıcıları, yüksek doğruluk oranında gerçek zamanlı olarak herhangi bir görüntüdeki elleri tespit edebilmektedir. Başarılı bir el tespiti yöntemi, el hareketi tanımayı da kolaylaştıracaktır. El bölgesinden elde edilecek öznelilikler, çeşitli makine öğrenmesi sınıflandırıcılarını eğitmekte kullanılıp hareket tespitini sağlamaktadır.

Bu bölümde, bu tez çalışmasında, çeşitli temel görüntü işleme metotları ve makine öğrenmesi teknikleri içeren el tespiti yöntemleri sunulmuştur. Daha sonra, tezde izlenen el tespiti yöntemi ve el hareketlerini sınıflandırmak kullanılan farklı veri modelleri ve derin ağ modelleri detaylı olarak anlatılmıştır.

2.1. Kullanılan Yazılım Araçları

Tez sürecinde kullanılan bilgisayar, programlama dili, kütüphaneler ve yazılım araçlarını kısaca bu başlıkta özetlenmiştir.

Python; yüksek seviyeli, nesne yönelimli, yorumlanan bir programlama dilidir (URL-10, 2023). Makine öğrenmesi başta olmak üzere çeşitli alanlarda barındırdığı birçok kütüphaneyle çok kullanışlı bir dildir. Yapay zeka, web geliştirme, oyun programlama, veri analizi gibi alanlarda yaygın olarak kullanılmaktadır. Guido van Rossum tarafından geliştirilmiş ve ilk olarak 1991 yılında kullanıma sunulmuştur. Tez çalışması uygulamaları Python dili ile kodlanmıştır.

OpenCV; 1999 yılında kullanıma sunulan, temel ve ileri düzey gerçek zamanlı görüntü işleme imkanları sunan, çok platformlu bir açık kaynak kodlu kütüphanedir (URL-11, 2023). Ayrıca, bünyesinde çeşitli makine öğrenmesi algoritmaları da sunmaktadır. C++ ile yazılmış olup, Python, Java, MATLAB dillerinde kullanılabilir.

PyKinect2; Kinect 2 kamerasının yazılım geliştirme kitinin (SDK) Python modülüdür (URL-12, 2023). Python kütüphanesi olarak Python dilinde Kinect kamerasına ve özelliklerine ulaşma imkanı sağlamaktadır. Renk, derinlik ve iskelet bilgisine ulaşmayı desteklemektedir.

TensorFlow; Google tarafından geliştirilen, açık kaynak kodlu bir makine öğrenmesi kütüphanesidir (URL-13, 2023). İlk kez 2015 yılında kullanıma sunulmuştur. Birçok farklı platform için makine öğrenmesi altyapısı sunan, Python, Cuda ve C++ ile yazılmış açık kaynak kodlu bir uygulama programlama arayüzüdür (Application Programming Interface, API). Python, C, C++, Go, Java, JavaScript ve Swift dillerinde kullanılabilir.

Keras; Python tabanlı bir derin öğrenme kütüphanesi olup, TensorFlow altyapısını kullanan ve derin öğrenme yöntemlerini pratik ve kişiye özel şekilde kullanabilmeyi sağlayan bir uygulama programlama arayüzüdür (URL-14, 2023). Derin öğrenme modelleri geliştirmede büyük kolaylık sağlamaktadır.

Ultralytics (URL-8, 2023); PyTorch (URL-15, 2023) tabanlı açık kaynak kodlu makine öğrenmesi kütüphaneleri sunan bir şirkettir. Özellikle nesne takibi için YOLO (You Only Look Once) yapay sinir ağlarını geliştirmektedir.

Blender; üç boyutlu modelleme ve animasyon yazılımıdır (URL-16, 2023). Oyun ve film sektöründe yoğun olarak kullanılan bu yazılım, Windows, Linux ve Mac işletim sistemlerinde kullanılabilir. Blender, C ve C++ dili kullanılarak programlanmıştır.

Temelleri, 1994 yılında Hollanda’da bulunan bir animasyon stüdyosu olan Neo Geo’nun kendi animasyon yazılımını geliştirmeye başlamasıyla atılmıştır. 1998 yılında ücretsiz kullanıma açılmıştır. Ton Roosendaal, bu ücretsiz dağıtılan yazılımı temel alarak, 2002 yılında Blender Foundation’u kurmuş, açık kaynak kodlu Blender projesini başlatmıştır. Günümüzde halen aktif olarak geliştirilmektedir. Blender, kendi içerisinde modelleme, heykeltraşlık, render motoru, görsel efektler, simülasyon, animasyon ve Python script desteği içermektedir. Tez çalışmasında Blender, sentetik veri setini görüntülerini oluşturmada kullanılmıştır.

GIMP (GNU Image Manipulation Program); 1995 yılında Berkeley üniversitesinde Spencer Kimball ve Peter Mattis tarafından geliştirilmeye başlanmış, ücretsiz açık kaynak kodlu bir görüntü işleme yazılımıdır (URL-17, 2023). C dilinde kodlanmış olup 1996 yılında kullanıma sunulmuştur. Katmanlı çalışma sistemi kullanılarak iki boyutlu raster görüntüler istenen şekilde modifiye edilebilmekte, efektlerden geçirilebilmekte, çizim yapılabilmektedir. Aktif bir biçimde geliştirilmesi devam eden bu yazılım, Windows, Linux ve Mac işletim sistemlerinde kullanılabilir. Tez çalışmasında GIMP, sentetik veri setinde dokuları oluşturmak için kullanılmıştır.

Makehuman; ücretsiz ve açık kaynak kodlu bir insan figürü modelleme yazılımıdır (URL-18, 2023). Temelleri 1999 yılına dayanmaktadır. İlk olarak Blender için bir eklenti olarak tasarlanmıştır. Python dilinde geliştirilmiş olup, Windows, Linux ve Mac işletim sistemlerinde çalışabilmektedir. Bu yazılım ile, gerçekçi insan modelleri üretilebilmekte, karakterlerin tüm fiziksel özellikleri (boy, kilo, uzuvların niteliği, dokular, poz, kıyafetler vb.) 3d kesintisiz dönüşüm olanağı sayesinde kolay bir arayüz ile ayarlanabilmektedir. Tez çalışmasında Makehuman, sentetik veri setinde karakterleri oluşturmak için kullanılmıştır.

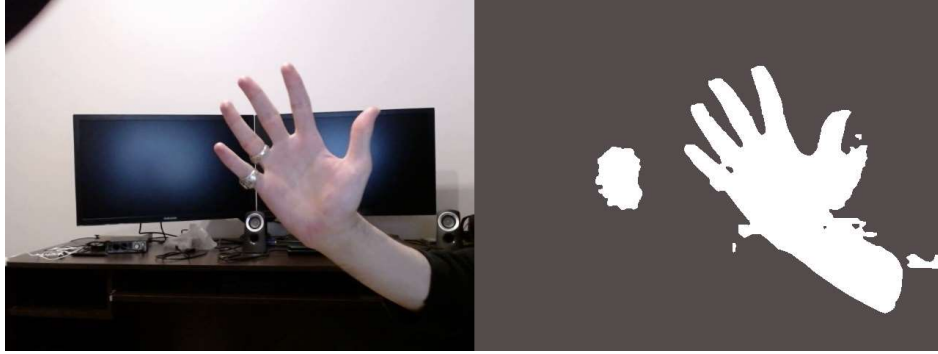
Scikit-Learn; ücretsiz ve açık kaynak kodlu bir makine öğrenmesi kütüphanesidir (URL-19, 2023). İlk olarak 2007 yılında kullanıma sunulmuştur. C, C++, Python dillerinde geliştirilmiş ve Python diliyle kullanılmak üzere tasarlanmıştır.

Tez süreci boyunca ana bilgisayar olarak Windows 10 64 bit işletim sistemi, Intel i5-3230M işlemci, AMD HD 7500M/7600M ekran kartı 16 GB DDR3 belleğe sahip olan bir laptop kullanılmıştır.

2.2. El Tanıma Çalışmaları

2.2.1. Arka Plan Çıkarımı ile El Tanıma

Görüntülerde hareketli nesnelerin algılanması için kullanılabilecek basit yöntemlerden birisi de arka plan çıkarımıdır. Arka plan çıkarımında, önceden tanımlanmış değişmeyen bir arka plan görüntüsü üzerinde hareket eden bir nesne varsa, o anki görüntüyle arka plan görüntüsü karşılaştırılarak değişiklik gösteren piksellerin nesne olduğuna karar verilebilir. Basit, etkili ve işlem yükü az olan bir yöntemdir. Ancak, arka planın veya onu etkileyebilecek ışıklandırmanın değişken olduğu durumlarda hatalı sonuçlar üretebilir. OpenCV kütüphanesi üzerinde, Gaussian Karışım Modelleri temelli MOG, MOG2 ve GMG isimli farklı yöntemlerle arka plan çıkarımı yapılabilir (URL-6, 2023). Şekil 28’de çerçeve farkı ile el bölgesinin çıkarımı, Şekil 29’da MOG2 arka plan çıkarıcı ile el bölgesinin çıkarımı görülmektedir.



Şekil 28. Çerçeve farkı ile arka plan çıkarımı örneği

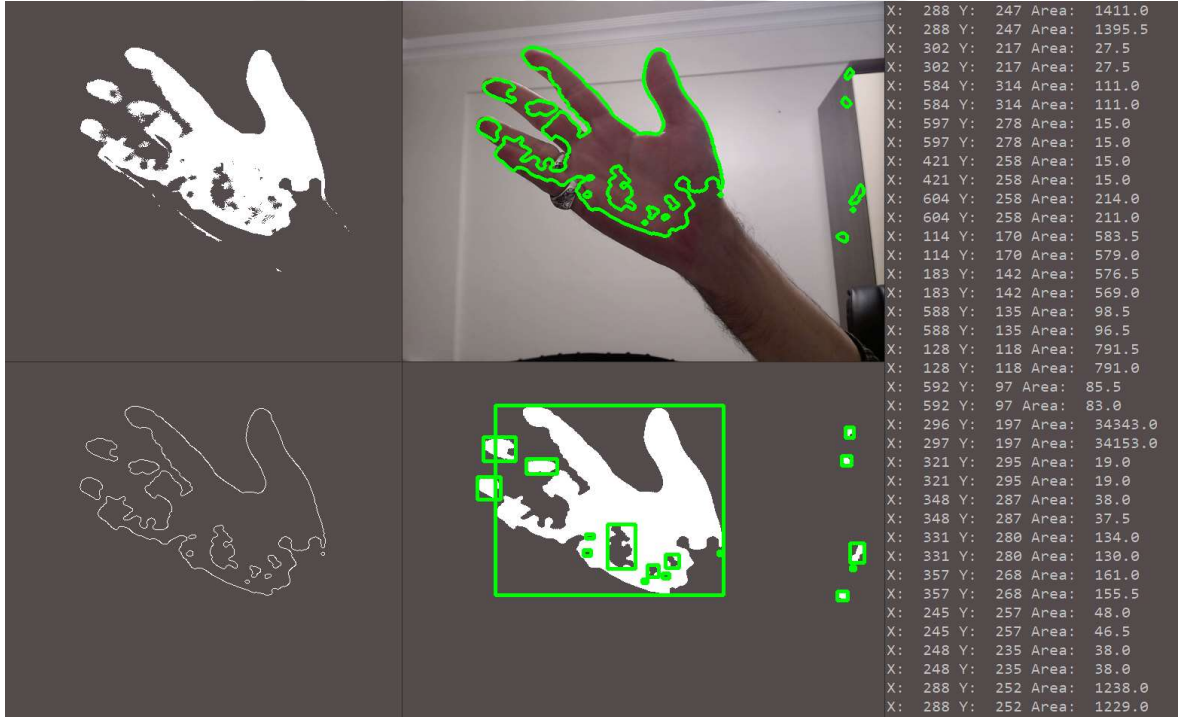


Şekil 29. MOG2 yöntemi ile arka plan çıkarımı

2.2.2. Deri Rengine Dayalı El Tanıma

Renk aralığı bazlı el tanıma için, kameradan gerçek zamanlı elde edilen görüntüler HSV renk uzayında Hue (renk tonu), Saturation (doygunluk), Value (değer) değerleri için (108, 23, 82) ile (179, 255, 255) aralığında deri rengi olarak tanımlanmıştır. OpenCV’de Hue değeri 0-179, Saturation değeri 0-255 ve Value değeri 0-255 aralığında değer almaktadır. Şekil 30’da deri rengi tespit edilen bölgeler içerisinde en büyük kontöre sahip olan bölgenin merkezine ait koordinatlar, bu bölgenin alanı, alanı çevreleyen kontör bilgisi OpenCV kütüphanesi fonksiyonlarıyla yazdırılmıştır. Renk aralığıyla el tespit etme yöntemi, ortamların arka plan, ışık ve kişilerin deri rengi farklılıkları sebebiyle tek başına yetersiz kalmaktadır. Formül 29’da eşikleme için genel ifadesi görülmektedir.

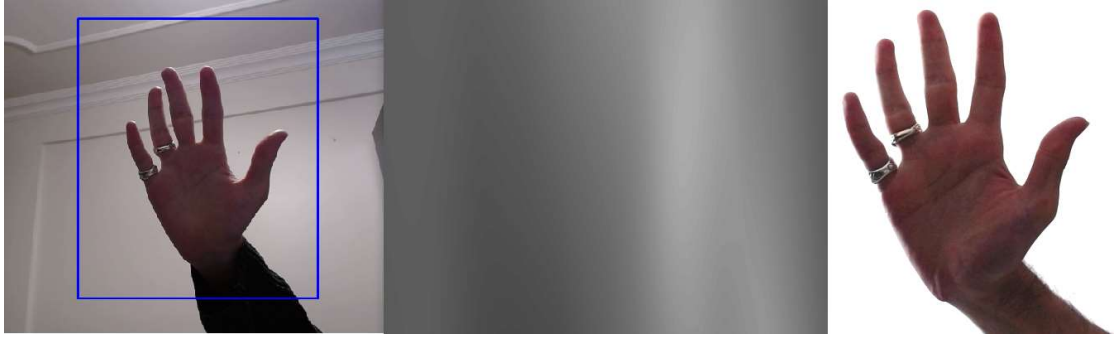
$$\left(\begin{array}{ll} e\check{s}ik_1 \geq pikselde\check{g}eri \geq e\check{s}ik_2 \rightarrow de\check{g}er: 255 \\ pikselde\check{g}eri < e\check{s}ik_2 \rightarrow de\check{g}er: 0 \\ pikselde\check{g}eri > e\check{s}ik_1 \rightarrow de\check{g}er: 0 \end{array} \right) \quad (29)$$



Şekil 30. Deri rengi aralığıyla el tanıma

2.2.3. Şablon Eşleştirmeye Dayalı El Tanıma

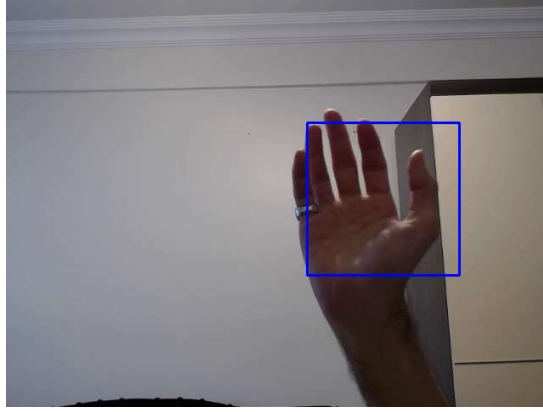
Şekil 31’de tez sürecinde gerçek zamanlı kamera görüntüsünde şablon tanıma kullanılarak tespit edilen bir el örneği görülmektedir. Ortadaki gri düzey görüntüde şablon eşleştirme algoritması sonucu en parlak rengi alan bölge, solda kamera görüntüsünde tespit edilen el bölgesine denk düşmektedir. Sağda ise eşleştirme için kullanılan şablon görülmektedir. Şablon eşleştirmede en iyi sonuçlar, boyutu ve renkleri sabit bir objeye ait tanıma işlemlerinde elde edilebilir. Cismin şablondan farklı açılarda olması ve renginin değişmesi doğruluk oranını düşürebilir. Birden fazla hareket içerecek olan ve hareketlerin devamlı olduğu el hareketi tanıma probleminde şablon eşleştirme kullanılırsa, hareket miktarı, hareketin ve ellerin varyasyonları kadar şablonun her görüntü çerçevesinde gerçek zamanda aramasının yapılması gerekir. Bu da işlem maliyeti yüksek ve pratik olmayan bir çözüm olup, şablon eşleştirmeyi hareket tanımda tercih edilmeyen bir yöntem yapmaktadır.



Şekil 31. Şablon eşleştirme örneği

2.2.4. Viola-Jones Algoritması ile El Tanıma

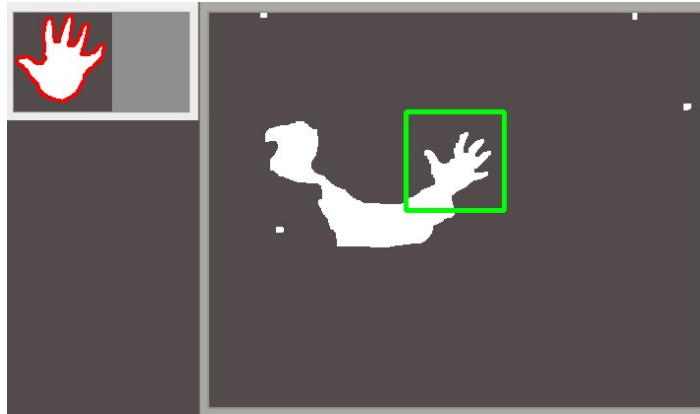
Yaygın olarak kullanılan nesne tanıma algoritmalarından biri olan Viola-Jones algoritması, el tanıma için de kullanılabilir. Bunun için, tanınmak istenen el şeklinin bulunduğu-bulunmadığı pozitif-negatif görüntüler eğitim veri seti olarak kullanılmaktadır. Viola – Jones algoritmasıyla bu görüntüler ile, OpenCV kütüphanesi yardımıyla bir Haar sınıflandırıcı eğitilir. Eğitim sonunda elde edilen XML (Extensible Markup Language) dosyası ile istenen görüntü üzerinde tanıma yapılabilir. Daha önceden eğitilmiş bir Haar cascade (URL-20, 2023) ile avuç tanınması yapan bir örnek uygulama Şekil 32’de görülmektedir.



Şekil 32. Haar cascade ile avuç bölgesi belirleme

2.2.5. Hausdorff Mesafesi ile El Tanıma

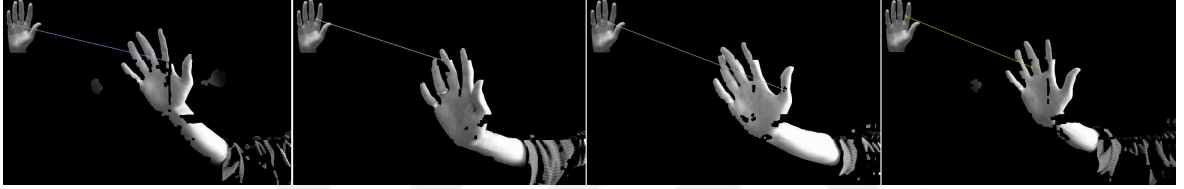
Hausdorff mesafesi ile görüntü işlemede cisimlere ait kontörler arasında benzerlik arama performans açısından maliyetli bir uygulamadır. Tez çalışmasında, kayan pencere yöntemiyle gerçek zamanlı ikili görüntü üzerinde bir el şekli bulunmaya çalışılmıştır. Temel olarak, boyutu aranan şablona uygun bir kayan pencere ikili gerçek zamanlı bir görüntü üzerinde kaydırılmıştır. Hausdorff mesafesinin en az hesaplandığı pencere, görüntü üzerinde çizdirilmiştir. Şekil 33'te uygulamadan bir ekran örneği görülmektedir. Solda önceden tanımlanmış el şekli ve kontörü, sağda ise ikili gerçek zamanlı görüntü üzerinde el şekline en yakın bölgenin tespiti görülmektedir.



Şekil 33. Kayan pencere yöntemiyle Hausdorff mesafesi ile el tespiti

2.2.6. Öznitelik Çıkarma Algoritmaları ile El Tanıma

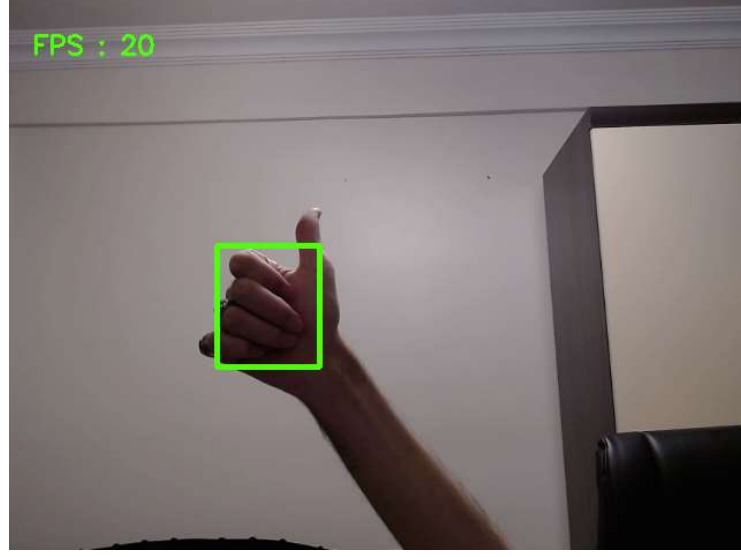
Öznitelik çıkarma ve eşleştirme algoritmalarından AKAZE (Alcantarilla vd., 2013), BRISK (Leutenegger vd., 2011), FAST (Rosten & Drummond, 2006), ve ORB (Rublee vd., 2011) yöntemleri, gerçek zamanlı görüntülerde önceden oluşturulmuş gri düzey bir el görüntüyle eşleştirme başarısı için test edilmiştir. Algoritmalar, gri düzey görüntüler üzerinde çalışmakta olup, renkli görüntüler üzerinde basit düzeyde arka plan çıkarımı yapılarak elde edilmiş gri düzey görüntülere uygulanmıştır. Avuç içi gözüken şekilde bir el görüntüsüne ait belirlenen noktaların eşleştirilmesinde bu yöntemlerin yetersiz kaldığı görülmüştür. Dinamik el hareketlerinin, sürekli değişken olması sebebiyle daha da başarısız sonuçlar üreteceği öngörülebilir. Şekil 34'te uygulamalardan bir örnek görülmektedir.



Şekil 34. Öznitelik çıkarma algoritmalarıyla gri düzey görüntülerde el tanıma. Soldan sağa; BRISK, ORB, FAST, AKAZE

2.2.7. Yapay Sinir Ağlarıyla El Tanıma

TensorFlow kütüphanesinde, obje tanıma için oluşturulmuş bir API (Application Programming Interface) mevcuttur (URL-21, 2023). Bu API içerisinde, önceden oluşturulmuş, obje tanıma en ileri teknoloji sayılan yapay sinir ağ modelleri ve yardımcı kodları bulunmaktadır. Bu modelleri kullanabilmek için, eğitim ve test için kullanılacak bir görüntü havuzunun olması gerekmektedir. Tez sürecinde, hareketli el görüntüleri içeren EgoHands (Bambach vd., 2015) veri setiyle bir SSD modeli adım adım eğitilip test edilmiştir (Victor, 2017). Şekil 35'te modelin çıktısı görülmektedir.



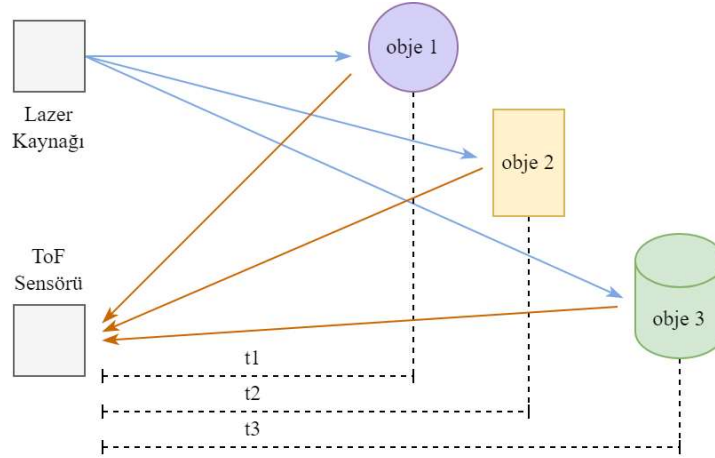
Şekil 35. EgoHands (Bambach vd., 2015) veri seti ile eğitilmiş SSD ağıyla el tanıma örneği

2.2.8. Kinect 2 Kamerası ile El Tanıma

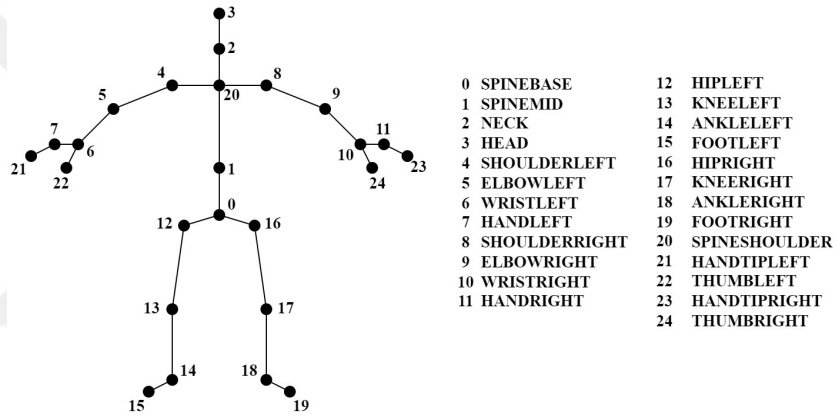
Kinect 2 kamerası (Şekil 36) hem derinlik hem RGB görüntüler sunabilen, Microsoft tarafından Xbox konsol sistemleri için üretilmiş bir kameradır. 1920x1080 piksel boyutlarında saniyede 30 çerçeve görüntü sunan bir RGB kamerası bulunur. Derinlik kamerası kızılötesi ToF (Şekil 37) teknolojisi ile çalışmakta olup, 512x424 piksel boyutlarında görüntü sunabilmektedir. 0,5 metre ile 4,5 metre aralığında derinlik görüntüleri elde edilebilmektedir. Kinect 2 ile beraber kullanılan yazılım geliştirme kiti ile beraber, üç boyutlu olarak insan vücudunun 25 eklemine gerçek zamanlı olarak tespit edip takip edebilmektedir. Bu eklem noktaları Şekil 38’de gösterilmiştir. Kinect 2 kamera aynı anda 6 kişiyi tespit edebilmektedir.



Şekil 36. Kinect 2 kamerası (URL-22, 2023)



Şekil 37. Time-of-Flight kameralarının çalışma prensibi



Şekil 38. Kinect 2 kamerasının iskelet takibi sisteminde kullandığı eklem noktaları

Kinect 2 kendi içinde bulunan eğitilmiş sınıflandırıcılar sayesinde el takibi de yapabilmektedir. Ancak bunun için, görüntüde kişinin kameranın karşısında olması ve vücudunun büyük bir kısmı ekranda görülecek şekilde konumlanması gerekmektedir.

Şekil 39’da görülen çalışmada, Kinect 2 kamerasının sınıflandırıcısı ile iskelet ve el takibi yapılmıştır. PyKinect2 kütüphanesi sayesinde, tespit edilen eklemlerin x , y , z koordinatları bilinmektedir. Sol karede, bu avuç içi takip noktasının Python’da OpenGL kütüphanesi yardımıyla izlediği yörünge (trajectory) çizdirilmiştir.



Şekil 39. Kinect 2 ile tespit edilen avuç bölgesi yörüngesinin OpenGL ile çizdirilmesi (solda), Kinect 2 ile el ve parmak tespiti (sağda)

2.3. El Yıkama Hareketi Tanıma İçin İzlenen Yöntemler

El yıkama hareketleri tanıma için sensör bazlı yaklaşımlar, suyla temas etme ve kısıtlayıcı olma dezavantajlarını barındırmaktadır. Bu sebeple, görüntü bazlı yaklaşımla bu hareketleri denetlemek kullanıcı açısından en rahat deneyim olacaktır. Bu bağlamda, tez çalışmasında izlenen yolda Kinect kamerası kullanılmıştır.

Kinect kamerasından elde edilen renkli görüntünün niteliği, ortamdaki ışığa, cisimlerin hareketine, Kinect içerisinde firmware olarak gelen poz (exposure) ayarlamalarına göre değişmektedir. Bu da renkli görüntüleri kullanarak arka plan çıkarımı yapmayı ve bozulmalar olmadan kişi tespit etmeyi zorlaştırmaktadır. Kinect kamerasının sunduğu bir imkan olan ve gri düzeyde elde edilen kızılötesi görüntüler de ışık değişiminden etkilenmektedir. Ayrıca, kızılötesi görüntülerde hareket eden cismin arka plandan farklı bir renkte temsil edilmesi her zaman geçerli değildir. Aynı kızılötesi enerjiyi yayan cisimler biri arka plan (background) biri cisim (foreground) olsalar da üst üste geldiklerinde fark edilmeyecektir.

Derinlik görüntüleri, renkli görüntülerin aksine ortamdaki ışığın değişmesinden etkilenmemektedir. Derinlik algısı sayesinde ortamdaki herhangi bir hareket rahatlıkla algılanabilmektedir. Bu sebeple, derinlik görüntülerinde değişmez bir arka plan üzerindeki hareket eden cisimleri tespit etmek çok daha kolay olmaktadır. Ayrıca, tespit edilen cisim daha az bozulmaya uğramakta, cismin şekli gerçektekine daha uygun elde edilmektedir.

Arka plandan başarılı bir biçimde izole edilen nesne el yıkamak için ortamda bulunan kişi olacaktır. Bu sayede ilgi alanımız olan el bölgesindeki hareketi incelemek mümkün

olacaktır. El bölgesi belirlemek için çeşitli algoritmalar incelenmiş, bunlar önceki başlıklar altında sunulmuştur. Bu yöntemler arasında, doğruluk oranı en yüksek olan model eğitilmiş bir sınıflandırıcı kullanılmasıdır. Ancak, daha önce belirtildiği üzere, bu sınıflandırıcının gerçek zamanlı bir sistem üzerindeki maliyeti gereğinden fazla olacak, bir sonraki aşama için kullanılacak sınıflandırıcının performansını da etkileyecektir. Bu durumda, daha düşük iş yüküne sahip bir çözüm geliştirilmesi uygun olacaktır.

Kameranın yerleştirileceği optimum konum, el yıkama esnasında elleri en iyi şekilde görebilecek, suyla temasın yaşanmayacağı ve sağlıklı bir derinlik verisinin elde edilebileceği bir konum olmalıdır. Yanlara veya çapraza yerleştirilecek olan kamera, ellerin ikisini de eşit miktarda göremeyecektir. Olası konumlardan en uygunu, kameranin lavaboyu yukarıdan göreceği ve lavaboya dik olarak yerleştirileceği konumdur. Hem veri toplamak için kurulacak düzeneklerde, hem de tez sonucunda oluşabilecek potansiyel bir ürün düşünüldüğünde, montaj açısından da konum uygun düşmektedir.

Derinlik görüntülerinin bir diğer avantajı ise, kendilerinden üç boyutlu noktalar kümesi elde edilebilmesidir. Derinlik görüntülerinde, her bir pikselin değeri, kamera kalibrasyon değerlerine göre hesaplanıp, üç boyutlu uzayda x , y , z koordinatlarına dönüştürülebilir. Böylece iki boyutlu görüntüden cisimlerin üç boyutlu şekilleri çıkarılabilir. Literatürde, bu noktalar kümeleri de tıpkı iki boyutlu görüntüler gibi işlenip, çeşitli sınıflandırıcılarda veri olarak kullanılabilmektedir. Bu da renkli görüntülerle sınıflandırmanın yanı sıra, noktalar kümesiyle de sınıflandırma yapmaya, hatta hibrit yöntemle iki veri türünün de bir arada kullanılmasına imkan sunmaktadır.

Üç boyutlu nokta kümelerinin kullanımı, derinlik görüntülerinin gelişerek daha çok kullanılmasına bağlı olarak hızla yaygınlaşmaktadır. Bu noktalar kümesinden obje sınıflandırması yapabilen yapay sinir ağı modelleri mevcuttur ve geliştirilmeye devam edilmektedir.

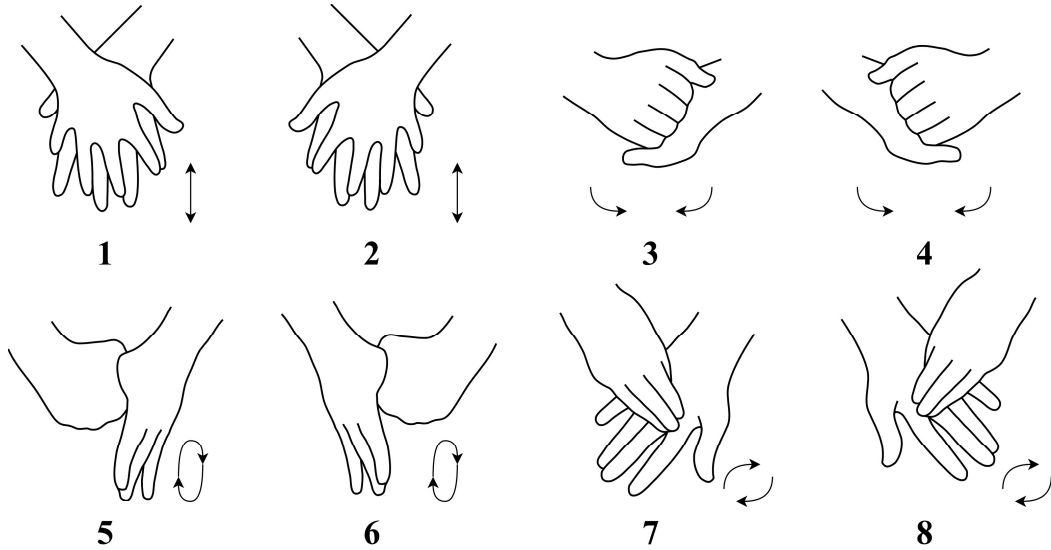
El yıkama hareketlerinin tanımlanabilmesi için gerekli olan tüm aşamalarda, literatürde yaygın olarak kullanılan çeşitli metotlar değerlendirilmiştir. Görüntü elde etme aşamasında, ortamdaki ışık ve renk değişimlerinden minimum etkilenmesi sebebiyle derinlik görüntüleri, el tespitinde ve noktalar kümesi verileri elde etmede tercih edilmiştir. Renkli görüntülerden elde edilen el bölgesi görüntüleri de yine sınıflandırma için kullanılmıştır. Elde edilen görüntülerin sıkıştırılmadan ve bit derinlik bilgisi kaybolmadan saklanabilmesi için çerçeve (frame) biçiminde diskte tutulması uygun bulunmuştur.

2.3.1. Tanımlanan El Hareketleri

El hijyeninin yeterli sağlanması için, DSÖ'nün önerdiği birtakım hareketler mevcuttur (URL-2, 2023). Makine öğrenmesi temelli el yıkama hareketi tanıma çalışmalarında, benzer hareketler tanımlanmışsa da bugüne kadar ortak kabul edilmiş hareketler bulunmamaktadır.

Bu tez sürecinde Şekil 40'ta görülen sekiz farklı el yıkama hareketi tanımlanmıştır. Bunlardan ilki sağ elin sol el üzerine koyularak ileri geri hareketle parmak aralarını ve elin üstünü yıkamadır. İkinci hareket bunun simetriğidir. Üçüncü hareket, sağ elin parmaklarının ve avucunun, sol elin parmakları ve avucu ile kilitlenerek, ellerin sağa sola hareketiyle yapılan yıkamadır. Dördüncü hareket bunun simetriğidir. Beşinci hareket, sol elin başparmağının sağ elin avucu içerisine alınarak, döndürme biçiminde yapılan yıkamadır. Altıncı hareket bunun simetriğidir. Yedinci hareket, sağ elin parmak uçlarının, sol elin avucu içerisinde dairesel hareket ettirilmesiyle yapılan yıkamadır. Sekizinci hareket ise bunun simetriğidir.

Yeterli hijyenin sağlanması için genellikle el yıkamanın en az bir dakika sürmesi tavsiye edilmektedir. Bu süreç içerisine sabunu alma, musluğu açma ve durulayıp musluğu kapatma da dahil edilirse, bu işlemlerin ortalama 20 saniye süreceği varsayımıyla, geriye kalan sekiz hareketin her biri en az beş saniye yapılmalıdır.



Şekil 40. DSÖ önerilerine uygun olarak belirlenen sekiz el yıkama hareketi

2.3.2. Kamera Görüntüleri Veri Seti

Veri seti, makine öğrenmesi alanında en önemli unsurlardan birisidir. İlgilenilen problemin çözümünün sağlanması, yeterli miktarda veri içeren, oluşabilecek tüm durumlarla ilgili örnek bulunduracak bir veri setine sahip olunmasına bağlıdır. Yüksek doğruluk oranı ile bir sınıflandırma gerçekleştirilebilmesi için, Kinect kamerasından elde edilen RGB ve derinlik görüntüleri üzerinden farklı türde veri setleri elde edilmiş ve hareket tanımadaki kullanılabilirlikleri ölçülmüştür.

Oluşturulan veri setinin etkili olabilmesi için, iki kişiden oluşan katılımcıyla farklı iki mekanda her 8 hareketten her biri toplamda 400 adet olmak üzere 3200 oturum yıkama hareketi kaydedilmiştir. Kayıtların alındığı dönem, Covid-19 nedeniyle yaşanan kapanmalarla aynı dönem olduğu için katılımcı sayısı sınırlı olmakla beraber, veri miktarı artırılarak çeşitlilik artırılmıştır. Derinlik görüntüleri 16-bit derinliğinde imgeler olarak kaydedilmiştir. El yıkama hareketleri, gerçek hayata uygun olacak şekilde değişken sabun miktarlarında ve hızlarda yapılmıştır.

Kinect kamerasından eş zamanlı olarak hem derinlik hem de renkli görüntülerin elde edilip işlenmesi ve uygun formatta kaydedilmesi için çeşitli çalışmalar yapılmıştır. Bu amaçla Python kodunda iş parçacıklarıyla (thread) çalışılmış ve performans artışı hedeflenmiştir.

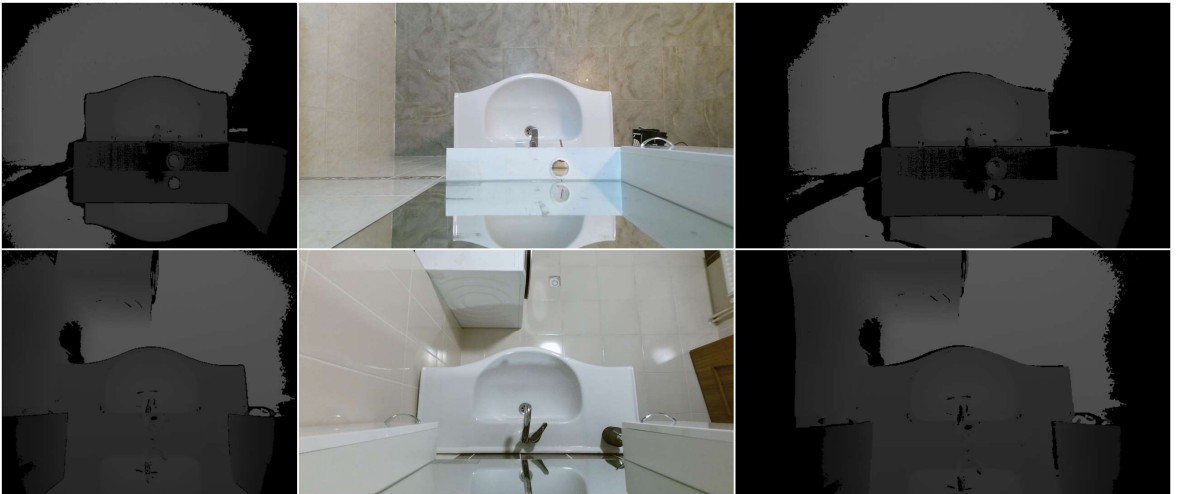
Başlangıçta, elde edilen görüntüler XVID kodlayıcı-kodçözücüsüne (codec) sahip video formatında kaydedilmiştir. Bu videoların işlenebilmesi için, her bir videonun tekrar okunarak OpenCV yardımıyla işlenmesi gerekmektedir. Buradaki dezavantajlardan birisi, kodlayıcı-kod çözücüyle kodlanarak kaydedilen videoların 8 bit çerçevelerden oluşmasıdır. Aslında, ham derinlik görüntüleri, 16 bit, ham renkli görüntüler 24 bit derinliğine sahiptir. 8 bit derinliğe dönüştürüldüğünde veri kaybı yaşanmakta, derinlik görüntülerinden noktalar kümesi oluşturmak için tekrar 16 bit düzeyine dönüştürülmesi gerekmektedir. Ayrıca, OpenCV ile video kaydederken, hesaplanarak elde edilmesi gereken çerçeve oranı değeri bulunmaktadır. Bu çerçeve oranı, sabit olmayıp, çalışan koddaki işlem yükü arttıkça düşmektedir. Anlık değişimler gösterebilen bu oranı çalışma anında sabit bir değer olarak belirleyebilmek gerekmektedir. Bu da videolarla çalışmayı zorlaştırmaktadır. Dolayısıyla, görüntünün çerçeveler şeklinde görüntü dosyaları olarak kaydedilmesi uygun olacaktır. Daha sonra eğitim için işlemek istendiğinde herhangi bir dönüşüm gerektirmediği için de ayrıca kolaylık sağlamaktadır.

Kinect kamerasının sunduğu renkli görüntülerin çözünürlüğü yüksek çözünürlüklü boyutta (1920x1080) olduğundan, gerçek zamanda işlenmesi işlem yükünü artırmaktadır. Dolayısıyla renkli görüntüler yarı boyutuna dönüştürülerek 960x540 el tespitinde kullanılmış ve diskte kaydedilmiştir. Derinlik görüntüleri ise 512x424 boyutunda standart olarak kaydedilmiştir.

Kinect üzerinde bulunan derinlik kamerası ve renkli kamera için konum, görüş açısı ve kalibrasyon değerleri farklıdır. Kinect kameranin sunduğu bir fonksiyon ile, derinlik görüntüleri renkli görüntüler ile eşleştirilebilmektedir. Kinect üzerindeki entegre çipin içerisinde gerçekleşen bu dönüşüm fonksiyonu ile, derinlik görüntüsü renkli kamera lensinden alınmış bir görüntü gibi yeniden hesaplanıp eşleştirilmektedir. Böylelikle, derinlik görüntülerinde el bölgesi belirlemek için kullanılan yöntemler, renkli görüntüler üzerinde de kullanılmıştır. Sonuç olarak hem derinlik hem de renkli görüntülerde el bölgesi tespiti yapılmıştır.

Bir noktanın üç boyutlu dünyadaki konumu biliniyorsa, bu noktanın bir kamera üzerindeki izdüşümü kameranin intrinsic (K_c) ve extrinsic (M_0) matrisleri bilindiği sürece bulunabilir. Formül 30'da görüldüğü üzere K_c , M_0 matrisleri ve x , y , z koordinatları ile iki boyutlu görüntüde pikseller (u , v) bulunabilir. Kinect kamerasında derinlik ve renkli görüntülerin eşleştirilmiş halleri Şekil 41'de görülmektedir.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{K_c} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}_{M_0} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 0 \end{bmatrix} \quad (30)$$



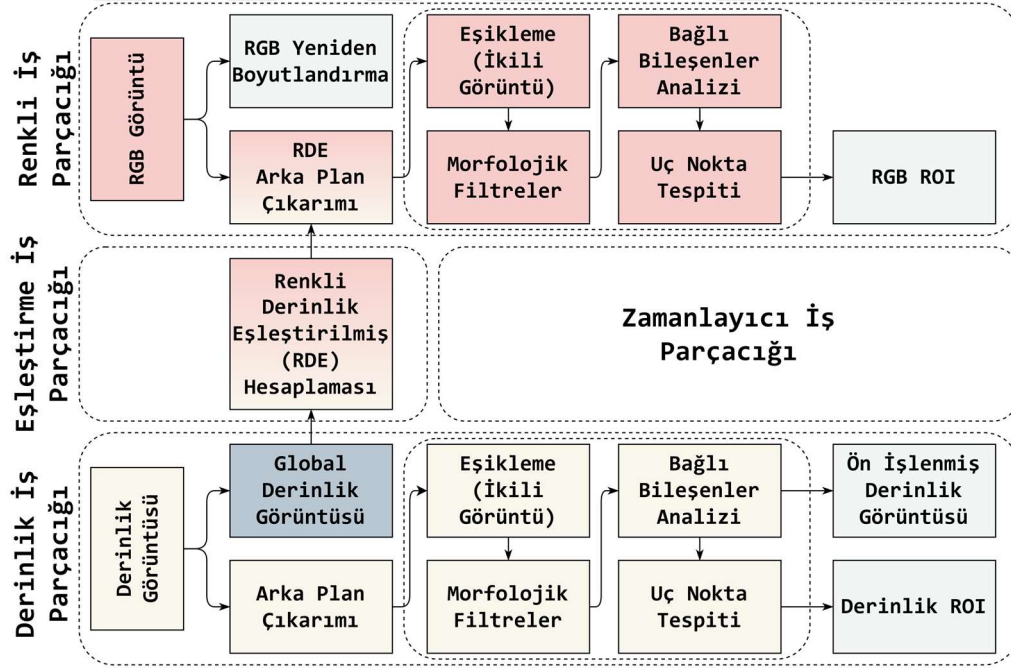
Şekil 41. Veri setinde kullanılan lavaboların arka plan görüntüleri. Derinlik görüntüleri (solda), RGB görüntüleri (ortada), renkli-derinlik eşleştirilmiş görüntü (sağda)

Renkli görüntüler üzerinde, elde edilen orijinal çerçeve ile başlangıçta elde edilen arka plan çerçevesinden çıkarıldığında, görüntüde gürültü meydana gelir ve cismin orijinal kontörleri tam olarak elde edilemez. Çünkü, cisim hareket ettikçe kameraya yansıyan ışık değişmiş ve otomatik parlaklık ayarlamaları sebebiyle renk düzeyleri değişmiştir. Burada çeşitli morfolojik filtreler ile gürültü temizlenebilir ancak, cismin kontörü daha da fazla bozulmaya uğrayacaktır. Renkli görüntüler üzerinde, kenar bulma yöntemiyle elde edilen ikili görüntülerde çerçeve farkına bakılmış ve kontör bilgisi morfolojik olarak iyileştirilmeye çalışılmıştır. Ancak, derinlik görüntülerinde elde edilen başarı burada elde edilememiştir. Farklı renk uzayları ayrı ayrı denenmiş, birden fazla renk uzayının bir arada kullanılıp farklarının birleştirildiği görüntüler üzerinde çalışılmış, fakat kontör bilgisinin bozulmaları engellenememiştir. Dolayısıyla bu yöntem yerine, renkli-derinlik eşleştirilmiş görüntü üzerinde el tespiti işlemleri yapılmış, renkli görüntülerle aynı koordinatlara denk düştüğü için, renkli görüntüden de el bölgesi bu yöntemle tespit edilmiştir. Bu yöntemle ilgili detaylı bilgi bir sonraki bölümde verilmiştir.

Kurulan sistemin çalışma mantığı, Şekil 42'deki gibi olup, aşamaları şu şekildedir;

- İlk aşamada sahnede herhangi bir hareketli nesne olmadan arka plan görüntüsü elde edilmektedir. Tez çalışmasında 3 saniye boyunca Kinect kamerası ile alınan görüntülerden renkli, derinlik ve renkli-derinlik eşleştirilmiş arka plan görüntüleri oluşturulmaktadır. Bu süreçte devamlı görüntü alınmakta olup, alınan en son çerçeve arka plan görüntüleri olarak kaydedilmektedir.
- Kameradan alınan görüntü bilgilerinden ön işleme yapmak ve el bölgesini bulmak için çalışan üç farklı iş parçacığı ile beraber dördüncü bir zamanlayıcı iş parçacığı kullanılmaktadır. Bu iş parçacıklarının biri renkli, diğeri derinlik ve üçüncüsü de renkli-derinlik eşleştirilmiş görüntü elde etmektedir.
- Renkli görüntüye ait iş parçacığının içinde elde edilen renkli-derinlik eşleştirilmiş görüntünün, daha önceden elde edilmiş renkli-derinlik eşleştirilmiş görüntü arka planı ile sadece iki çerçevenin farkı alınmakta, elde edilen fark görüntüsü eşikleme parametresi 10 olacak şekilde ikili görüntüye dönüştürülmektedir.
- Elde edilen ikili görüntüde, morfolojik filtrelerden sonra kişi tespitini yapmak için bağlı bileşenler analizi yöntemiyle en büyük parça tespit edilip, diğer kısımlar gürültü kabul edilip temizlenmektedir.

- Bu en büyük parçanın kontör bilgisi elde edilmektedir. Sınır koordinatları yardımıyla, görüntünün koordinat sistemine göre y koordinatı sayısal olarak en küçük olan nokta, ellerin uç noktası kabul edilmektedir.
- Bu nokta etrafında, renkli görüntülerde nokta çerçevesinin alt kenarının ortasında olacak şekilde 150x150'lik bir ROI bölgesi kırılmakta ve el bölgesi olarak kabul edilmektedir. Tespit edilen nokta ile görüntünün alt kenarı arasında 25 piksellik bir tampon bölge bırakılmaktadır. El yıkama hareketlerinin sığabileceği bölge sabit bir kutu olarak belirlenmiştir.
- Benzer işlemler, derinlik iş parçacığı içerisinde derinlik görüntüsüne de uygulanmaktadır. Derinlik görüntüleri noktalar kümesi elde edilmesi için kullanılacağından, tampon bölge burada bırakılmamıştır. Bu görüntülerdeki ROI bölgesi de 100x100 piksel boyutuyla kırılmıştır.
- El yıkayan kişiye kayıt süresi boyunca beş saniyede bir sinyal sesi gönderilmiştir. Burada zamanlayıcı iş parçacığı kullanılmıştır. Bu da yıkama hareketlerini uygun zaman aralıklarıyla yapmasını sağlamıştır.



Şekil 42. Gerçek zamanlı el bölgesi tespiti ve veri seti oluşturma sisteminin genel çalışma şekli

Gerçek zamanlı görüntüler elde edilirken, orijinal derinlik görüntüsü ve renkli-derinlik eşleştirilmiş görüntü üzerinde arka plan çıkarımı ve hareketli nesnenin bulunması işlemleri aynı zamanda yapılmaktadır. Bunun sebebi, renkli-derinlik eşleştirilmiş görüntünün, Kinect'in çalışma anında içerisinde bulunan donanımı sayesinde elde edilebilmesidir.

Veri seti oluşturulurken, 8 hareketten oluşan tanımlı problem için, her bir hareket 400 kere kaydedilmiştir. Bunlardan 184 kayıt birinci lavaboda, 226 kayıt ise ikinci lavaboda alınmıştır. Bu 400 kayıt içerisinde her bir kayıt için aşağıdaki klasörlere ayrılmış şekilde veriler tutulmaktadır;

- color: Orijinal RGB görüntünün, yarı boyuta indirgenmiş (960 x 540) halini içerir.
- color_roi: Ön işleme ve el bulma sonrasında el bölgesini (150 x 150 ROI) içerir.
- depth: Orijinal 16 bitlik gri düzey derinlik görüntülerini içerir.
- depth_clean: Arka planı temizlenmiş kişiyi içeren derinlik görüntülerini içerir.
- depth_roi: Ön işleme ve el bulma sonrasında el bölgesini (100 x 100 ROI) içerir.

Bu veri setine Veri Seti-A adı verilmiştir. Buradaki derinlik görüntülerini (depth_roi) kullanarak oluşturulan veri setine de veri seti Veri Seti-B adı verilmiştir. Buradaki veri tipleri şu şekildedir;

- Noktalar kümesi (Point cloud) verisi: Derinlik görüntülerinde bulunan her bir pikselin değeri, kamera parametreleri de kullanılarak üç boyutlu uzayda temsil edilen bir noktayı hesaplamakta kullanılabilir. Buradan elde edilen noktalar kümesi, iki boyutlu görüntüden üç boyuta geçmeyi sağlar. Bu dosya türü point cloud (noktalar kümesi) olarak adlandırılmaktadır. Bu veri tipini işleyen ağlar sayesinde, noktalar kümeleri üzerinden de sınıflandırma işlemi yapılabilmektedir. Open3d (URL-23, 2023) kütüphanesi yardımıyla, Kinect 2 kamerasının intrinsic parametreleri ile derinlik görüntüleri PLY uzantılı noktalar kümelerine dönüştürülmüştür.
- Birleştirilmiş noktalar kümesi verisi: Bir hareketin her bir çerçevesinden elde edilen noktalar kümeleri, aynı üç boyutlu uzayda birleştirilirse, hareketin toplamının bir haritası üç boyutlu olarak elde edilmiş olur. Yine noktalar kümesi kullanarak sınıflandırabilen ağlar eğitilerek, hareketlerin toplam noktalar kümeleri üzerinden bir sınıflandırma gerçekleştirilebilir. Bu veri tipinde, daha önce her bir çerçeve için oluşturulmuş noktalar kümesi, tüm hareketi temsil etmesi için üç boyutlu uzayda indeksi miktarınca z ekseninde kaydırılarak birleştirilmiş ve her yıkama seansı için PLY dosyası olarak kaydedilmiştir.

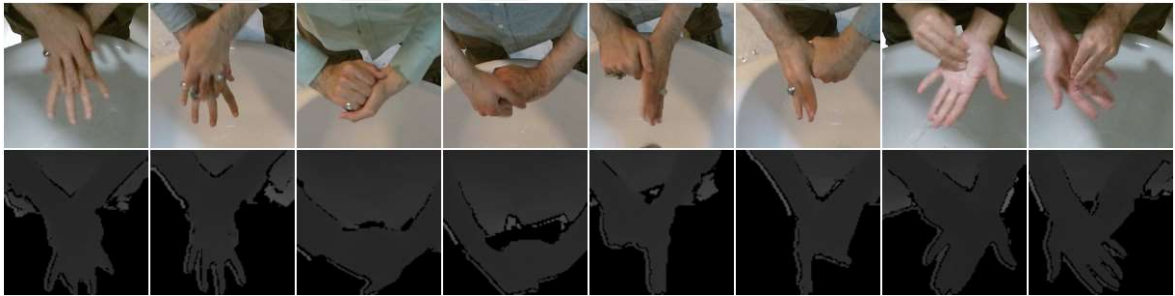
- Noktalar kümesi ile elde edilmiş projeksiyon görüntüsü: Bir noktalar kümesine ait noktalar üç boyutlu uzayda yer aldığı için, bu noktaları istenen açılardan görüntülemek mümkündür. Tek bir noktadan alınmış (tez problemi için yukarıdan) derinlik görüntüleri sayesinde, elde edilen noktalar kümesi istenen açılarda görüntülenebilir, böylece bir görüntü çerçevesi, birden fazla kamerayla aynı anda görüntülenmişçesine bir veri edinilir. Bu durum, bize hareketi daha iyi tanımlayacak öznitelikler sağlayabilir. Noktalar kümesi şeklindeki verilerden, Open3d kütüphanesi yardımıyla ellerin yukarıdan (sol üst), karşıdan (sağ üst), sağdan (sağ alt) ve soldan (sol alt) görünüşü her bir çerçeve için elde edilip tek bir görüntüde birleştirilmiş ve 300x300'lük projeksiyon görüntüleri elde edilmiştir (Şekil 44).

Noktalar kümesi verilerinde nokta sayısı sabit olmadığından, sınıflandırıcı ağ yapısına uygun hale getirmek için bu veriden sabit bir miktarda örneklem (sampling) nokta seçilmesi gerekmektedir. Noktalar kümesi olarak kategorize edilmiş verilerde her bir çerçeve için rastgele 512 nokta örneklem alınmıştır. Birleştirilmiş noktalar kümesi dosyalarında ise 12.800 rastgele nokta örneklem alınmıştır.

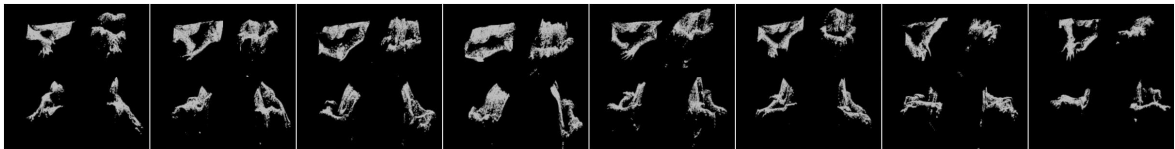
Tüm veri setinde yapılan ortalama, RGB görüntülerin ortalama çerçeve sayısı 9,7 olarak hesaplanmıştır. En düşük çerçeve oranı 8, en yüksek çerçeve oranı 11,4 olarak ölçülmüştür. Aynı şekilde derinlik görüntülerindeki ortalama çerçeve oranı 29 olarak ölçülmüştür. En düşük çerçeve oranı 24, en yüksek çerçeve oranı 33,6 olarak ölçülmüştür. Bu çerçeve oranlarının, görüntüler alınırken ön işlemden geçip, aynı anda işlenerek el bölgesinin çıkarılması ve yukarıda verdiğimiz veri türlerine bölünmesi sebebiyle, standart çerçeve oranından daha düşük olması normal ve beklenen bir durumdur. Tablo 1'de veri setinde bulunan çerçeve miktarlarının, hareketlere göre dağılımları verilmiştir.

Tablo 1. Kamera veri setindeki çerçevelerin miktarı ve dağılımı

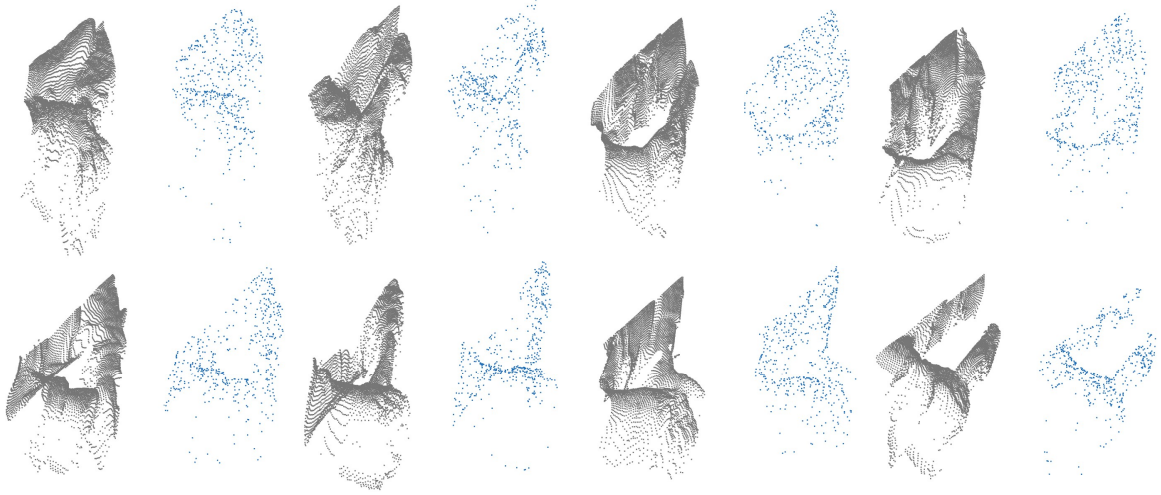
Hareket	Veri seti – A			Veri seti – B		
	Eğitim & Doğrulama	Test	Toplam	Eğitim & Doğrulama	Test	Toplam
1	17.589	3.048	20.637	50.888	8.995	59.883
2	16.983	2.848	19.831	50.575	8.598	59.173
3	16.546	2.819	19.365	49.558	8.554	58.112
4	16.306	2.783	19.089	49.208	8.459	57.667
5	16.358	2.767	19.125	49.009	8.374	57.383
6	16.298	2.808	19.106	48.974	8.509	57.483
7	16.232	2.797	19.029	49.194	8.557	57.751
8	16.180	2.880	19.060	49.118	8.668	57.786
Toplam	132.492	22.750	155.242	396.524	68.714	465.238



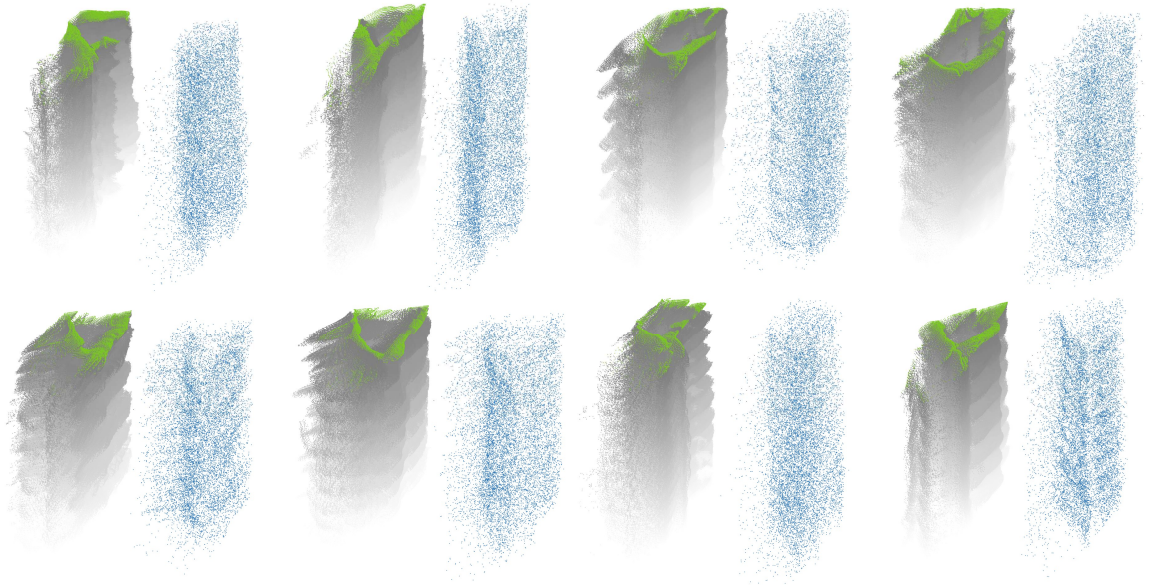
Şekil 43. Sekiz yıkama hareketi için RGB görüntülerinden elde edilen el bölgesi örnekleri (ROI) (üstte), derinlik görüntülerinden elde edilen el bölgesi örnekleri (altta). Soldan sağa hareketler: 1-8



Şekil 44. Sekiz yıkama hareketi için projeksiyon görüntülerinden örnekler. Soldan sağa hareketler: 1-8



Şekil 45. Sekiz yıkama hareketi için, noktalar kümesi tipinde elde edilen verilerden örnekler. Üst sıra soldan sağa: hareketler 1-4. Alt sıra soldan sağa: hareketler 5-8



Şekil 46. Sekiz yıkama hareketi için, birleştirilmiş noktalar kümesi tipinde elde edilen verilerden örnekler. Üst sıra soldan sağa: hareketler 1-4. Alt sıra soldan sağa: hareketler 5-8

Şekil 43'te, RGB ve derinlik görüntülerinden elde edilen el bölgeleri her bir hareket için gösterilmektedir. Şekil 44'te her bir hareket için projeksiyon görüntüleri örnekleri verilmiştir. Şekil 45'te, her bir hareket için noktalar kümesi verilerinden örnekler verilmiştir. Soldaki gri görüntü ham veri, mavi ise örneklem sonrası elde edilen 512 noktayı göstermektedir. Şekil 46'da birleştirilmiş noktalar kümesi verilerinden örnekler verilmiştir.

Soldaki gri görüntü ham veri, yeşil kısım ham veriyi oluşturan çerçevelerin bir örneğini, sağdaki mavi ise örneklem sonrası elde edilen 12.800 noktayı göstermektedir.

2.3.3. Sentetik Veri Seti

El yıkama hareketlerinin sınıflandırılması probleminde kullanılacak veri setlerini oluşturmak, mekanların lavabolar olması sebebiyle çeşitli zorluklar barındırmaktadır. Farklı kişilere ve farklı mekanlara ait çok sayıda veri elde etmek oldukça zahmetli olabilmektedir. Örneğin, kameranın lavabonun üstünde duvara monte edilmesi için, ayrıca bir aparat tasarlanması gerekmekte ve duvarın delinmesi gerekebilmektedir. Ya da tripod kullanılmak istense bile, lavabo kısmı yerleşim için uygun olmayabilmektedir. Aynı zamanda, birçok farklı kişiden veri toplanmak istediğinde, uzunca süreler lavaboyu kullanmak ve katılımcılara su ve sabun kullandırarak el yıkama yaptırmak gerekmektedir. Elde edilen veri setinin herkese açık bir hale getirilmesi için, özel izinler temin edilmesi gerekmektedir. Bu ve benzeri durumlar için, sentetik veri üretimi, yöntemlerin denenmesi için oldukça sık başvurulan bir yaklaşımdır.

Bilgisayar üretimli imgeleme (Computer Generated Imagery, CGI), uzun yıllardan beri var olan ve bilgisayarların hesaplama gücü kullanılarak görüntü elde etme tekniğidir. Çeşitli modelleme ve animasyon yazılımları sayesinde, gerçeğe oldukça yakın animasyonlar veya sabit görüntüler elde etmek mümkündür. Bugün sinema sektörünün vazgeçilmezi olmuş olan bu görüntüler, veri seti oluşturmanın zor olduğu konularda yardımcı görev üstlenebilir. Buradan hareketle, tez konusu için farklı karakterler ve farklı lavabolar içerecek şekilde, üç boyutlu modellerle gerçeğe benzer animasyon görüntüler elde edilmiştir.

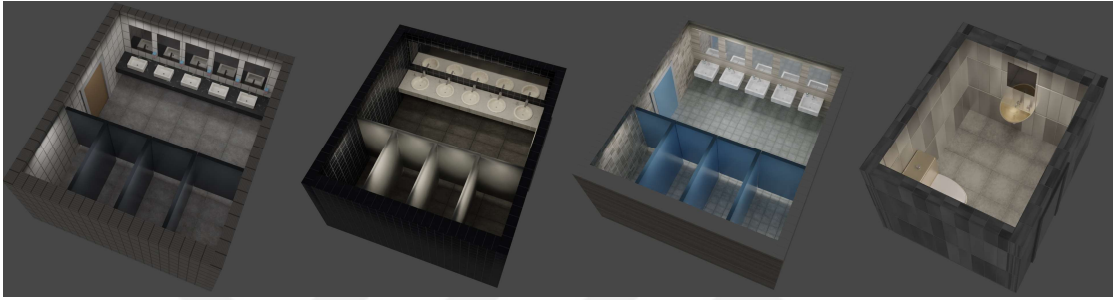
Üç boyutlu bilgisayar grafikleri, temel olarak üç boyutlu bir uzayda yer alan köşe noktaları (vertice, vertex) ve bunların birbiri ile bağlantıları sonucu oluşan poligon (polygon) geometrik şekilleriyle temsil edilmektedir. Bu poligonların bir düzlemi temsil ettiği düşünülürse, her bir poligon yüzeyi yüz (face) adını almaktadır. Her bir yüzeyin dönük olduğu ve temsil ettiği, yüzeye dik olan yöne normal denilmekte ve bir vektörle temsil edilmektedir. Poligonların türü, miktarı bir modelin topolojisini (topology) oluşturmaktadır. Modellemede, en az sayıda köşe ve yüz kullanılarak, mümkün mertebe en detaylı ve gerçekçi görüntünün elde edilmesine çalışılmaktadır. Poligonların genellikle üçgen veya dörtgen olmasına dikkat edilmektedir. Daha fazla kenar sayısına sahip poligonlar, render işleminde istenmeyen sonuçlar doğurabilmektedir.

Yüzlerin çeşitli renk ve desenlerle kaplanması, modellere dokusunu (texture) vermektedir. Üç boyutlu oyun veya render motorları, iki boyutlu bir görsele ait renk bilgilerinin, üç boyutlu bir modelde hangi yüzlere hangi koordinatlarda denk düşeceğini hesaplayıp gösterebilmektedir. Bu tekniğe UV haritalama (UV mapping) ismi verilmektedir. U ve V harfleri, x ve y eksenlerini temsil etmektedir. Modelleme yazılımları sayesinde bu işlem görsel olarak yapılabilmektedir. Bu doku bilgisinin yanı sıra, ortamdaki aydınlatmanın oluşturacağı etki ve yansımalar, yüzeydeki pürüzler ve parlaklıklar, o yüzeye ait materyali oluşturmakta ve parametreler ile belirlenebilmektedir. Sahneye ait kameranın gördüğü alandaki tüm yüzeyler için belirlenen tüm bu özellikler, uzun hesaplamalar sonucunda her bir piksele denk düşecek şekilde renk bilgisine dönüşmekte ve bir görüntü üretilmektedir. Buna rendering adı verilmektedir. Rendering için güçlü işlemciler ya da ekran kartları daha hızlı sonuç vermektedir.

Herhangi bir modele, bir iskelet benzeri yapı eklenerek köşelerin ve poligonların hareket ettirilmesiyle animasyonlar tasarlanabilmektedir. Bu işleme rigging adı verilmektedir. Blender’da, insan anatomisine uygun varsayılan bir rig, karakterlere eklenip, onları hareket ettirmekte kullanılabilmektedir. Ayrıca bu varsayılan iskelet, ters kinematik hesaplamalarını otomatik yapmaya imkan sunduğu için, insana ait hareketler gerçekçi bir biçimde animasyona dönüştürülebilmektedir. Blender’ın rig sisteminde, bir eklem oynatıldığında etkilenmesi gereken diğer eklemler otomatik olarak bu kinematik hesaplamalarla oynatılmaktadır. Bir karaktere ait iskeletin, modeldeki hangi noktaları ne kadar hareket ettireceği de yine Blender yazılımıyla belirlenebilmektedir. Bu ağırlıklar, otomatik atanabildiği gibi, kullanıcı tarafından da değiştirilebilmektedir. Bu özellik weight painting olarak adlandırılmaktadır. Örneğin, bir karakter kolunu hareket ettirdiğinde, bunun omuzlara yansması, gradyen şeklinde azalan bir şiddette olmalıdır. Bunu gerçekleştirmek için, ilgili köşe noktaları belirli bir ağırlık değeriyle boyanmakta ve hesaplamalara bu şekilde katılmaktadır.

Tez çalışmasında, dört farklı banyo tasarımı yapıp, Blender ile modellenmiştir. Tüm modeller tamamen el ile modellenmiş olup, hazır model kullanılmamıştır. Banyolar modellenirken, gerçek hayattaki banyolara benzer şekilde lavabo, mermer, fayans, ayna ve musluklar ile tasarlanmıştır. Banyoların her birinin boyutları, gerçekçi ölçülerle modellenmiş, ışıklandırması ve malzemeleri birbirinden farklı seçilmiştir. Her bir banyoda, kamera ideal senaryo gereği kullanıcının ve lavabonun üstünde olacak şekilde yerleştirilmiş ve odak uzaklığı (field of view, FOV) 80 derece olarak ayarlanmıştır. Her bir banyoda

kameraların yüksekliği ufak farklar göstermektedir. Böylece farklı yükseklikler ve açılarla aynı el yıkama hareketleri yakalanmış olacaktır. Mekanlardaki materyallerde, daha gerçekçi sonuç üretmeleri için, yüzey pürüzleri, değişik matlık / parlaklık, girinti çıkıntı ve derzleri de temsil edecek şekilde Normal, Roughness ve Displacement dokuları da kullanılmıştır. Şekil 47’de modellenen banyoların izometrik gönrüntüleri, Şekil 48’de banyolara yerleştirilen kameraların elde ettiği render görüntüleri görülmektedir. Şekil 49’da modellenen dört farklı karakter görülmektedir.



Şekil 47. Modellenen dört farklı lavabonun genel görünüşü

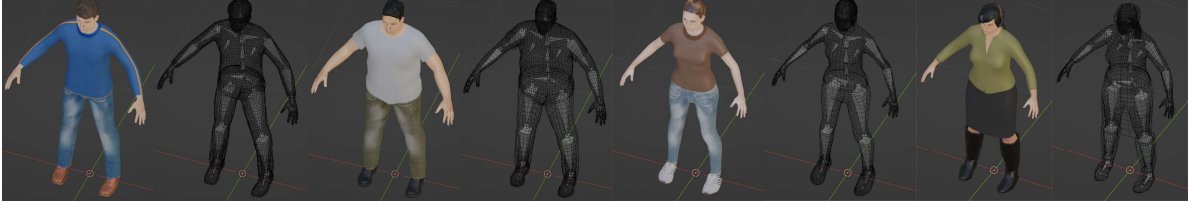


Şekil 48. Lavabolara yerleştirilen kameraların ürettiği render görüntüleri

Normal Map: Yüzey geometrisinde, geometride bulunmayan ışık ve gölge efektlerinin yönlerinin gösterilmesini sağlamak için kullanılan doku biçimidir. İşlem maliyeti az olduğundan ve modele detay eklediğinden dolayı çok yaygın olarak kullanılmaktadır.

Rougness Map: Bir yüzeyin hangi bölgelerinin ne kadar yansıtıcı olduğunu belirlemek için kullanılan doku türüdür. Pürüzlü ya da düzlem olmayan yüzeylerin daha gerçekçi görünmesini sağlamak için kullanılabilir.

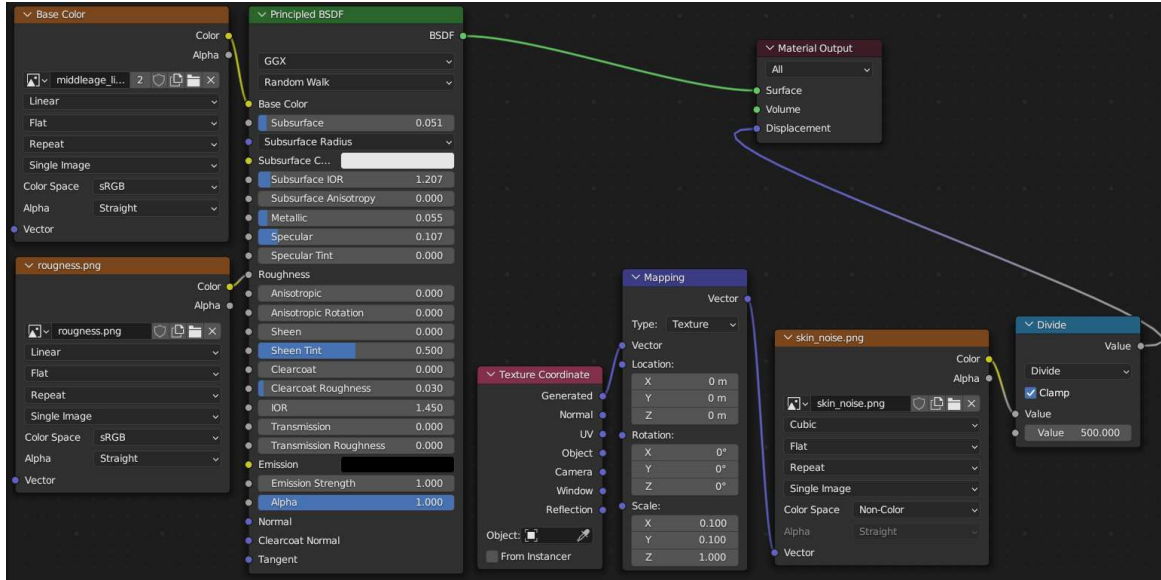
Displacement Map: Yüzey üzerindeki poligonları değiştirmeden, yüzeye pürüz ve engebe efekti katmak için kullanılan kaplamalardır.



Şekil 49. Makehuman ile modellenip çeşitli modifikasyon yapılan dört farklı karakter ve Blender'daki iskelet sistemleri

Blender'da, materyallerin özellikleri, düğüm (node) sistemi adı verilen bir görsel arayüz ile konfigüre edilmektedir. Çeşitli düğümlerin farklı sıralamalar ve ayarlarla birbiri ardınca bağlanmasıyla istenen görüntü özellikleri sağlanabilmektedir.

Makehuman ile üretilen karakter modelleri, yaygın bir üç boyutlu dosya formatı olan DAE (Digital Asset Exchange) uzantısıyla kaydedilmiş ve Blender'a aktarılmıştır. Burada bazı düzeltmelerinden geçen modellere iskelet (rig) giydirilmiş ve animasyona hazır hale getirilmiştir. İki kadın ve iki erkek model, farklı vücut yapılarına, deri renklerine ve kıyafetlere sahip olacak şekilde üretilmiştir. Yine mekanlarda olduğu gibi, dokuların gerçekçi olması amacıyla pürüz, parlaklık / matlık için ayrı dokular kullanılmıştır. Tüm renderlar, Blender'daki en gerçekçi görüntüleri elde etmek için kullanılan Cycles render motoru ile alınmaktadır. Şekil 50'de, Blender'da karakterlerin doku özelliklerinin belirlendiği node sistemi görülmektedir.



Şekil 50. Blender’da Node sistemi ile bir karakterin deri dokusunun nitelikleri

Her bir karakter, her bir banyo içerisinde sekiz farklı yıkama hareketini yapacak şekilde render görüntüleri üretilmiştir. Bu yıkama hareketleri yapılırken, farklı sabun seviyeleri için her seferinde aynı hareketin farklı animasyonları üretilmiştir. Bu sayede, tıpkı gerçek hayatta olduğu gibi, her yıkamada varyasyonlar oluşturulmuş ve veri çeşitliliği sağlanmış olmaktadır. Her bir el yıkama prosedürü hareketi için 31 çerçeve animasyon görüntüsü üretilmiştir. Sekiz hareket için, bir banyoda bir karaktere ait 248 çerçeve görüntü elde edilmiştir. Dört farklı sabun seviyesiyle bu hareketler yapıldığı için tüm hareketlerde 992 çerçeve görüntü elde edilmiştir olacaktır. Dört farklı banyoda bu hareketler tekrarlandığı için, 3.968 çerçeve görüntü elde edilmiş olacaktır. Bunun dört farklı karakter için yapıldığı düşünüldüğünde, toplamda veri seti 15.872 çerçeve veri içermektedir.

2.3.4. Elde Edilen Görüntülerin Niteliği ve Kategorizasyonu

Sentetik veri seti için, dört farklı banyoda 2 erkek ve 2 kadın model olmak üzere 4 farklı modelin, 4 farklı sabun seviyesiyle el yıkadığı animasyon görüntüleri oluşturulmuştur. Bu sabun seviyeleri sabunsuz (no soap), az (deficient soap), normal (normal soap) ve çok (adequate soap) şeklinde belirlenmiştir.

Elde edilen veri seti modellerde değerlendirilmek üzere üç farklı kategoriye ayrılmıştır. Birinci kategori, bir modelin ellerin sadece sabun miktarını tespit edip edemeyeceğini ölçecek görüntüler olacaktır. Bu yüzden, tüm veri setindeki görüntüler sadece sabun miktarına göre bölünüp ilgili etiketleme farklı klasörlere yerleştirilerek

yapılmıştır. Bu kategoride 4 sınıf bulunmaktadır. İkinci kategori, bir modelin el yıkama hareketini tanıyıp tanıyamayacağına yöneliktir. Bu kategorideki veriler de sabun miktarına bakılmaksızın tüm hareketlerin ilgili sınıflara ayrılmasıyla oluşturulmuştur. Bu kategoride 8 sınıf bulunmaktadır. Son kategori ise, bir modelin hem sabun miktarını hem de ellerin şu anda yaptığı yıkama hareketini birlikte tespit edip edemeyeceğine yöneliktir. Dolayısıyla, veriler 8 hareket ve 4 farklı sabun miktarına göre bölünerek, 32 sınıf elde edilmiştir.

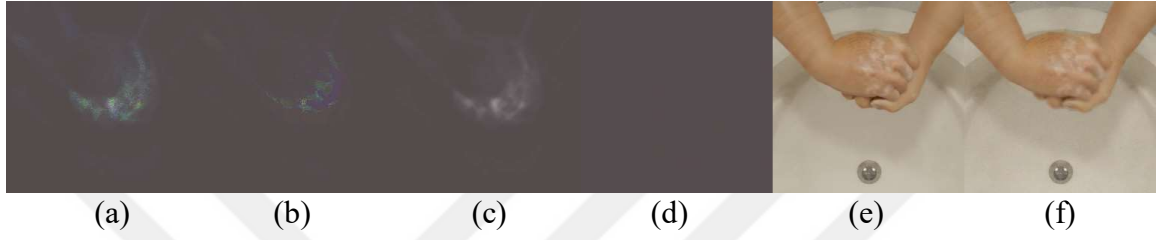
Render görüntülerinden elde edilen bu farklı kategoriler için, iki farklı ağ türünde eğitim yapılmıştır. Bu ağlar Inception-V3 ve YOLO-8 Nano ağları olarak belirlenmiştir. Çalışma prensiplerinden daha önce bahsedilen bu ağların aldığı girdi görüntüleri de farklıdır. Inception-V3 ağının ilgili sınıfları değerlendirme yapabilmesi için, el bölgesinin belirlenip, görüntüden kesilip girdi olarak ağa iletilmesi gerekmektedir. YOLO-8n ağının eğitilebilmesi için de her görüntüdeki el bölgesinin etiketlenip, ağ modelinin tanıyacağı bir formatta veri setinde “TXT” uzantılı olarak koordinat ve sınıfının belirtilmesi gerekmektedir.

Her iki ağın da standart bir değerlendirmesi olması açısından, her bir banyo için el bölgesini kapsayacak standart bir ROI bölgesi belirlenmiş ve Inception-V3 için görüntüler bu koordinatlarla kırılmış, YOLO-8n için ise bu koordinatlarla etiketlenmiştir. Daha sonra, 4, 8 ve 32 sınıfı eğitmek ve test etmek için kullanılacak 6 farklı klasöre veri seti çoğaltılmıştır (3 Inception, 3 YOLO).

Render görüntüleri ile gerçek dünya görüntüleri karşılaştırıldığında, render motorunun ve modellerin kapasitesinin yanı sıra, fark oluşturan iki temel etmen göze çarpmaktadır. Gerçek kamera görüntülerinde kamera gürültüsü bulunmaktadır ve hareket eden cisimler hareket izine (motion blur) sahiptir. Render görüntülerine bu iki etkiyi sağlamak için, gürültü ve hareket izi efektinin bir benzeri elde edilmesi için bir algoritma önerilmiştir. Bu algoritma OpenCV yardımıyla uygulanmış olup adımları şunlardır;

1. Peş peşe son 10 çerçeve görüntüsü alınır. 10 çerçeveden az durumlarda gelen çerçeveye kadar görüntü alınır.
2. Her biri bir önceki çerçeve ile karşılaştırılır ve farkı alınır.
3. Bu farka Gaussian filtresi uygulanır.
4. Bu görüntü bir biriktirici görüntüde biriktirilir.
5. Şu anki çerçevenin bir kopyası bu biriktirici görüntüsüyle “ve” operatörüyle maskelenir.

6. Maskelenen görüntü biriktirici görüntüyle “veya” operatörüyle maskelenir.
7. 6’da elde edilen görüntüye Gaussian filtresi uygulanır.
8. Orijinal görüntünün bir kopyasına Gaussian filtresi uygulanır.
9. Görüntünün orijinali ile 7’deki görüntü karıştırılır.
10. Lens gürültüsü eklemek için rastgele Gaussian gürültüsü ile oluşturulmuş boş bir görüntü elde edilen son görüntü ile karıştırılır.

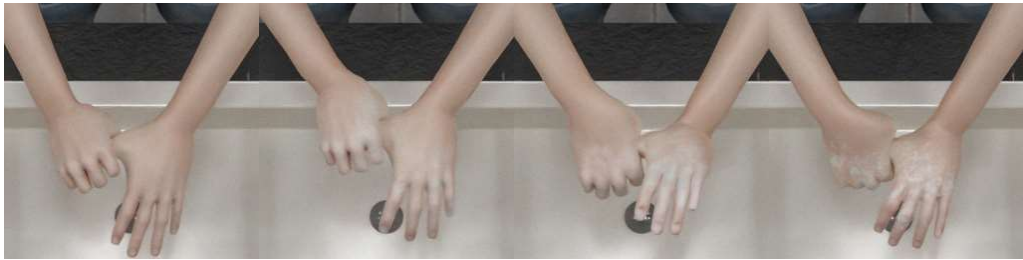


Şekil 51. Pseudo-Blur işleminin aşamalarından çerçeveler. (a) 4. adım sonrası oluşan görüntü. (b) 5. adım sonrası oluşan görüntü. (c) 6. adım sonrası oluşan görüntü. (d) Gaussian gürültüsü. (e) Görüntünün orijinali. (f) Pseudo-Blur işleminden sonra oluşan görüntü

Şekil 51’de Pseudo-Blur işleminin aşamaları görülmektedir. Şekil 52’de sentetik veri setinde farklı banyo ve karakterlere ait el bölgesi örnekleri görülmektedir. Şekil 53’te ise farklı sabun miktarlarına ait örnek görüntüler verilmiştir.



Şekil 52. Sekiz yıkama hareketinin animasyonlarından örnek çerçeveler. Soldan sağa: hareket 1-8



Şekil 53. Bir yıkama hareketinin dört farklı sabun seviyesiyle animasyonundan çerçeveler. Soldan sağa, sabunsuzdan çok sabunluya

Görüntüler ayrıldıkları üç farklı kategoride ilgili klasörlerinde Inception-V3 ve YOLO-8n ağlarına göre etiketlendikten sonra eğitim yapılmıştır. Eğitim yapılırken, kullanılan veri miktarları (eğitim, doğrulama ve test) Tablo 2’de verilmiştir. 4 sınıflı modellerde her bir sınıfa ait ayrılan eğitim veri sayısı 3.563’tür. 8 sınıflı modellerde bu sayı 1.784 ve 1.788 olmuştur. 32 sınıflı modellerde ise 444 ve 448 olmuştur.

Tablo 2. Sentetik veri seti için oluşturulan modellerin veri tablosu

Model	Eğitim & Doğrulama	Test
InceptionV3 4 sınıf	4 x 3.563	4 x 396
Toplam	14.252	1.584
Yolo-8n 4 Sınıf	4 x 3.563	4 x 396
Toplam	14.252	1.584
InceptionV3 8 sınıf	8 x 1.784	8 x 197
Toplam	14.272	1.576
Yolo-8n 8 sınıf	8 x 1.788	8 x 196
Toplam	14.304	1.568
InceptionV3 32 sınıf	32 x 444	32 x 49
Toplam	14.208	1.568
Yolo-8n 32 sınıf	32 x 448	32 x 48
Toplam	14.336	1.536

2.3.5. Derinlik Görüntüleriyle El Tanıma

Kameranın yerleştirildiği konum gereği, ekrandaki hareketli cismin (yani el yıkayan kişinin) etrafını çevreleyen kontör bilgisi yardımıyla, ellerin konumu rahatlıkla bulunabilir. Kameranın bakış açısı dolayısıyla, kontörün görüntünün y ekseninde en aşağıda kalan kısmı, yıkama esnasında ellerin en uç bölgesi olacaktır. İşlem yükü gerektirmeden basit geometrik işlemlerle, bu nokta etrafında oluşturulacak bir diktörtgen bölge (ROI), yüksek doğruluk oranıyla el bölgesini tespit etmiş olacaktır.

El takibi aşamasında, renk uzaylarına göre eşikleme uygulaması, farklı deri renklerine ve ortam ışığına hassas olması sebebiyle tercih edilmemiştir. Tez senaryosunda el şeklinin sürekli değişken olması sebebiyle şablon eşleştirme, Viola-Jones algoritmaları iyi sonuçlar elde edememiştir. Hausdorff mesafesi ile el kontörü tespiti, birden fazla kontör gerektirecek olan farklı el şekilleri için uygulanabilir olmamakla birlikte, gerçek zamanlı uygulamalarda

performansı büyük ölçüde azaltmaktadır. Öznitelik çıkarıcı – eşleştirici algoritmalar ise, şekli değişmeyen sabit objeler için uygun olmakla birlikte, sürekli değişen formda cisimler için uygun değildir. Kinect kamerasında gömülü olarak gelen ve vücut tespiti yaptıktan sonra yapılabilen el tespiti ve takibi, kamera pozisyonu sebebiyle mümkün olmamaktadır. Kinect'in el takibi yapabilmesi için, öncelikle kişinin vücudunu karşıdan göreceğ şekilde konumlanması gerekmektedir.

Bu sebeplerden ve daha verimli sonuçlar üretmesinden dolayı elleri tespit etmek için derinlik görüntüleri kullanılmıştır. Renkli görüntülere göre derinlik görüntülerinin birçok avantajı bulunmaktadır. Işık ve deri rengi değişiminden etkilenmemesi bunlardan biridir. Arka plan çıkarımı yöntemi ile, derinlik-renkli eşleştirilmiş görüntüler üzerinde elleri yıkayan kişi tespit edilmiştir. Arka plan çıkarımı için sadece iki çerçeve kullanılmıştır. Bunlardan ilki kişinin sahnede olmadığı sırada alınmış olan arka plan derinlik-renkli eşleştirilmiş görüntüsü, diğeri ise kameradan gelen anlık derinlik-renkli eşleştirilmiş görüntüdür. Anlık gelen eşleştirilmiş görüntü, veri seti ile ilgili bölümde belirtildiği üzere kendine özel bir iş parçacığı içerisinde hesaplanmaktadır.

Arka plan çıkarımından sonra elde edilen ikili görüntüde eşikleme değeri 10 olacak şekilde bir eşikleme yapılmış ve yeni bir ikili görüntü elde edilmiştir. Morfolojik operatörler ile, 5x5'lik bir çekirdek kullanılarak açma (opening) ve kapama (closing) bu ikili görüntüye uygulanmıştır. Elde edilen görüntüde Bağlı Bileşenler Analizi yöntemi ile cisimler ve alanları tespit edilmiş, en büyük cisim kişi olarak kabul edilip, bu parça yeni bir ikili görüntüye aktarılmıştır. Diğer bölgeler gürültü olarak kabul edilmiştir. Bu aşamalar Şekil 54'te gösterilmiştir.

Elde edilen son ikili görüntü hem derinlik görüntüsü hem de renkli görüntüyle aynı koordinatlara sahip olduğu için, buradaki el tespiti iki görüntüde de ellerin tespit edilmesini sağlamaktadır. Kamera konumu gereği, el yıkayan kişi her zaman elleri görüntüde en alt kısımda kalacak şekilde görülmektedir. İkili görüntüde lavabo kısmına denk düşen en düşük y koordinatlı beyaz piksel, ellerin konumunu verecektir. Bu nokta etrafına çizilen bir kare ile, veri seti bölümünde anlatıldığı gibi ROI'ler belirlenmiştir. Bu yaklaşımın avantajı, oldukça hızlı olmasının yanı sıra doğruluk oranı yüksek ve ellerin konumunu takip edebilecek şekilde olmasıdır. Kişi ellerini lavabonun farklı kısımlarında yıkasa bile bu yöntem onu takip edebilmektedir. Şekil 54'te ellerin tespitiyle ilgili örnek bulunmaktadır. Solda orijinal görüntü, soldan ikinci arka plan çıkarımını ve eşikleme sonrasını, soldan

üçüncü morfoloji sonrasını, soldan dördüncü Bağlı Bileşenler Analizi sonrasını göstermektedir.



Şekil 54. Derinlik görüntülerinin ön işlemden geçirilmesi

2.3.6. Kamera Veri Setinde Sınıflandırma

Elde ettiğimiz renkli iki boyutlu görüntüler ve renksiz üç boyutlu noktalar kümeleri, sınıflandırma için kullanabileceğimiz veri setini oluşturmuş olmaktadır. Bu verileri sınıflandırmak için çeşitli yapay sinir ağları mimarileri ve yöntemleri kullanılabilir. El yıkama hareketi, zaman içerisinde gerçekleşen ve başlangıcı ve bitişi olan hareketlerden oluştuğu için (temporal), zaman akışındaki hareketleri sınıflandırabilen RNN temelli LSTM ağlarının kullanılması doğru olacaktır. LSTM ağı, belirli bir süre veya çerçeve miktarına ait görüntülerle eğitildiği takdirde, belirlenen süreye sahip hareketleri tespit etmekte kullanılabilmektedir.

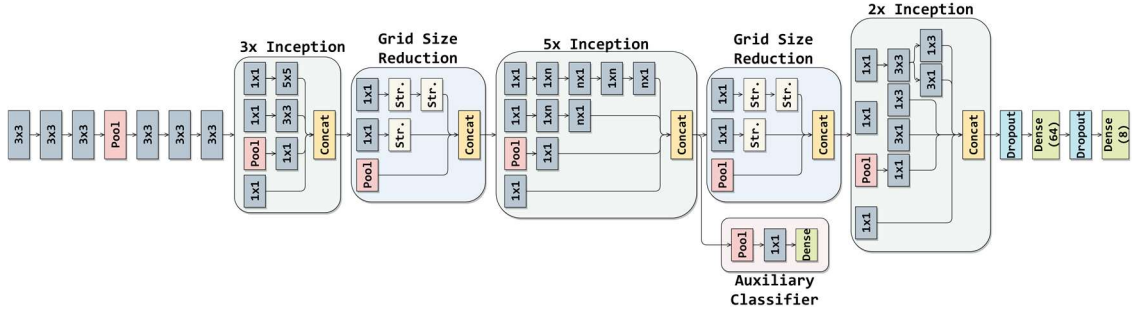
LSTM ağı girdi olarak doğrudan görüntüleri kullanmaz. LSTM girdileri oluşturmak için, sahip olduğumuz görüntülerin özniteliklerinin çıkarılması ve bu özelliklerin ağa sunulması gerekmektedir. Öznitelik çıkarmak için kullanılan en popüler yöntemlerden biri de VGG-16 ağlarıdır. Bu ağlar, kendilerine herhangi boyuttaki bir görüntüye ait öznitelikleri çıkarıp, 512 öznitelik içeren bir vektör sunmaktadır. Bu öznitelik vektörleri, her bir çerçeve için çıkarılıp, ilgili hareketin süresi miktarınca bir dizi şekline getirilip, LSTM ağlarına girdi oluştururlar. Saniyede 10-12 çerçeve oranıyla elde edilen bir görüntüde, 5 saniyelik bir görüntü yaklaşık 60 çerçeveye tekabül edecektir. 60 çerçeve için VGG-16 ağı ile çıkarılan öznitelik dizisi LSTM ağına sunulur ve ağ eğitilmiş olur. LSTM ağlarının sahip olduğu kısıtlama da burada ortaya çıkmaktadır, doğru bir sınıflandırma için, tespit edilecek hareketlerin zaman içerisinde bölünmeden ve beklenen miktarda olması gerekmektedir.

RNN ağlarından farklı olarak, CNN ağları da hareket tanımlamak için kullanılabilir. CNN ağları, LSTM gibi bir zamansal boyuta sahip değildir. Elde bulunan tek bir görüntü çerçevesi üzerinden sınıflandırma için kullanılır. Bir harekete ait tüm çerçeveler kullanılarak

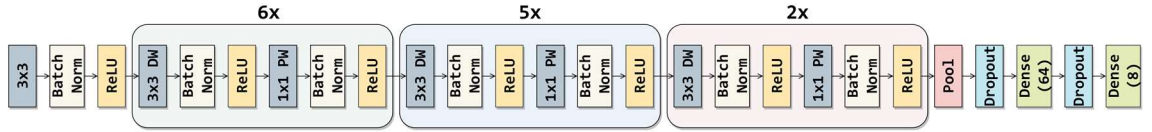
bir sınıflandırıcı eğitildiğinde, gerçek zamanlı görüntü esnasında elde edilen çerçevelerde anlık olarak hangi hareketin içinde bulunduğu tespit edilebilir. Tez senaryosuna uygun olarak, bir el yıkama hareketinin kaç saniye boyunca sürdüğü, bir CNN bazlı sınıflandırıcı yardımıyla hesaplanabilir. LSTM ağlarındaki hareket süresi dezavantajı CNN ile bertaraf edilmiş olacaktır.

Tez çalışmasında kullanılan ağların yapıları aşağıdaki şekillerde gösterilmiştir. Şekil 55'te, Inception-V3 ağının yapısı görülmektedir. Str. Kelimesiyle kısaltılan katman Stride katmanıdır. Şekil 56'da MobileNet-V1 ağının yapısı görülmektedir. Şekil 57'de ResNet-50 ağının yapısı görülmektedir. Şekil 58'de PointNet ağının yapısı görülmektedir. Şekil 59'da PointNet – LSTM ağının yapısı görülmektedir. Şekil 60'ta VGG-16 – LSTM ağının yapısı görülmektedir. Şekil 61'de öznitelik füzyonu için kullanılan ağların yapısı görülmektedir.

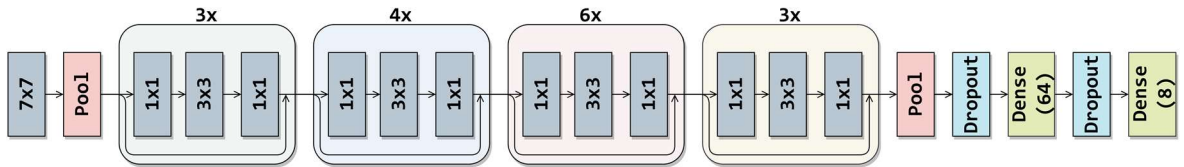
Ağlarda kullanılan parametreler, veri setinin bir bölümü ve tamamı kullanılarak yapılan eğitim denemelerinde elde edilen doğruluk oranı / kayıp gözlemlenerek seçilmiş ve en iyi değerler elde edilmeye çalışılmıştır.



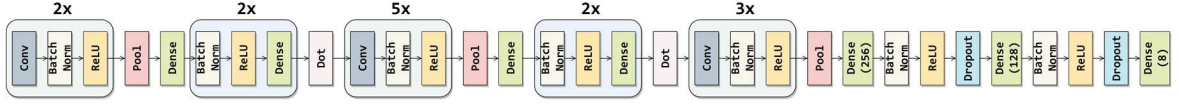
Şekil 55. Inception-V3 ağının yapısı



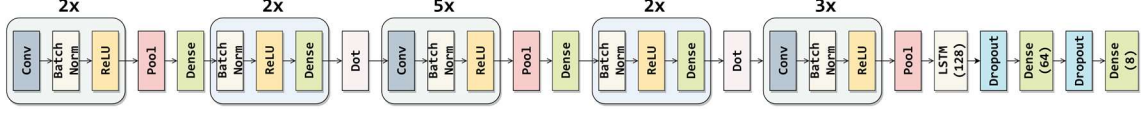
Şekil 56. MobileNet-V1 ağının yapısı



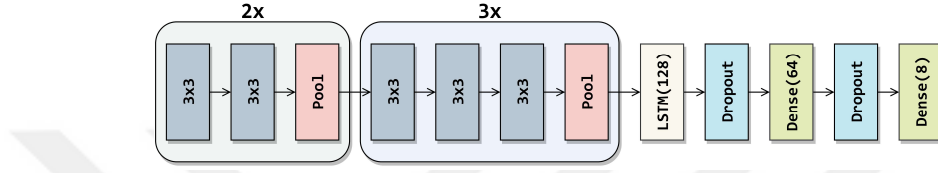
Şekil 57. ResNet-50 ağının yapısı



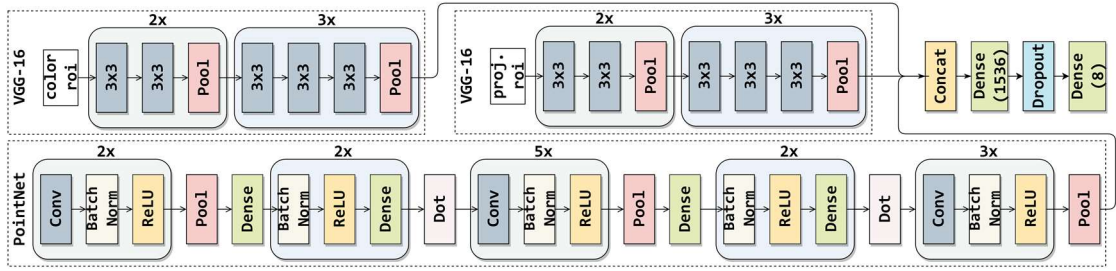
Şekil 58. PointNet ağının yapısı



Şekil 59. PointNet – LSTM ağının yapısı



Şekil 60. VGG-16 – LSTM ağının yapısı



Şekil 61. Öznitelik füzyonu için kullanılan üç farklı ağ yapısı

- RGB Görüntüleriyle Çerçeve Bazlı Sınıflandırma: El bölgesini içeren RGB görüntülerle Inception-V3 ağı, 96x96x3'lük görüntülerle eğitilmiştir. Orijinal boyutu 150x150x3 olan renkli görüntüler, daha verimli olması için 32'nin katı olarak boyutlandırılmıştır. Inception ağının son katmanları dahil edilmemiş, bunun yerine ağa sırasıyla overfitting durumunu engellemek için 0,2 parametre değerine sahip Dropout katmanı, 64 nörondan oluşan Dense katmanı, yine bir Dropout katmanı ve 8 hareketi sınıflandırması için 8 nörondan oluşan Dense katmanı eklenmiştir. Pooling parametresi ortalama (avg) değeriyle, ağın ağırlık matrisi ise rastgele olarak belirlenmiştir. Keras için batch size varsayılan olan 32 olarak belirlenmiştir. Ağın toplam parametre sayısı 21.934.440 olmuştur. MobileNet-V1 ve ResNet-50 ağları da Inception-V3 ağında anlatıldığı gibi aynı parametrelerle kurulmuş ve eğitilmiştir. Dropout → Dense (64) → Dropout → Dense (8) katmanları burada da aynı şekilde uygulanmıştır. MobileNet-V1 ağının toplam

parametre sayısı 3.294.984 olmuştur. ResNet-50 ağının toplam parametre sayısı 23.719.368 olmuştur.

- **Projeksiyon Görüntüleriyle Çerçeve Bazlı Sınıflandırma:** Veri seti bölümünde anlatıldığı üzere, noktalar kümesi tipi verilerin farklı kamera açılarıyla render edilmesiyle ve birleştirilmesiyle elde edilen görüntüler tıpkı RGB görüntüler gibi bir CNN tarafından sınıflandırma için kullanılabilir. Bunun için Inception-V3 ağı, 128x128x3'lük görüntülerle eğitilmiştir. Orijinal boyutu 300x300x3 olan bu görüntüler daha verimli olması için 32'nin katı olarak boyutlandırılmıştır. Inception-V3 RGB ile aynı mimari ve parametreler kullanılmıştır. Ağın toplam parametre sayısı da RGB ağıyla aynı olmuştur. ResNet-50 ve MobileNet-V1 ağları da 128x128x3'lük görüntülerle aynı parametrelerle kurulmuştur. Ağların toplam parametre sayıları RGB ağlarıyla aynı olmuştur.
- **Noktalar Kümesi Verisi ile Çerçeve Bazlı Sınıflandırma:** Veri setinde daha önce elde edilen her bir derinlik çerçevesine ait noktalar kümesi verisi, sekiz yıkama hareketi sınıfından birine ait bir obje gibi kabul edilerek, hareket sınıflandırmada kullanılabilir. Bu amaçla, TensorFlow Keras'ta bulunan PointNet implementasyonu (URL-24, 2023) kullanılarak, mevcut her el hareketi çerçevesinden oluşturulmuş noktalar kümesinden, 512 nokta rastgele örneklem olarak (sampling) seçilerek, bir PointNet ağı eğitilmiştir. Dropout ve Dense katmanları, RGB ve projeksiyon görüntülerinin ağlarındaki yapıyla aynı şekilde ağın sonuna eklenmiştir. Ağın toplam parametre sayısı 748.721 olmuştur.
- **RGB ve Projeksiyon Görüntüleriyle Zamansal Bazlı Sınıflandırma:** Varsayılan olarak ImageNet (Deng vd., 2009) ağırlıklarına sahip bir VGG-16 ağına, 96x96x3 boyutunda RGB görüntüleri girdi olarak verildiğinde, her bir görüntü için 1x512 boyutunda bir öznitelik vektörü elde edilmektedir. Her bir hareket süresi boyunca öznitelik vektörleri çıkarılmış olan tüm bu çerçeveler LSTM ağının girdisi olarak tanımlanmıştır (örneğin, 40 çerçeve için 40x1x512). Böylece, aynı uzunlukta gelecek test verileri VGG-16-LSTM kombinasyonu ile sınıflandırılabilir. RGB görüntülerinde beş saniyelik görüntüler içerisinde en düşük ortak çerçeve oranı 40 olarak belirlenmiştir. Dolayısıyla her hareket için 40 çerçevelik RGB görüntüleri, VGG-16 ağının öznitelik çıkarma katmanlarına ek olarak 128 hücre içerecek şekilde tanımlanmış LSTM katmanı ve diğer ağlardaki Dropout ve Dense katmanları içeren bir ağ sınıflandırma için kullanılmıştır. VGG-16 kısmı hariç

(eğitim gerektirmediyinden), ağın toplam parametre sayısı 336.968 olmuştur. RGB görüntülerin eğitildiği VGG-16-LSTM ağına benzer olarak, 128x128x3 boyutlarına sahip projeksiyon görüntüleri ile de sınıflandırılma yapılmıştır. Projeksiyon görüntülerinde en düşük ortak çerçeve oranı 120 olarak belirlenmiştir.

- **Noktalar Kümesi ile Zamansal Bazlı Sınıflandırma:** PointNet ağına, noktalar kümesi verileriyle zamana yayılmış bir hareketi temsil etmenin bir yolu, RGB ve projeksiyon görüntüleriyle olduğu gibi, peş peşe veri dizileri tanımlamak ve LSTM gibi bir ağ katmanıyla kullanmak olabilir. Bu amaçla, önceden eğitilmiş bir PointNet ağı bir öznitelik çıkarıcı (feature extractor) olarak kullanılıp, elde edilen öznitelik dizileriyle bir PointNet-LSTM ağı kombinasyonu eğitilebilir. Bunun için, daha önceden veri setindeki verilerle eğitilmiş bir PointNet ağı, 35. katmanından itibaren kesilmiş ve ardına LSTM katmanı ve Dense (64), Dropout ve Dense (8) katmanları yerleştirilmiştir. PointNet ağı hariç (o kısmın parametre sayısı, çerçeve bazlı sınıflandırma ağıyla aynı olacaktır) ağın toplam parametre sayısı 336.968 olmuştur.
- **Noktalar Kümesi Birleştirme ile Sınıflandırma:** Noktalar kümesi verileri tek tek her bir çerçeve için elde edildiği gibi, topluca tüm hareket için elde edilebilir. Daha önce elde edilme yöntemi açıklanan birleştirilmiş noktalar kümesi şeklinde temsil edilen verileri, hareketin bütünü bir yansıması olarak kabul edersek, bu veriler obje sınıflandırma kategorisinde değerlendirilebilir. Dolayısıyla, PointNet ağıyla çerçeve bazlı sınıflandırmayla aynı yapıya sahip olmak üzere, birleştirilmiş noktalar kümesi verilerinden oluşan bir veri setiyle eğitilebilecek bir model oluşturulmuştur. Farklı olarak, örneklem sayısı 12.800 olduğu için giriş katmanı bu şekilde verilere uygun hale getirilmiştir. Ağın toplam parametre sayısı 748.721 olmuştur.
- **Öznitelik Füzyonu ile Sınıflandırma:** RGB, projeksiyon ve noktalar kümesi verilerinin hibrit bir biçimde sınıflandırmada başarısını ölçmek için, bu üç veri tipinden de elde edilen öznitelikler birleştirme (fusion) yöntemiyle basit düzeyde bir yapay sinir ağı katmanını eğitmek için kullanılmıştır. İki adet VGG-16 ağı (RGB ve projeksiyon veriler için) ve eğitilmiş bir çerçeve bazlı PointNet ağı öznitelik çıkarmak için kullanılmıştır. Her bir ağ, kullanıldığı haliyle son katmanında 512 adet öznitelik vektörü sunmaktadır. Bu öznitelik vektörleri birleştirilip 1.536 boyutunda bir öznitelik vektörü elde edilmiştir. Her bir hareket oturumu için, RGB, projeksiyon ve noktalar kümesi veri sayıları, çerçeve oranından dolayı farklı olduğu

için, görüntü seçerken bir aralıktaki değeri diğerine haritalamak amacıyla Formül 31'den yararlanılmıştır. Bir (a, b) aralığında olan x sayısının, (c, d) aralığında denk düştüğü sayısal değer bulunmak isteniyorsa bu formülden yararlanılır. Verilerden birleştirmek amacıyla öznitelik çıkarılırken, alınan kayıt oturumları genelinde RGB görüntülerinde en düşük 40, projeksiyon ve noktalar kümesi verilerinde en düşük 120 çerçeve miktarı bulunduğu için, RGB verisi 1-40 aralığında seçilirken, seçilen örnek indeksinin 1-120 aralığındaki karşılığı verilen formülle elde edilerek projeksiyon ve noktalar kümesi verisi seçilmiştir. Daha fazla kare içeren oturumlarda bu indeks dışındaki kareler dahil edilmemiştir.

$$y = (x - a) \frac{(d - c)}{(b - a)} + c \quad (31)$$

2.3.7. Sentetik Veri Setinde Sınıflandırma

Tüm Inception-V3 ağlarında, görüntü boyutları 256x256x3 olarak belirlenmiştir. Inception-V3 ağının varsayılan katmanlarına ek olarak ağın son katmanlarında 0,2 parametrelili Dropout, 64 nöronlu Dense, Dropout ve 4 sınıf için 4 nöronlu Dense katman kullanılmıştır. Aktivasyon fonksiyonu olarak son katmanda Softmax, bir önceki sıkı katmanda ReLU kullanılmıştır. 8 sınıflı Inception-V3 ağının mimarisi de 4 sınıflı ile aynı olmakla beraber, dropout değeri 0,3 olarak seçilmiş ve son sıkı katman 8 nöronla oluşturulmuştur. 32 sınıflı Inception-V3 ağında da mimari 4 ve 8 sınıfla aynı olmakla beraber dropout değeri 0,2 seçilmiş ve son sıkı katmanın nöron sayısı 32 olarak belirlenmiştir. Dropout katmanları ve değerleri deneysel olarak daha iyi sonuç veren değerlerden seçilmiştir. Pooling parametreleri ortalama (avg) olarak belirlenmiştir. Başlangıç ağırlıkları olarak ImageNet ağırlıkları seçilmiş, optimize edici olarak Adam algoritması 0,001 öğrenme oranıyla kullanılmıştır. 4 sınıflı ağın parametre sayısı 21.934.180, 8 sınıflı ağın parametre sayısı 21.934.440, 32 sınıflı ağın parametre sayısı ise 21.936.000 olmuştur.

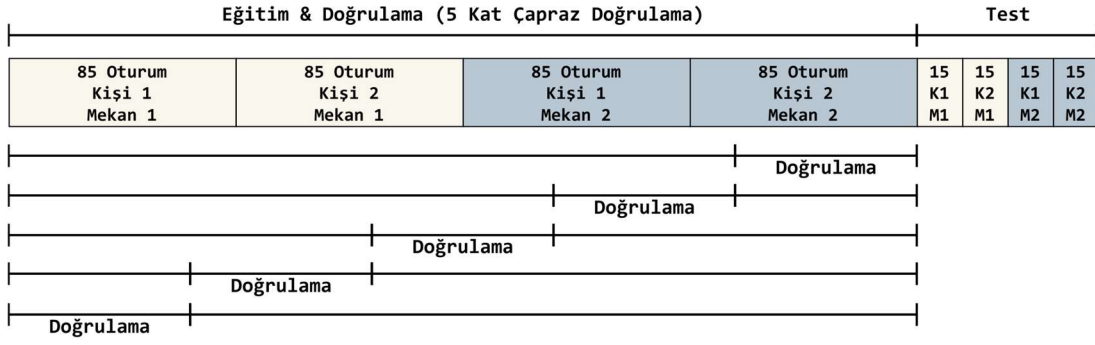
YOLO-8n ağının 4 sınıflı olanı varsayılan parametrelerle kullanılmıştır. 8 ve 32 sınıflı ağda, eğitim sırasında kullanılan ve veriyi sentetik biçimde artırmaya yarayan, görüntülerin yatay ve düşey simetriğinin kullanılması seçenekleri (flip) devre dışı bırakılmıştır. Çünkü hareket sınıflandırmada, hareketler birbirinin simetriği olduğundan tüm görüntülerin simetriğinin eğitim için kullanılması olumsuz etkiye neden olabilecektir.

YOLO modellerinde Inception-V3'ten farklı olarak, kırılmış el bölgesi görüntüleri değil, veri setinin orijinal kamera görüntüleri kullanıldığından, el bölgesi doğru olarak tespit edilen çerçeve sayısı da hesaplanmıştır. YOLO modeli tüm görüntülerde el bölgesini tespit edememiş ve bir sınıf belirtmemiştir. El bölgesi tespit oranını ölçmek için, beş modelin en az dördünün doğru tespit yaptığı durumlar başarılı olarak değerlendirilmiştir. 4 sınıflı modelde el tespiti 155.102 çerçeve içerisinde 109.280 çerçeve olmuştur. 8 sınıflı modelde el tespiti 155.102 çerçeve içerisinde 98.508 olmuştur. 32 sınıflı modelde el tespiti 155.102 çerçeve içerisinde 101.725 olmuştur. 4 sınıflı modelde el tespiti %70, 8 sınıflı modelde %63, 32 sınıflı modelde %65 olmuştur.

2.3.8. Ağların Eğitimi

Ağlar eğitilirken kamera veri setinde oluşturulan doğrulama ve test dağılımı Şekil 62'de gösterilmiştir. Her harekette bulunan 400 oturumun 340'ı eğitim ve doğrulama, 60'ı test için ayrılmıştır. Beş katlı çapraz doğrulama işlemi (5-fold cross validation) tüm ağların eğitiminde uygulanmıştır. Bu doğrulama yöntemi, ağıdaki eğitim verilerinin her seferinde farklı bir kısmını, doğrulama verisi olarak ayırıp yeni bir model eğitir. Modelin başarısı, elde edilen beş farklı modelin başarılarının ortalaması olarak hesaplanır. Böylece, veri setinde modele etki edebilecek düzensizlikler minimuma indirilmiş olur.

Tüm çerçeve bazlı RGB görüntülerinin ağları ve çerçeve bazlı noktalar kümesi ağı 10 epok (epoch) kadar eğitilmiştir. Projeksiyon görüntülerinin ağları 85 epok kadar eğitilmiştir. Veri fazlalığı sebebiyle tüm verilerin ağı eğitmek için önbelleğe alınamaması sonucunda, projeksiyon görüntüleri parça parça ağı eğitim olarak verilmiştir. Her beş oturumun bir oturumu, doğrulama verisi olarak ve kalan dört oturumu eğitim verisi olarak ağı girdi olarak verilmiştir. Daha sonra bu oturumlar kaydırılarak tüm veri setinin kapsanması sağlanmıştır. Beş katlı doğrulama gerçekleştirebilmek için, eğitim ve doğrulama oturumları her modelde farklı bloklara kaydırılmıştır. Tüm LSTM katmanını içeren ağlar 50 epok kadar eğitilmiştir. Noktalar kümesi toplamı ağı ise 25 epok kadar eğitilmiştir. Epok miktarları, ağı doğrulama (validation) ve doğrulama kaybı (validation loss) değerlerinin durumuna göre gözlemle belirlenmiştir.



Şekil 62. Kamera veri setinin eğitim – doğrulama – test dağılımı

Tüm ağlarda, sondan bir önceki sıkı katmanda ReLU, son katmanda Softmax aktivasyon fonksiyonu kullanılmıştır. Yine tüm ağlarda Adam optimize edici 0,001 parametresiyle kullanılmıştır. Kayıp fonksiyonu (loss function) olarak, çoklu sınıfa ait veriler sınıflandırıldığı için kategorik çapraz entropi (categorical cross entropy) kullanılmıştır.

Sentetik veri setinde de eğitimler beş katlı çapraz doğrulama kullanılarak yapılmıştır. Inception-V3 ağlarında; 4 sınıflı modelde 7, 8 sınıflı modelde 5, 32 sınıflı modelde 10 epok eğitim yapılmıştır. YOLO-8n ağlarında 4 sınıflı ve 8 sınıflı modellerde 10 epok, 32 sınıflı modelde 25 epok eğitim yapılmıştır.

Tüm sınıflandırmalarda kullanılan ağlar, girdi boyutları, veri sayıları ve parametre sayıları Tablo 3'te özetlenmiştir.

Tablo 3. Sınıflandırmada kullanılan ağlar ve parametreleri

Veri Seti	Ağ	Girdi Boyutu	Veri Sayısı	Parametre Sayısı
Kamera	Inception-V3 RGB	96x96	155.242	21.934.440
	MobileNet-V1 RGB	96x96	155.242	3.294.984
	ResNet-50 RGB	96x96	155.242	23.719.368
	Inception-V3 Projeksiyon	128x128	465.238	21.934.440
	MobileNet-V1 Projeksiyon	128x128	465.238	3.294.984
	ResNet-50 Projeksiyon	128x128	465.238	23.719.368
	PointNet Çerçeve	1x512	465.238	748.721
	PointNet-LSTM	120x512	3.200	336.968
	PointNet Birleşik	12.800	400	748.721
	VGG-16-LSTM RGB	40x512	3.200	336.968
	VGG-16-LSTM Projeksiyon	120x512	3.200	336.968
	Füzyon	1x1.536	3.200	2.373.128
Sentetik	Inception-V3 4 sınıf	256x256	15.872	21.934.180
	Inception-V3 8 sınıf	256x256	15.872	21.934.440
	Inception-V3 32 sınıf	256x256	15.872	21.936.000
	YOLO-8n	1920x1080	15.872	3.200.000

3. BULGULAR VE TARTIŞMA

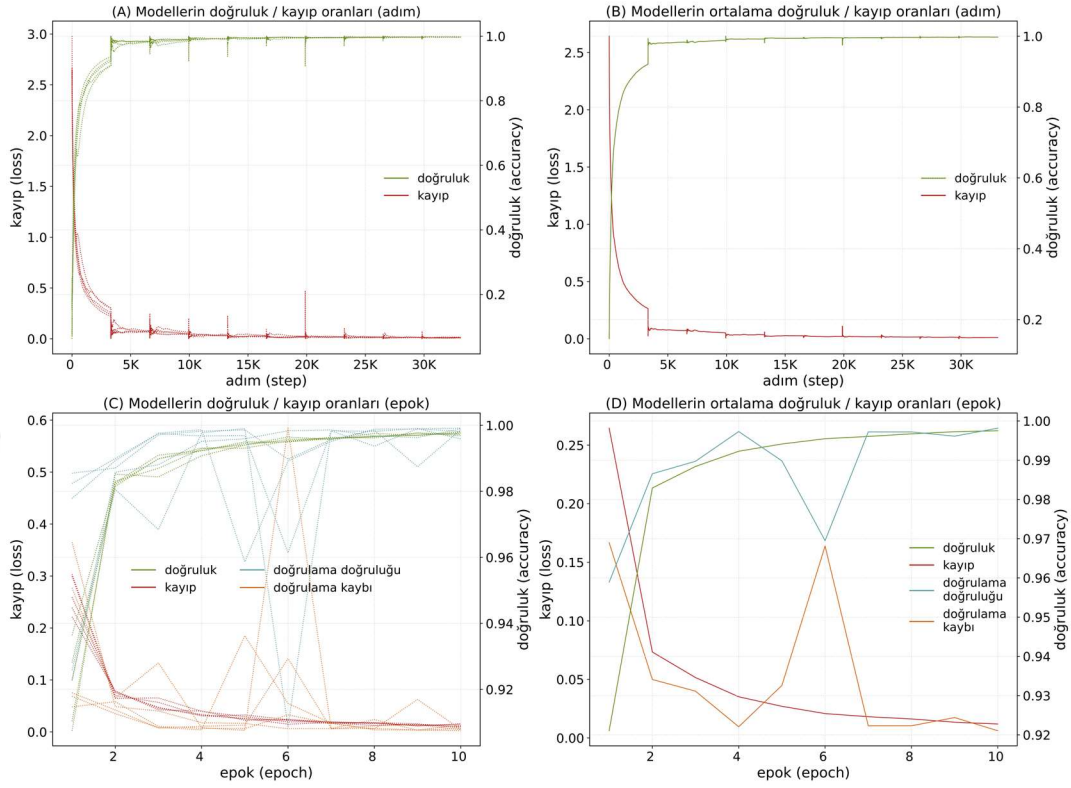
3.1. Kamera Veri Seti ile Sınıflandırma Sonuçları

Kamera görüntülerinden elde edilmiş veri setinde sınıflandırma için kullanılan 12 farklı ağı eğitimi süreçleri ve elde ettiği sonuçlar bu bölümde gösterilmiştir. Modellerin eğitim grafiklerinde yer alan (A) grafiğinde, her bir modele ait beş kat çapraz doğrulama modelinin adım (step) bazlı doğruluk ve kayıp oranları tek bir grafikte çizdirilmiştir. (B) grafiğinde, (A) grafiğindeki beş eğrinin ortalaması çizdirilmiştir. (C) grafiğinde, her bir modele ait beş kat çapraz doğrulama modelinin epok bazında doğruluk (accuracy), doğrulama doğruluğu (validation accuracy), kayıp (loss) ve doğrulama kaybı (validation loss) tek grafikte çizdirilmiştir. (D) grafiğinde (C) grafiğindeki beş eğrinin ortalaması çizdirilmiştir. Hata matrisi grafiklerinde, her bir modelde beş kat çapraz doğrulama modellerinin elde ettiği tüm hata matrislerinin ortalaması kusurlar dahil edilmeden gösterilmiştir.

Şekil 63, 65 ve 67’de, RGB çerçeve bazlı Inception-V3, MobileNet-V1 ve ResNet-50 modellerinin eğitim grafikleri görülmektedir. Şekil 64, 66 ve 68’de, bu modellerin hata matrisleri görülmektedir. Şekil 69, 71 ve 73’te projeksiyon çerçeve bazlı Inception-V3, MobileNet-V1 ve ResNet-50 modellerinin eğitim grafikleri görülmektedir. Şekil 70, 72 ve 74’te, bu modellerin hata matrisleri görülmektedir. Projeksiyon modellerinin eğitim grafiklerinde, daha önce bahsedilen, eğitim ve doğrulama verilerinin parçalanarak ağa girdi olarak verilmesinin sonucunda, doğrulama oranı ve kayıp eğrilerinde salınımlar oluşmuştur. Doğrulama verileri sayıca az olduğunda, veri seti yeterince temsil edilememekte, bu da hata eğrisinde salınım olarak yansıyabilmektedir.

Şekil 75’te, noktalar kümesi çerçeve bazlı PointNet modelinin eğitim grafikleri, Şekil 76’da ise bu modelin hata matrisi görülmektedir. Şekil 77’de RGB görüntüleri bazlı VGG-16 – LSTM modelinin eğitim grafikleri, Şekil 78’de ise bu modelin hata matrisi görülmektedir. Şekil 79’da projeksiyon görüntüleri bazlı VGG-16 – LSTM modelinin eğitim grafikleri, Şekil 80’de ise bu modelin hata matrisi görülmektedir. Şekil 81’de noktalar kümesi bazlı PointNet – LSTM modelinin eğitim grafikleri, Şekil 82’de ise bu modelin hata matrisi görülmektedir. Şekil 83’te birleştirilmiş noktalar kümesi bazlı PointNet modelinin

eğitim grafikleri, Şekil 84’te ise bu modelin hata matrisi görülmektedir. Şekil 85’te öznetelik füzyonu modelinin eğitim grafikleri, Şekil 86’da ise bu modelin hata matrisi görülmektedir.

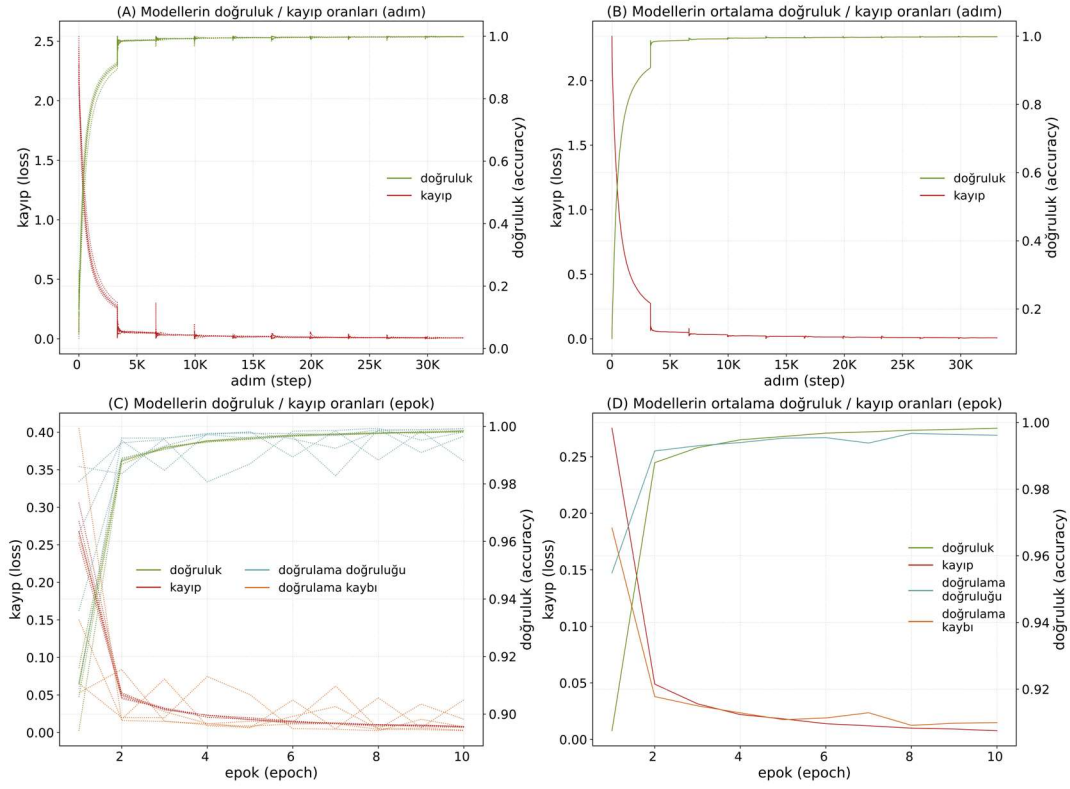


Şekil 63. Inception-V3 RGB çerçeve bazlı modelin eğitim grafikleri

1	3047 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	3 (0.00)	2842 (1.00)	0 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	2803 (0.99)	0 (0.00)	12 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	0 (0.00)	2774 (1.00)	5 (0.00)	2 (0.00)	0 (0.00)	0 (0.00)
5	0 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)	2760 (1.00)	3 (0.00)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	1 (0.00)	2806 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	2795 (1.00)	0 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)	2877 (1.00)
	1	2	3	4	5	6	7	8

Tahmin

Şekil 64. Inception-V3 RGB çerçeve bazlı sınıflandırmanın hata matrisi

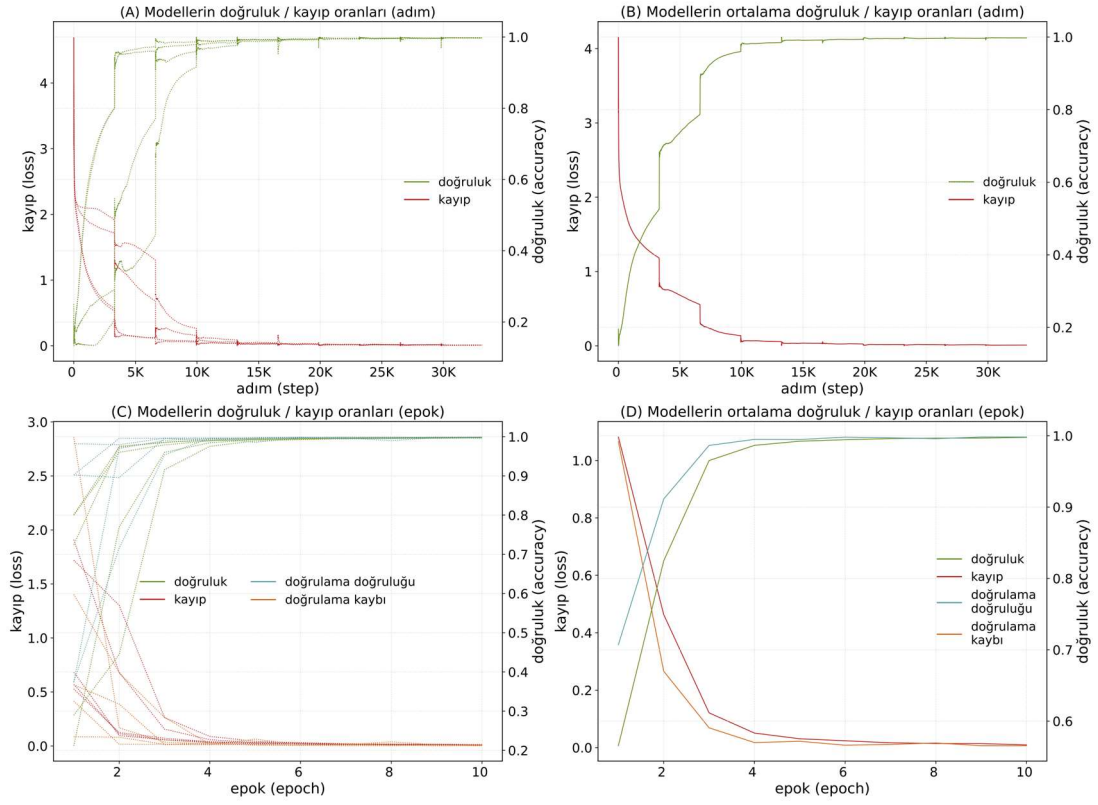


Şekil 65. MobileNet-V1 RGB çerçeve bazlı modelin eğitim grafikleri

1	3046 (1.00)	1 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	0 (0.00)	2841 (1.00)	0 (0.00)	0 (0.00)	5 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	2809 (1.00)	3 (0.00)	3 (0.00)	1 (0.00)	1 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	3 (0.00)	2759 (0.99)	6 (0.00)	10 (0.00)	4 (0.00)	0 (0.00)
5	1 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	2757 (1.00)	5 (0.00)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	2 (0.00)	2805 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	2 (0.00)	1 (0.00)	2791 (1.00)	0 (0.00)
8	0 (0.00)	1 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)	26 (0.01)	0 (0.00)	2849 (0.99)
	1	2	3	4	5	6	7	8

Tahmin

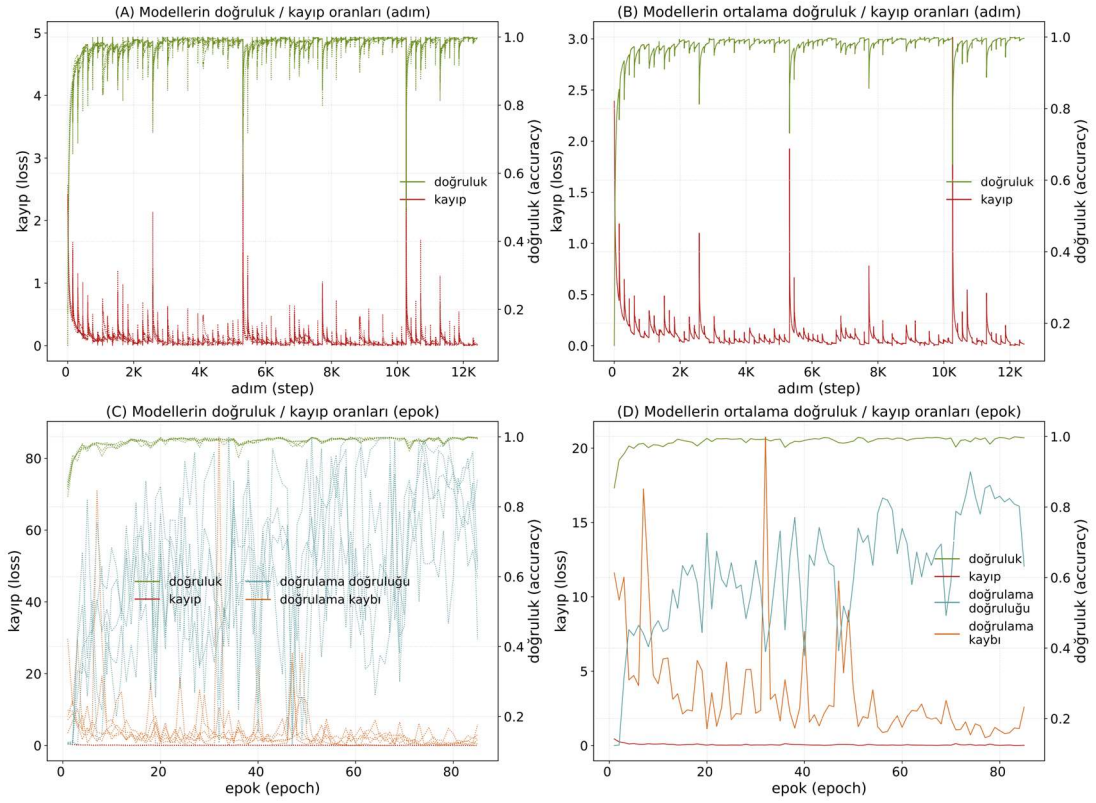
Şekil 66. MobileNet-V1 RGB çerçeve bazlı sınıflandırmanın hata matrisi



Şekil 67. ResNet-50 RGB çerçeve bazlı modelin eğitim grafikleri

1	3044 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	3 (0.00)	0 (0.00)	0 (0.00)
2	4 (0.00)	2824 (0.99)	0 (0.00)	0 (0.00)	12 (0.00)	6 (0.00)	0 (0.00)	1 (0.00)
3	0 (0.00)	0 (0.00)	2802 (0.99)	4 (0.00)	1 (0.00)	9 (0.00)	1 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	1 (0.00)	2773 (1.00)	4 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)
5	1 (0.00)	1 (0.00)	1 (0.00)	0 (0.00)	2742 (0.99)	16 (0.01)	2 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	2806 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	2 (0.00)	1 (0.00)	2792 (1.00)	0 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	2878 (1.00)
	1	2	3	4	5	6	7	8
	Tahmin							

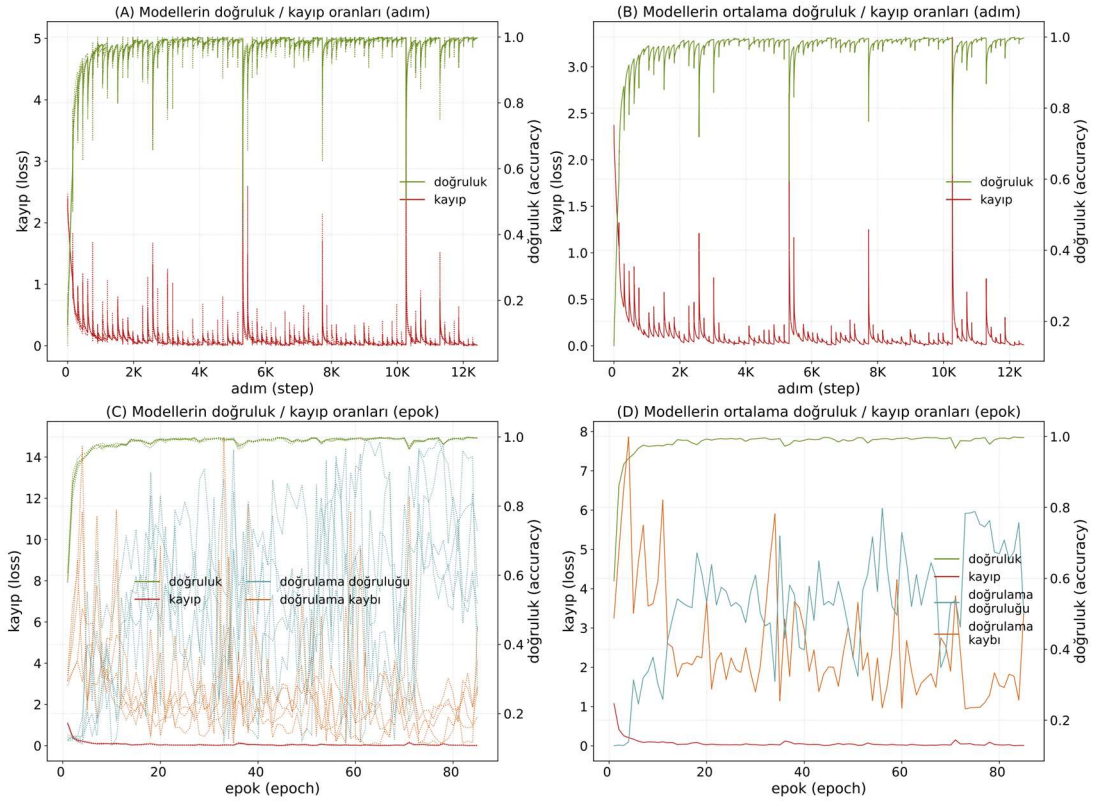
Şekil 68. ResNet-50 RGB çerçeve bazlı sınıflandırmanın hata matrisi



Şekil 69. Inception-V3 projeksiyon çerçeve bazlı modelin eğitim grafikleri

Hareket	1	8773 (0.98)	51 (0.01)	0 (0.00)	38 (0.00)	98 (0.01)	28 (0.00)	2 (0.00)	1 (0.00)
	2	16 (0.00)	8541 (0.99)	0 (0.00)	2 (0.00)	11 (0.00)	25 (0.00)	0 (0.00)	0 (0.00)
	3	120 (0.01)	68 (0.01)	6883 (0.80)	726 (0.08)	191 (0.02)	496 (0.06)	28 (0.00)	38 (0.00)
	4	57 (0.01)	6 (0.00)	29 (0.00)	7846 (0.93)	81 (0.01)	434 (0.05)	0 (0.00)	2 (0.00)
	5	369 (0.04)	414 (0.05)	0 (0.00)	23 (0.00)	7421 (0.89)	118 (0.01)	4 (0.00)	21 (0.00)
	6	686 (0.08)	226 (0.03)	0 (0.00)	136 (0.02)	4 (0.00)	7450 (0.88)	0 (0.00)	6 (0.00)
	7	249 (0.03)	72 (0.01)	44 (0.01)	11 (0.00)	373 (0.04)	140 (0.02)	7587 (0.89)	78 (0.01)
	8	27 (0.00)	765 (0.09)	0 (0.00)	10 (0.00)	33 (0.00)	945 (0.11)	3 (0.00)	6883 (0.79)
Tahmin		1	2	3	4	5	6	7	8

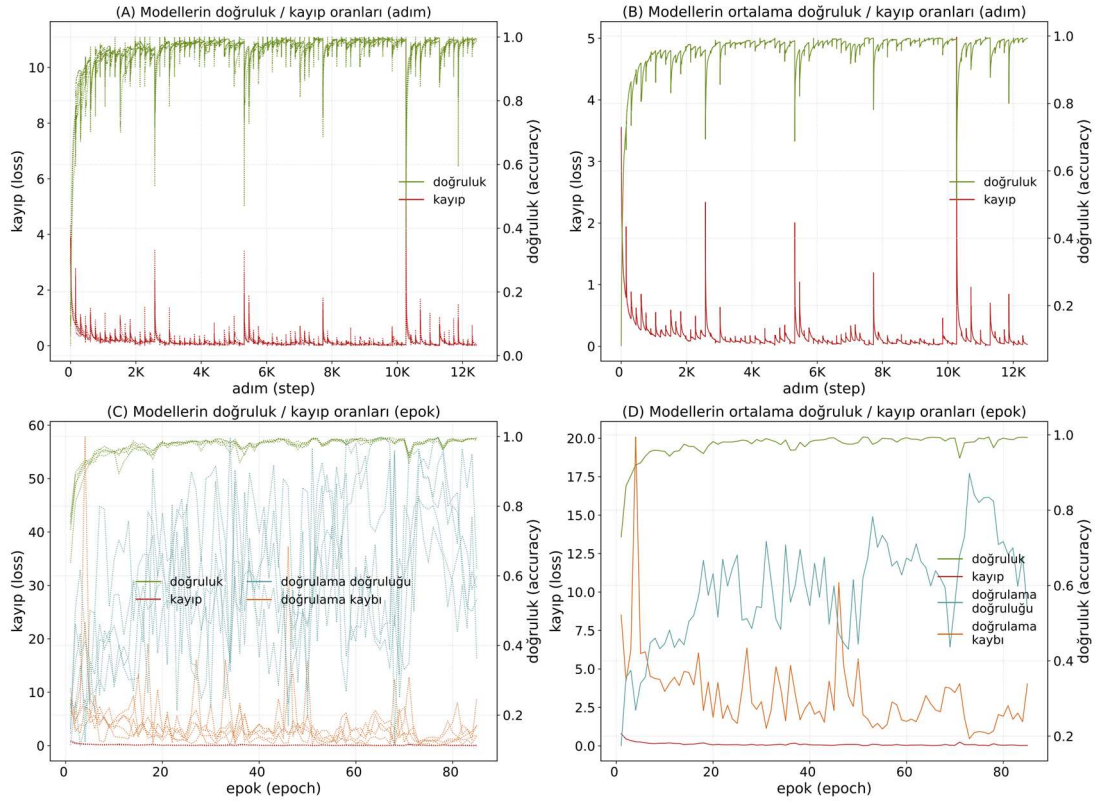
Şekil 70. Inception-V3 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi



Şekil 71. MobileNet-V1 projeksiyon çerçeve bazlı modelin eğitim grafikleri

Hareket	1	8616 (0.96)	111 (0.01)	0 (0.00)	103 (0.01)	33 (0.00)	74 (0.01)	55 (0.01)	0 (0.00)
	2	253 (0.03)	8199 (0.95)	0 (0.00)	7 (0.00)	20 (0.00)	114 (0.01)	1 (0.00)	1 (0.00)
	3	175 (0.02)	41 (0.00)	6759 (0.79)	787 (0.09)	702 (0.08)	75 (0.01)	7 (0.00)	4 (0.00)
	4	213 (0.03)	72 (0.01)	146 (0.02)	7172 (0.85)	558 (0.07)	286 (0.03)	2 (0.00)	7 (0.00)
	5	439 (0.05)	298 (0.04)	94 (0.01)	146 (0.02)	7321 (0.87)	69 (0.01)	3 (0.00)	1 (0.00)
	6	798 (0.09)	440 (0.05)	19 (0.00)	426 (0.05)	333 (0.04)	6481 (0.76)	7 (0.00)	1 (0.00)
	7	488 (0.06)	38 (0.00)	181 (0.02)	35 (0.00)	631 (0.07)	105 (0.01)	7060 (0.83)	16 (0.00)
	8	206 (0.02)	1487 (0.17)	6 (0.00)	371 (0.04)	197 (0.02)	401 (0.05)	2 (0.00)	5994 (0.69)
		1	2	3	4	5	6	7	8
		Tahmin							

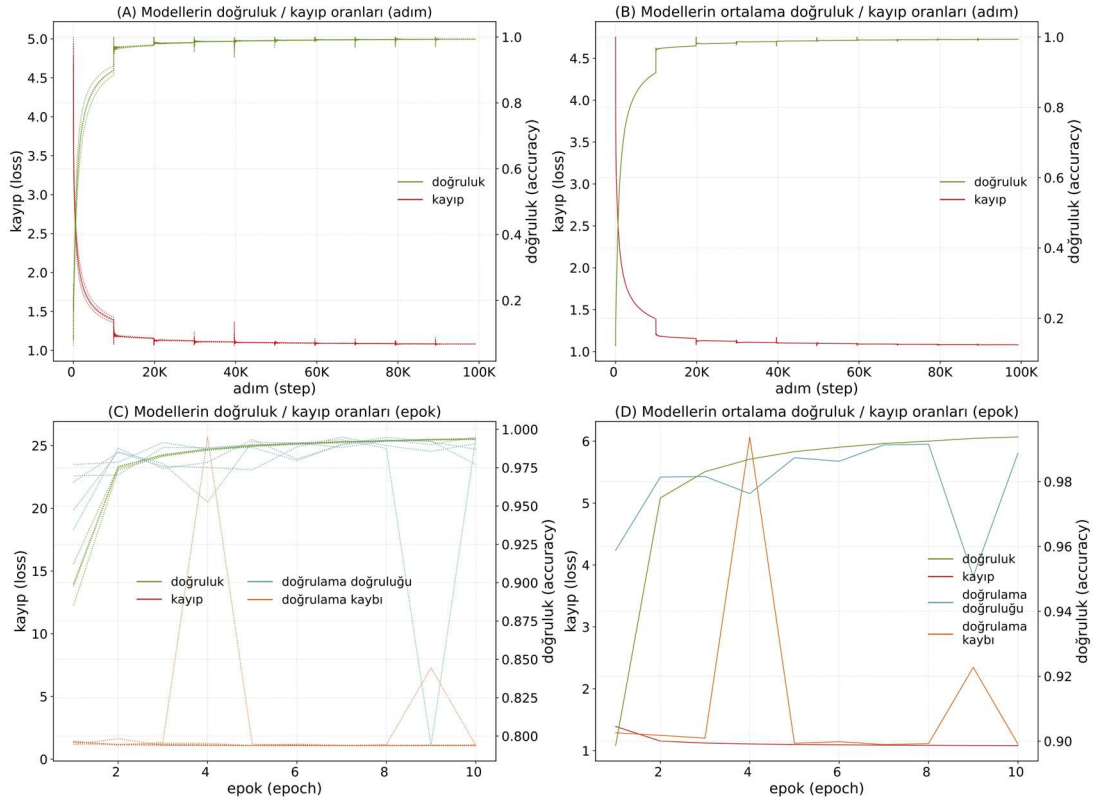
Şekil 72. MobileNet-V1 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi



Şekil 73. ResNet-50 projeksiyon çerçeve bazlı modelin eğitim grafikleri

Hareket	1	8453 (0.94)	34 (0.00)	5 (0.00)	1 (0.00)	309 (0.03)	165 (0.02)	23 (0.00)	0 (0.00)
	2	843 (0.10)	7496 (0.87)	0 (0.00)	0 (0.00)	158 (0.02)	71 (0.01)	0 (0.00)	27 (0.00)
	3	425 (0.05)	9 (0.00)	6655 (0.78)	436 (0.05)	875 (0.10)	94 (0.01)	6 (0.00)	49 (0.01)
	4	121 (0.01)	22 (0.00)	306 (0.04)	6768 (0.80)	864 (0.10)	325 (0.04)	10 (0.00)	40 (0.00)
	5	284 (0.03)	28 (0.00)	43 (0.01)	0 (0.00)	7924 (0.95)	85 (0.01)	4 (0.00)	2 (0.00)
	6	1334 (0.16)	39 (0.00)	186 (0.02)	20 (0.00)	793 (0.09)	6123 (0.72)	1 (0.00)	9 (0.00)
	7	336 (0.04)	10 (0.00)	125 (0.01)	17 (0.00)	646 (0.08)	249 (0.03)	7096 (0.83)	75 (0.01)
	8	205 (0.02)	1062 (0.12)	6 (0.00)	30 (0.00)	380 (0.04)	1486 (0.17)	95 (0.01)	5400 (0.62)
		1	2	3	4	5	6	7	8
		Tahmin							

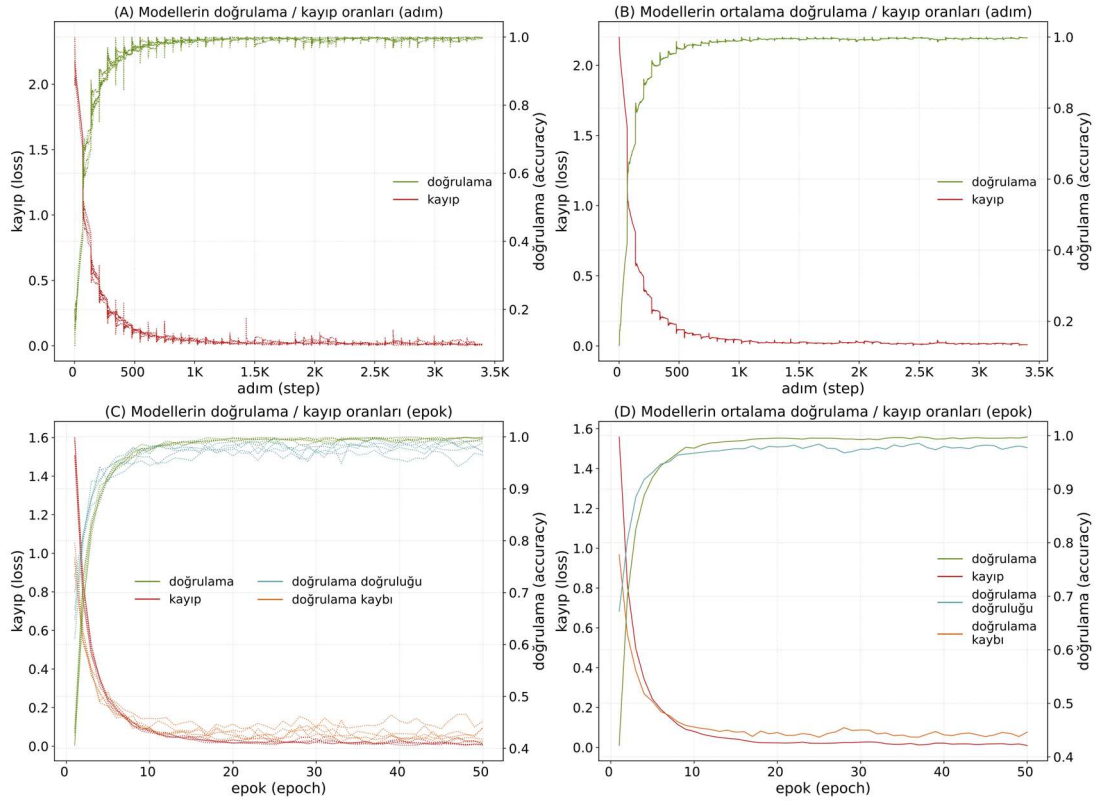
Şekil 74. ResNet-50 projeksiyon çerçeve bazlı sınıflandırmanın hata matrisi



Şekil 75. PointNet çerçeve bazlı modelin eğitim grafikleri

	1	2	3	4	5	6	7	8
1	8980 (1.00)	2 (0.00)	0 (0.00)	0 (0.00)	3 (0.00)	3 (0.00)	4 (0.00)	0 (0.00)
2	32 (0.00)	8546 (0.99)	0 (0.00)	0 (0.00)	1 (0.00)	11 (0.00)	1 (0.00)	5 (0.00)
3	0 (0.00)	0 (0.00)	8385 (0.98)	112 (0.01)	11 (0.00)	5 (0.00)	39 (0.00)	0 (0.00)
4	2 (0.00)	0 (0.00)	25 (0.00)	8359 (0.99)	52 (0.01)	6 (0.00)	12 (0.00)	0 (0.00)
5	13 (0.00)	1 (0.00)	6 (0.00)	15 (0.00)	8233 (0.98)	94 (0.01)	8 (0.00)	2 (0.00)
6	9 (0.00)	173 (0.02)	8 (0.00)	9 (0.00)	27 (0.00)	8278 (0.97)	3 (0.00)	2 (0.00)
7	0 (0.00)	0 (0.00)	35 (0.00)	11 (0.00)	18 (0.00)	5 (0.00)	8476 (0.99)	9 (0.00)
8	1 (0.00)	75 (0.01)	1 (0.00)	32 (0.00)	7 (0.00)	33 (0.00)	8 (0.00)	8509 (0.98)
	1	2	3	4	5	6	7	8
	Tahmin							
	Hareket							

Şekil 76. PointNet çerçeve bazlı sınıflandırmanın hata matrisi

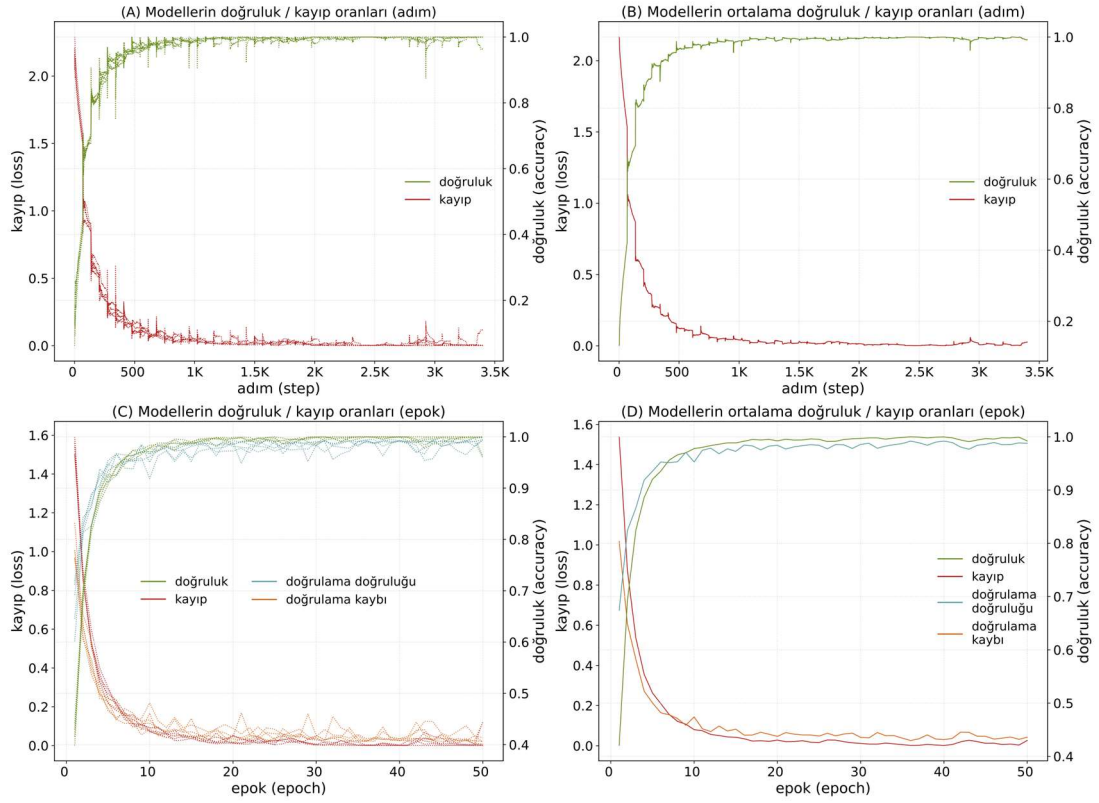


Şekil 77. VGG-16 - LSTM RGB modelinin eğitim grafikleri

1	59 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	7 (0.12)	52 (0.87)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	54 (0.90)	5 (0.09)	0 (0.00)	0 (0.01)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	2 (0.04)	57 (0.95)	0 (0.00)	0 (0.01)	0 (0.00)	0 (0.00)
5	0 (0.00)	0 (0.01)	0 (0.00)	0 (0.00)	51 (0.87)	6 (0.11)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	58 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	58 (0.97)	1 (0.03)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.01)	2 (0.04)	57 (0.95)
	1	2	3	4	5	6	7	8

Tahmin

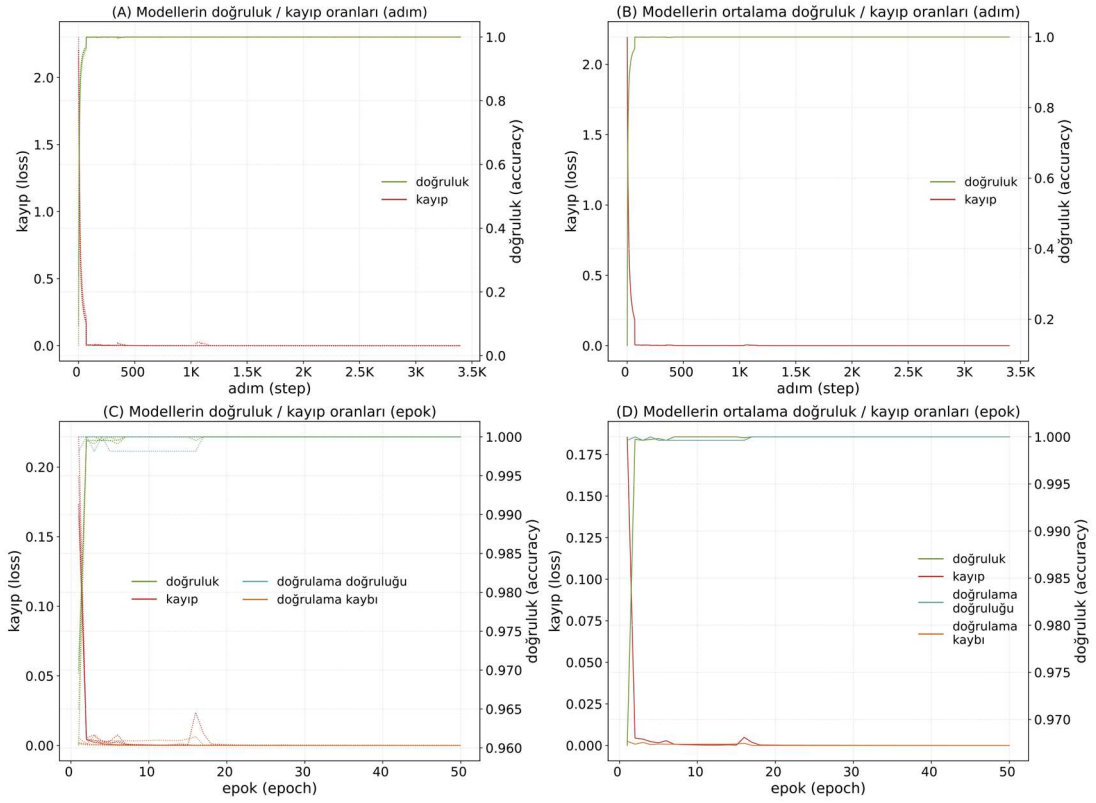
Şekil 78. VGG-16 - LSTM RGB sınıflandırmanın hata matrisi



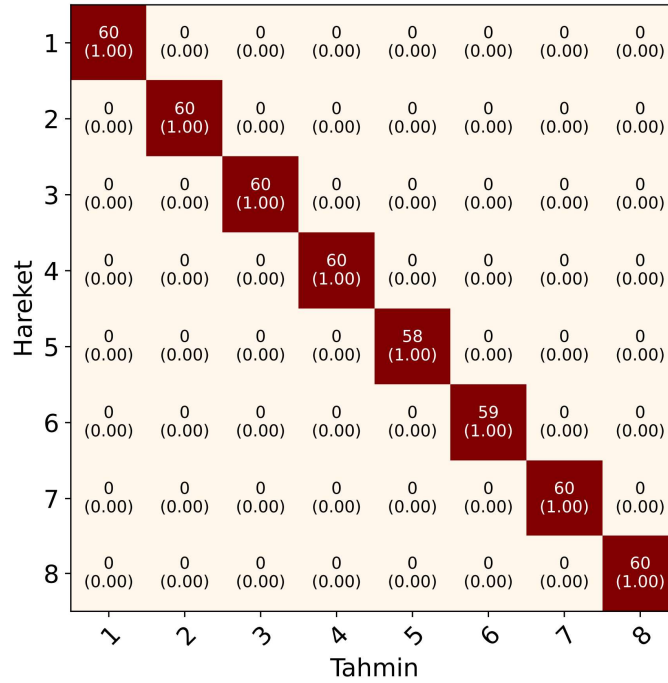
Şekil 79. VGG-16 - LSTM projeksiyon modelinin eğitim grafikleri

1	58 (0.98)	1 (0.02)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	1 (0.02)	58 (0.97)	0 (0.00)	0 (0.00)	0 (0.01)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	58 (0.97)	1 (0.02)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	0 (0.00)	60 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
5	0 (0.01)	0 (0.01)	1 (0.02)	0 (0.00)	54 (0.94)	0 (0.01)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	57 (0.97)	0 (0.00)	1 (0.02)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	59 (1.00)	0 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	60 (1.00)
	1	2	3	4	5	6	7	8
	Tahmin							

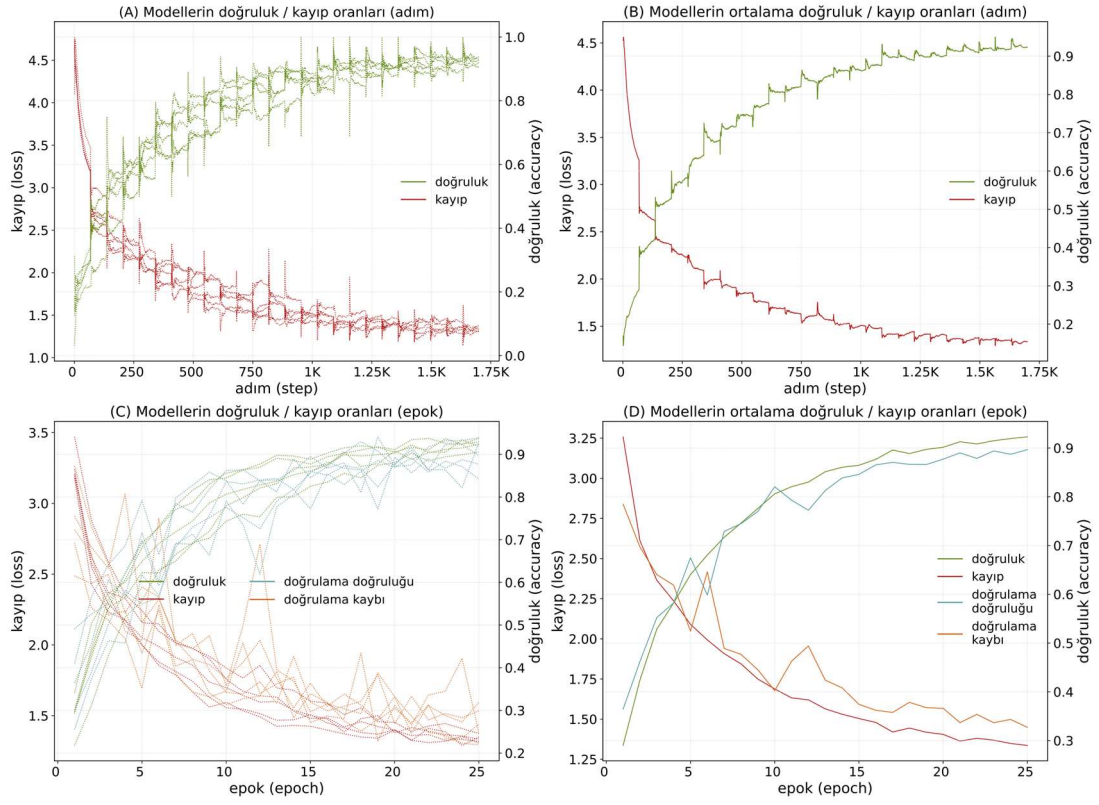
Şekil 80. VGG-16 - LSTM projeksiyon sınıflandırmanın hata matrisi



Şekil 81. PointNet - LSTM modelinin eğitim grafikleri



Şekil 82. PointNet - LSTM sınıflandırmanın hata matrisi

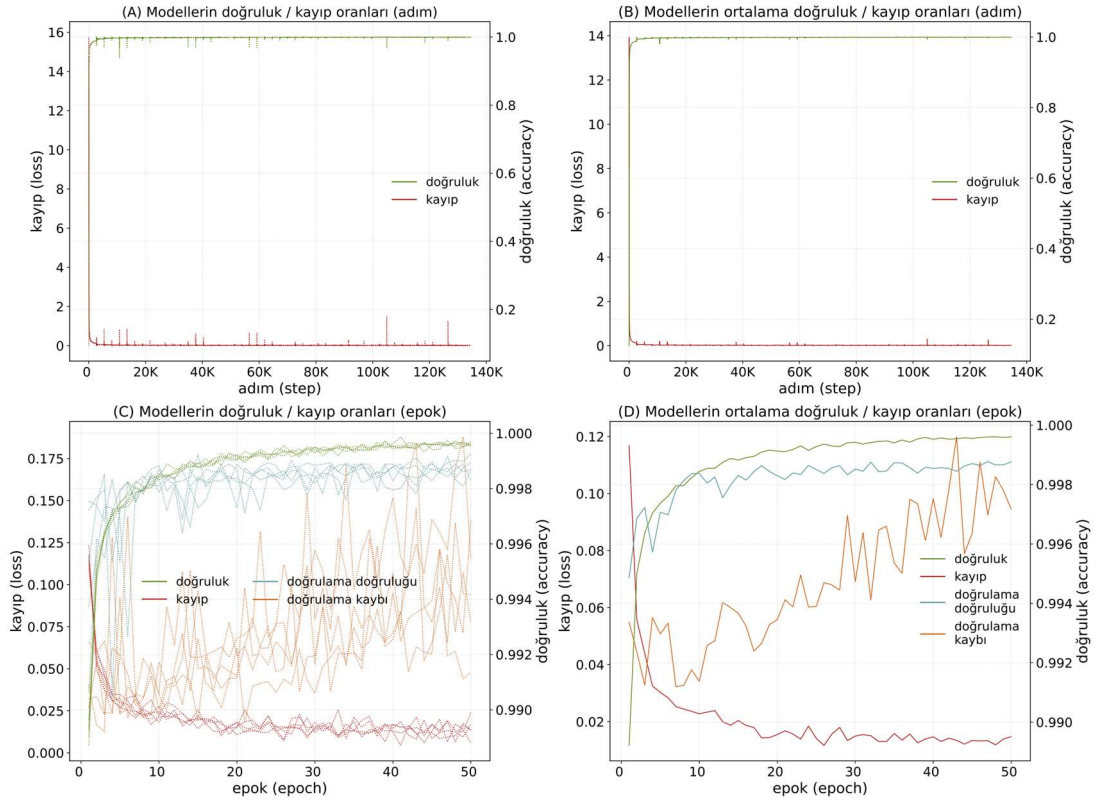


Şekil 83. PointNet noktalar kümesi toplamı modelinin eğitim grafikleri

1	55 (0.92)	0 (0.01)	1 (0.02)	0 (0.01)	0 (0.00)	0 (0.00)	2 (0.04)	0 (0.00)
2	4 (0.07)	52 (0.87)	0 (0.00)	0 (0.00)	0 (0.00)	3 (0.05)	0 (0.00)	0 (0.01)
3	0 (0.00)	0 (0.01)	52 (0.88)	5 (0.10)	0 (0.01)	0 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	7 (0.12)	52 (0.87)	0 (0.00)	0 (0.01)	0 (0.00)	0 (0.00)
5	0 (0.01)	0 (0.00)	0 (0.01)	1 (0.02)	48 (0.80)	9 (0.16)	0 (0.00)	0 (0.01)
6	0 (0.00)	0 (0.00)	1 (0.02)	1 (0.02)	5 (0.08)	52 (0.88)	0 (0.00)	0 (0.00)
7	3 (0.06)	0 (0.00)	3 (0.06)	0 (0.00)	1 (0.02)	0 (0.00)	50 (0.83)	1 (0.02)
8	0 (0.01)	1 (0.03)	0 (0.00)	0 (0.01)	0 (0.00)	1 (0.02)	0 (0.00)	56 (0.93)
	1	2	3	4	5	6	7	8

Tahmin

Şekil 84. PointNet noktalar kümesi toplamı sınıflandırmasının hata matrisi



Şekil 85. Öznitelik füzyonu modelinin eğitim grafikleri

1	2398 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	7 (0.00)	2387 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	1 (0.00)	0 (0.00)	1 (0.00)
3	0 (0.00)	0 (0.00)	2386 (0.99)	10 (0.00)	1 (0.00)	1 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	3 (0.00)	2306 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
5	0 (0.00)	0 (0.00)	0 (0.00)	2 (0.00)	2302 (0.99)	10 (0.00)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	7 (0.00)	2329 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	1 (0.00)	2 (0.00)	10 (0.00)	2 (0.00)	2377 (0.99)	4 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	4 (0.00)	0 (0.00)	1 (0.00)	2392 (1.00)
	1	2	3	4	5	6	7	8

Tahmin

Şekil 86. Öznitelik füzyonu sınıflandırmasının hata matrisi

Tablo 4. Kamera veri seti modellerinin doğruluk oranları

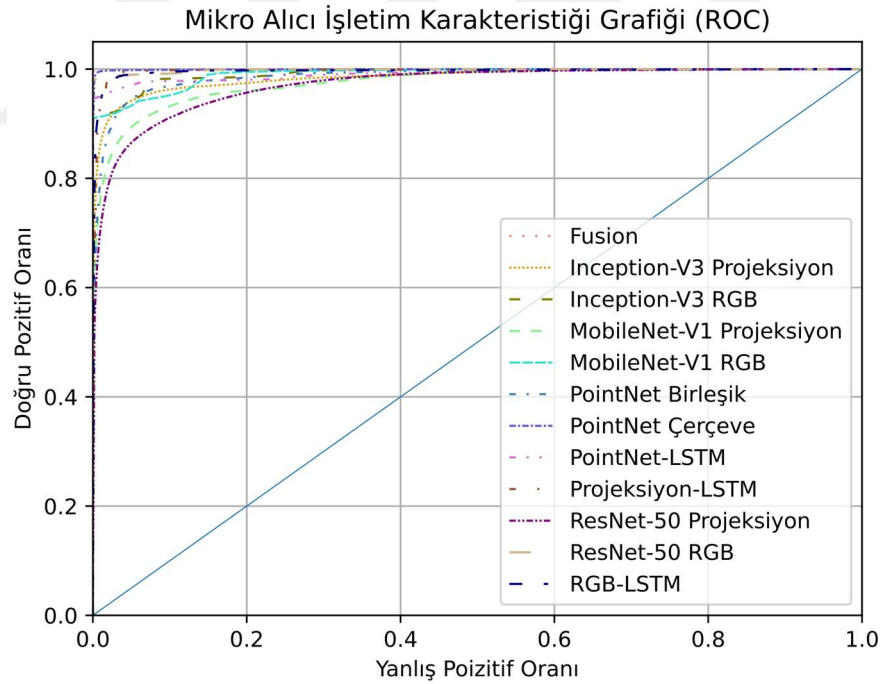
Model	Epok Miktarı	Doğruluk Oranı (%)
Inception-V3 RGB	10	99,87
MobileNet-V1 RGB	10	99,75
ResNet-50 RGB	10	99,62
Inception-V3 Projeksiyon	85	89,37
MobileNet-V1 Projeksiyon	85	83,75
ResNet-50 Projeksiyon	85	81,37
PointNet Çerçeve	10	98,50
PointNet Birleştirilmiş	25	87,25
VGG-16-LSTM RGB	50	93,87
VGG-16-LSTM Projeksiyon	50	97,87
PointNet-LSTM	50	100,00
Öznitelik Füzyonu	50	99,62

Tablo 5. Kamera veri seti modellerinin değerlendirme metrikleri

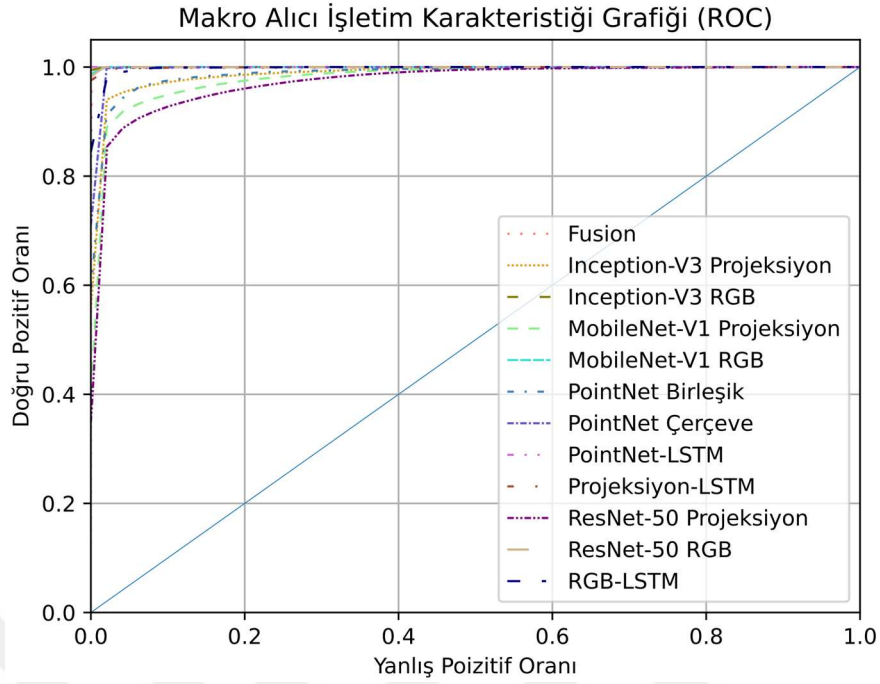
Model	Duyarlılık (Sensitivity)	Özgüllük (Specificity)	F1 Mikro	F1 Makro	ROC AUC
Inception-V3 RGB	0,99808	0,99975	0,99810	0,99808	0,99998
MobileNet-V1 RGB	0,99604	0,99943	0,99607	0,99604	0,99998
ResNet-50 RGB	0,99623	0,99946	0,99624	0,99621	0,99998
Inception-V3 Projeksiyon	0,90312	0,98493	0,89338	0,89351	0,99118
MobileNet-V1 Projeksiyon	0,85620	0,97722	0,83834	0,83829	0,98403
ResNet-50 Projeksiyon	0,84028	0,97381	0,81379	0,81508	0,97673
PointNet Çerçeve	0,98617	0,99802	0,98617	0,98609	0,99969
PointNet Birleşik	0,87751	0,98181	0,87249	0,87332	0,98884
VGG-16-LSTM RGB	0,94282	0,99135	0,93891	0,93878	0,99833
VGG-16-LSTM Projeksiyon	0,97917	0,99701	0,97903	0,97891	0,99955
PointNet-LSTM	1,00000	1,00000	1,00000	1,00000	1,00000
Öznitelik Füzyonu	0,99559	0,99937	0,99563	0,99561	0,99876

Tablo 4’te, kamera veri seti sınıflandırma modellerinin doğruluk oranları (accuracy) ve eğitim epok miktarları gösterilmiştir. Bu tabloya göre en başarılı model PointNet-LSTM modeli, en az başarılı model ise ResNet-50 Projeksiyon modeli olmuştur. Tablo 5’te, kamera veri seti sınıflandırma modellerinin Scikit-Learn kütüphanesi ile hesaplanan duyarlılık (sensitivity), özgüllük (specificity), F1-Makro, F1-Mikro ve alıcı işletim karakteristiği grafiği (Receiver Operating Characteristic-Area Under Curve, ROC-AUC) değerleri görülmektedir. Bu değerler her bir ağ için sekiz sınıfa ait değerlerin ortalamasının, beş kat doğrulama modellerinde elde edilen değerlerinin ortalamasıdır.

Şekil 87 ve 88’de, kamera veri setinde kullanılan modellerin mikro ve makro ROC eğrileri çizdirilmiştir. Eğriler, Scikit-Learn kütüphanesinde hesaplandıktan sonra, farklı sayıda veri noktalarına sahip oldukları için, eğrilerin aynı grafikte çizdirilebilmeleri adına, değerler x eksenindeki sabit miktarda (50 adet) noktaya göre enterpolasyonla elde edilmiştir. PointNet-LSTM modelinin makro ROC eğrisi, x eksenini doğrultusunda y ekseninde 1,0 değerine en çabuk ulaşan model olarak görülmektedir.

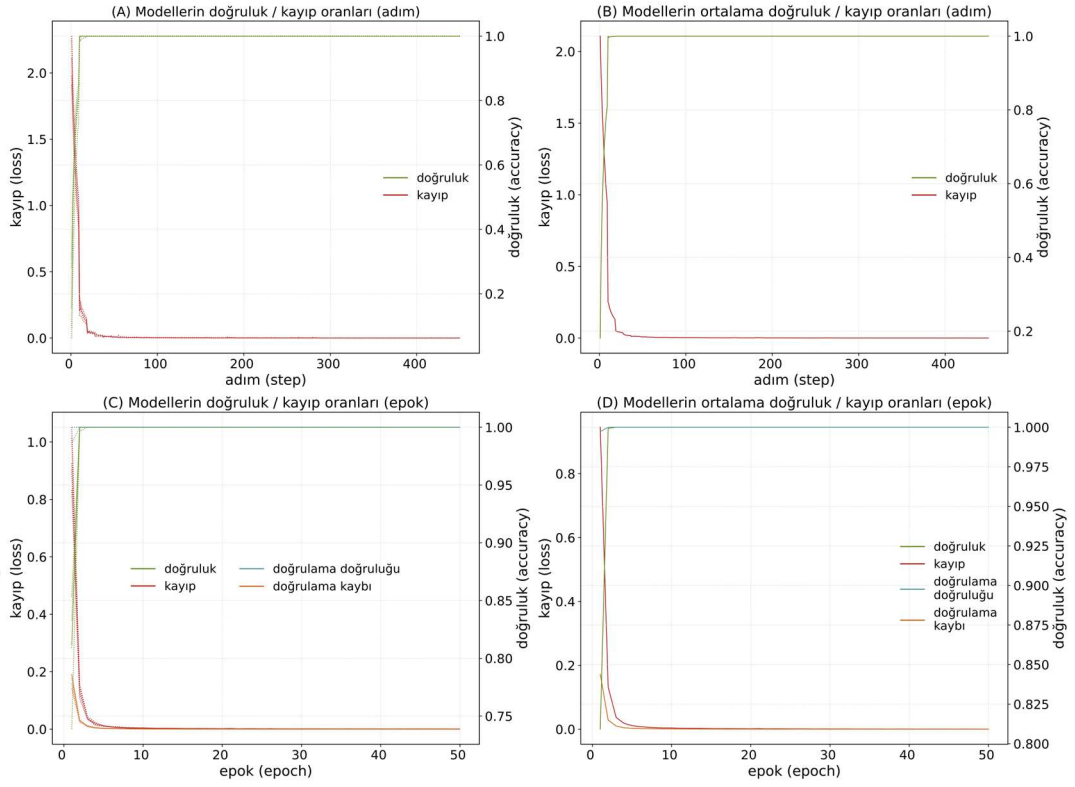


Şekil 87. Kamera veri seti modelleri için Mikro-ROC eğrisi grafiği

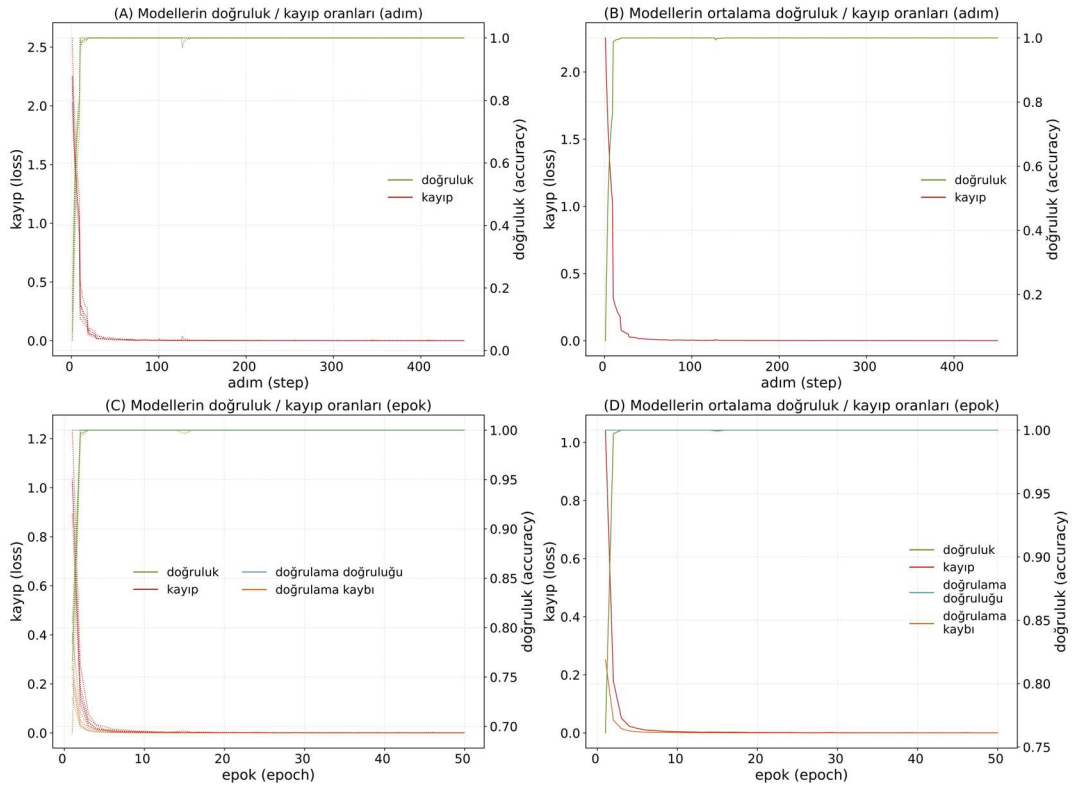


Şekil 88. Kamera veri seti modelleri için Makro-ROC eğrisi grafiği

Sınıflandırma başarısı %100 olan PointNet-LSTM ağı için, yeniden bir değerlendirme daha gerçekleştirilmiştir. Bu ağda, birinci lavabodaki birinci kişinin verileriyle eğitilen bir ağ, ikinci lavabodaki ikinci kişinin verileriyle test edilmiştir. Aynı şekilde, ikinci lavabodaki ikinci kişinin verileriyle eğitilmiş bir ağ, birinci lavabodaki birinci kişinin verileriyle test edilmiştir. Yine, her bir oturumda bulunan noktalar kümesi verilerinden 120 çerçevelik bir miktarın LSTM ağı için uygun hale getirilmesiyle veriler elde edilmiştir. Bu iki ayırım için belirlenen modeller, beş kat çapraz doğrulama yöntemiyle eğitilmiştir. Her iki model için de 45 oturum eğitim, 15 oturum doğrulama ve 15 oturum test verisi kullanılmıştır. Sınıflandırma sonucunda, yine tüm test örnekleri %100 doğrulukla sınıflandırılmıştır. Ağlara ait eğitim grafikleri Şekil 89-90'da ve hata matrisi Şekil 91'de verilmiştir.



Şekil 89. PointNet-LSTM yeniden değerlendirme modellerinden birinci modelin eğitim grafikleri



Şekil 90. PointNet-LSTM yeniden değerlendirme modellerinden ikinci modelin eğitim grafikleri

1	15 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	0 (0.00)	15 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	15 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	0 (0.00)	15 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
5	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	15 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	15 (1.00)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	15 (1.00)	0 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	15 (1.00)
	1	2	3	4	5	6	7	8

Tahmin

Şekil 91. PointNet-LSTM ikinci değerlendirme sınıflandırmasının hata matrisi

3.2. Sentetik Veri Seti ile Sınıflandırma Sonuçları

Kamera veri setiyle eğitilmiş sekiz sınıflı modellerden Inception-V3 RGB ağı, beş kat çapraz doğrulama modeliyle beraber elde edilen sentetik veri setinde kırpılmış görüntüler üzerinde sınıflandırma için kullanılmıştır. Buradaki amaç, sentetik görüntülerin gerçek görüntülere ne kadar benzer olduğuna dair bir fikir elde etmektir. 256x256x3 boyutundaki sentetik görüntüler, ağlara uyumlu olması için 96x96x3 boyutuna getirilmiştir. Inception-V3 RGB ağının beş kat çapraz doğrulama modeli için elde edilen ortalama hata matrisi Şekil 92’de verilmiştir. Doğruluk oranı %38 olmuştur. 2., 3. ve 8. hareketlerde doğruluk oranı düşük olurken, 1., 5. ve 6. hareketlerde yüksek olmuştur.

1	1471 (0.74)	1 (0.00)	0 (0.00)	47 (0.02)	48 (0.02)	124 (0.06)	279 (0.14)	12 (0.01)
2	1125 (0.57)	166 (0.08)	7 (0.00)	38 (0.02)	179 (0.09)	208 (0.10)	246 (0.12)	12 (0.01)
3	350 (0.18)	0 (0.00)	84 (0.04)	235 (0.12)	380 (0.19)	811 (0.41)	101 (0.05)	20 (0.01)
4	211 (0.11)	3 (0.00)	1 (0.00)	740 (0.37)	510 (0.26)	446 (0.22)	50 (0.03)	21 (0.01)
5	129 (0.07)	5 (0.00)	1 (0.00)	139 (0.07)	1379 (0.70)	243 (0.12)	85 (0.04)	0 (0.00)
6	49 (0.03)	1 (0.00)	11 (0.01)	111 (0.06)	111 (0.06)	1662 (0.84)	29 (0.01)	7 (0.00)
7	138 (0.07)	0 (0.00)	231 (0.12)	157 (0.08)	688 (0.35)	292 (0.15)	457 (0.23)	19 (0.01)
8	67 (0.03)	16 (0.01)	11 (0.01)	595 (0.30)	172 (0.09)	908 (0.46)	109 (0.06)	103 (0.05)
	1	2	3	4	5	6	7	8
	Tahmin							

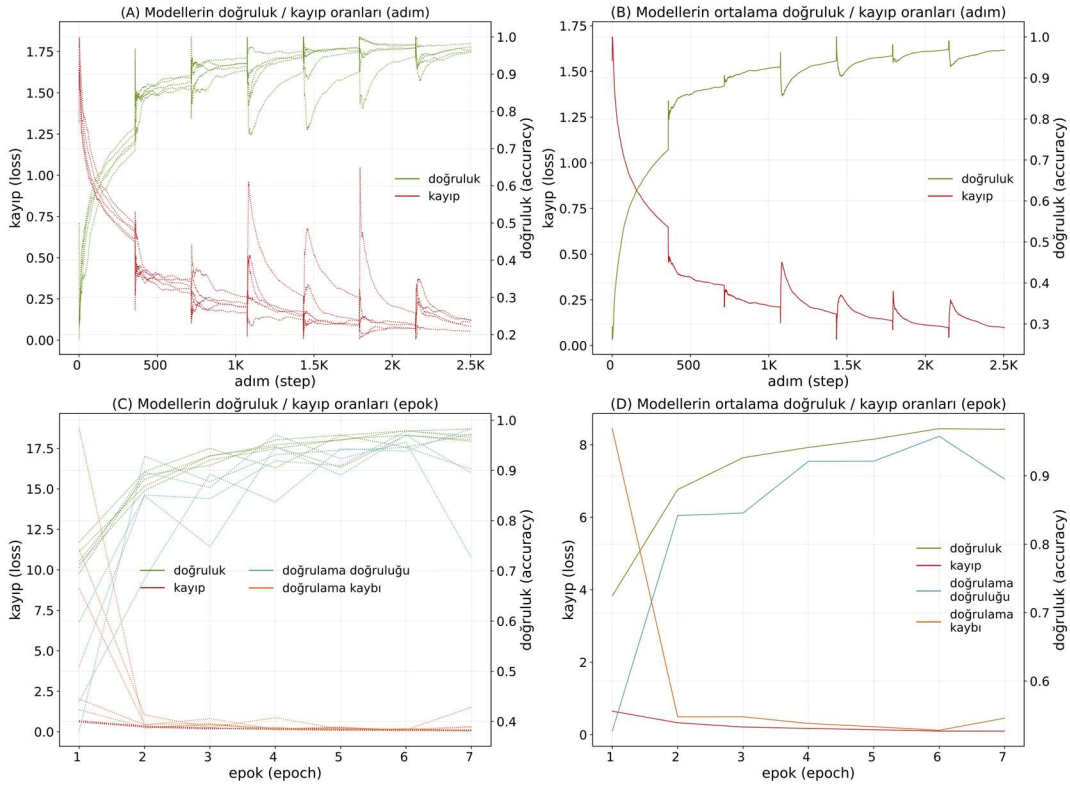
Şekil 92. Kamera veri setiyle eğitilmiş Inception-V3 ağının sentetik veri seti sınıflandırmasının hata matrisi

Sentetik veri setiyle Inception-V3 ve YOLO-8n ağları, 4, 8 ve 32 sınıflı modeller şeklinde toplamda altı model olarak eğitilmiştir. 4 sınıflı modeller, ellerin sabun miktarını sınıflandırmak, 8 sınıflı modeller hareket sınıflandırmak, 32 sınıflı modeller ise hem hareket hem sabun miktarı sınıflandırmak için hazırlanmıştır. Sentetik veriler üzerinde, her seferinde farklı eğitim ve doğrulama verileri ile, beş kat çapraz doğrulama sağlayacak şekilde her bir sınıf ayrımı için beş farklı model eğitilmiştir. Sekiz hareketin sınıflandırması, veri setinin halihazırda hareketlere göre etiketlenmiş olması sebebiyle doğrudan yapılmıştır. Tüm kamera veri seti görüntüleri birer test verisiymiş gibi Inception-V3 ve YOLO-8n ağlarının 8 sınıflı modellerinden geçirilmiş, her birine ait beş modelin hata matrisleri elde edilmiştir ve ortalaması alınmıştır.

4 ve 32’li sınıf modelleri, veri setinin bu şekilde etiketli veriler içermemesi dolayısıyla, tüm kamera veri seti üzerinde sınıflandırıcı olarak kullanılmış, elde edilen sonuçlar, veri setinin tüm çerçeveleri üzerine Python diliyle OpenCV kütüphanesi kullanılarak yazılmıştır. Böylece, modellerin verdiği tüm sonuçlar bir bakışta görüntülenebilecek şekilde birleşik görüntüler oluşturulmuştur.

3.2.1. Inception-V3 ile Sabun Miktarı Sınıflandırma

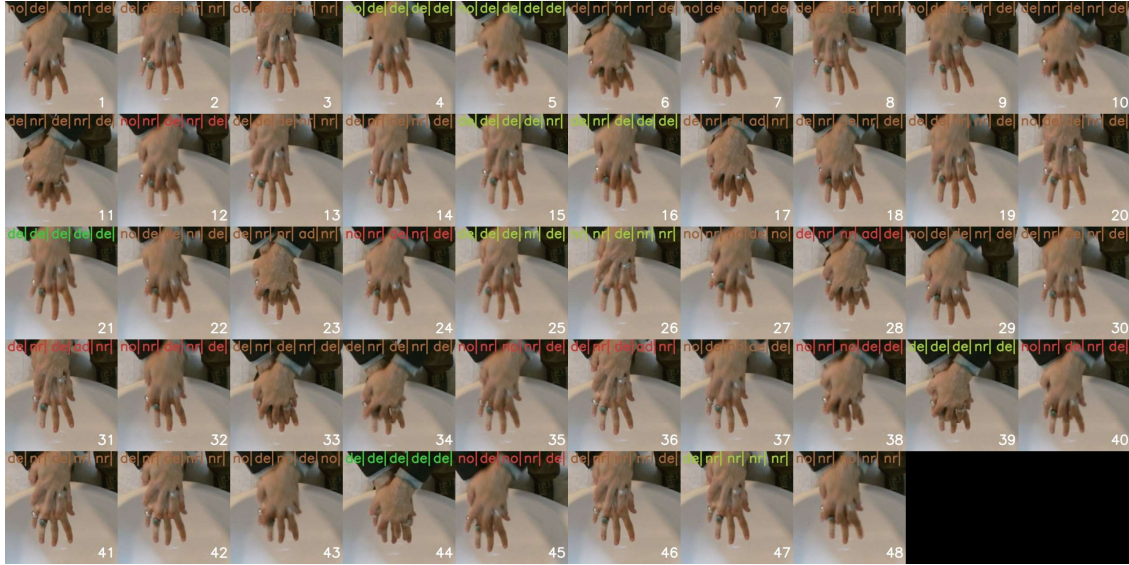
4 sınıflı Inception-V3 modeli sınıflandırmasında, kamera görüntüleri veri seti üzerinde RGB görüntülerde her bir çerçeve için beş kat çapraz doğrulama modellerinin verdiği sonuç görüntülenebilecek şekilde birleşik görüntüler oluşturulmuştur. Birleşik görüntüler, bir harekete ait bir oturumdaki yıkama görüntülerinin tamamıdır. Bu görüntülerin kare oranı değişken olduğu için, elde edilen görüntüler de dinamik bir biçimde oluşturulmakta, görüntülerin bittiği noktada kalan boşluk siyah olmaktadır. Bu görüntülerin bir örneği Şekil 95'te görülmektedir. Test sırasında modellerin çıktıkları, her bir çerçeve üzerine no (no soap) sabunsuz, de (deficient soap) az sabunlu, nr (normal soap) normal sabunlu, ad (adequate soap) çok sabunlu olmak üzere yazılmıştır. Modellerin çıktılarının benzerliğine göre yazıların rengi kırmızıdan yeşile doğru (farklıdan aynıya doğru) seçilmiştir. Modellerin sentetik veri seti üzerinde eğitim grafikleri Şekil 93'te görülmektedir. Bu grafikler kamera veri setinde açıklandığı üzere adım ve epok bazlı çizdirilmiştir. Şekil 94'te ise, beş kat çapraz doğrulama modelinin sentetik veri setindeki test verilerinde elde ettiği ortalama hata matrisi görülmektedir. Sentetik veriler üzerinde doğruluk oranı %89,5 olmuştur.



Şekil 93. Inception-V3 4 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri

Hareket	sabunsuz	371 (0.94)	23 (0.06)	0 (0.00)	0 (0.00)
	az sabun	21 (0.05)	341 (0.86)	18 (0.05)	15 (0.04)
	normal sabun	1 (0.00)	7 (0.02)	321 (0.81)	65 (0.17)
	çok sabun	0 (0.00)	1 (0.00)	11 (0.03)	383 (0.97)
		sabunsuz	az sabun	normal sabun	çok sabun
		Tahmin			

Şekil 94. Inception-V3 4 sınıflı modelin sentetik veri seti üzerinde hata matrisi



Şekil 95. Inception-V3 ağı 4 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)

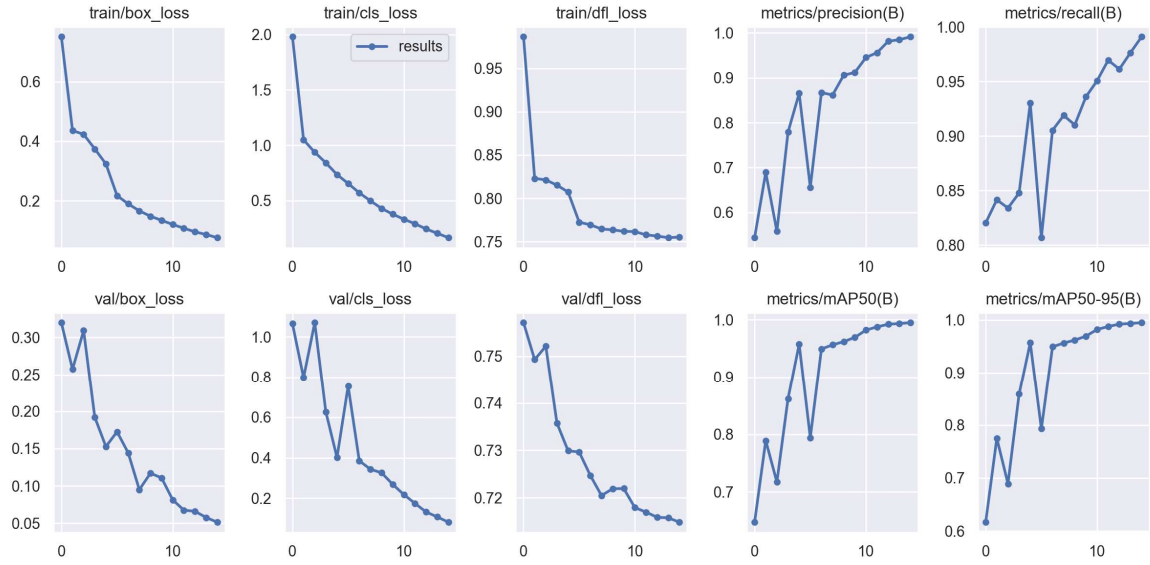
3.2.2. YOLO-8n ile Sabun Miktarı Sınıflandırma

YOLO-8n modeliyle, Inception-V3 modeli gibi kamera veri seti üzerindeki RGB görüntüler üzerinde sınıflandırma yapılmış, farklı olarak kırılmış görüntüler yerine kırılmamış ham görüntüler kullanılmıştır. Bir hareketin bir yıkama oturumunda bulunan tüm görüntülerden, beş kat çapraz doğrulama modellerinin el bölgesi olarak tespit ettiği

bölgeler ve tespit ettikleri sınıflar, çerçeveler yan yana olmak üzere birleşik bir görüntüde birleştirilmiştir. El bölgesi tespiti yapamayan modelin çıktı bölgesi boşluk (siyah) olmuştur. Modellerin sınıf çıktısı, Inception-V3 modelindeki renklendirme metodu uygulanarak çerçeveler üzerine yazılmıştır. Şekil 99'da elde edilen birleşik görüntülerden bir örnek verilmiştir. Bu görüntülerde, beş kat çapraz doğrulamadaki her modelin tespit ettiği el bölgesi ve sınıfı her bir görüntü karesi için yan yana getirilmekte ve birleştirilmektedir. Şekil 96'da, beş modelden birine ait eğitim grafikleri görülmektedir. Parametrelerin ifade ettiği kavramlar aşağıdaki gibidir;

- `box_loss`: Obje tespiti kutularının koordinatlarının hata miktarını belirtir.
- `cls_loss`: Tespit edilen objelerin sınıflarının hata miktarını belirtir.
- `dfl_loss`: "Distribution Focal Loss" obje tespiti kutularının iyileştirilmesi için kullanılan kayıp fonksiyonudur.
- `metrics/precision`: Bir sınıfa dair yapılan tüm tespitlerin doğruluk oranını belirtir.
- `metrics/recall`: Bir sınıfa ait doğru tespit edilen objelerin, verideki tüm doğru objelere oranını belirtir.
- `metrics/map50`: Intersection over Union (IoU) hesaplamasının, 0,50 eşik değeriyle yapıldığında elde edilen doğruluk oranının ortalamasını belirtir.
- `metrics/map50-95`: Intersection over Union (IoU) hesaplamasının, 0,50 - 0,95 eşik değerleri arasında yapıldığında elde edilen doğruluk oranının ortalamasını belirtir.

Şekil 96'da x ekseni epok miktarını, y ekseni ise grafiklerin üzerinde bulunan parametre değerini belirtmektedir. Şekil 97'de, beş kat çapraz doğrulama modeli için sentetik veri seti üzerindeki test verilerinde yapılan sınıflandırmanın ortalama hata matrisi görülmektedir. Modellerin doğruluk oranı 95,75 olmuştur. Şekil 98'de, etiketli eğitim verilerinin eğitimden önce hazırlandığı bir görüntü yer almaktadır. Bu görüntüde daha önceden etiketlenmiş el bölgeleri ve sınıfları görülmektedir. YOLO ağı bu örnek görüntüleri eğitimden önce otomatik oluşturmaktadır.



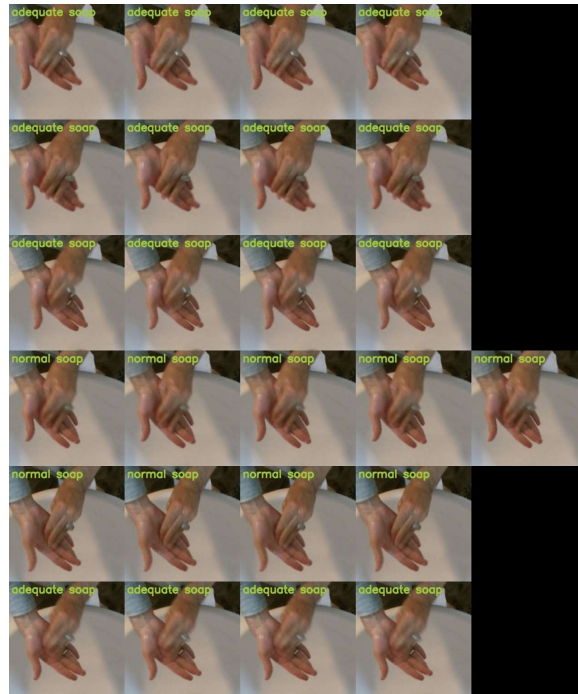
Şekil 96. YOLO-8n 4 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri

Hareket	sabunsuz	395 (0.99)	3 (0.01)	0 (0.00)	0 (0.00)	2 (0.01)
	az sabun	0 (0.00)	385 (0.95)	5 (0.01)	0 (0.00)	13 (0.03)
	normal sabun	0 (0.00)	7 (0.02)	384 (0.93)	3 (0.01)	16 (0.04)
	çok sabun	0 (0.00)	0 (0.00)	6 (0.02)	392 (0.96)	8 (0.02)
	arka plan	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
Tahmin		sabunsuz	az sabun	normal sabun	çok sabun	arka plan

Şekil 97. YOLO-8n 4 sınıflı modelin sentetik veri seti üzerindeki hata matrisi



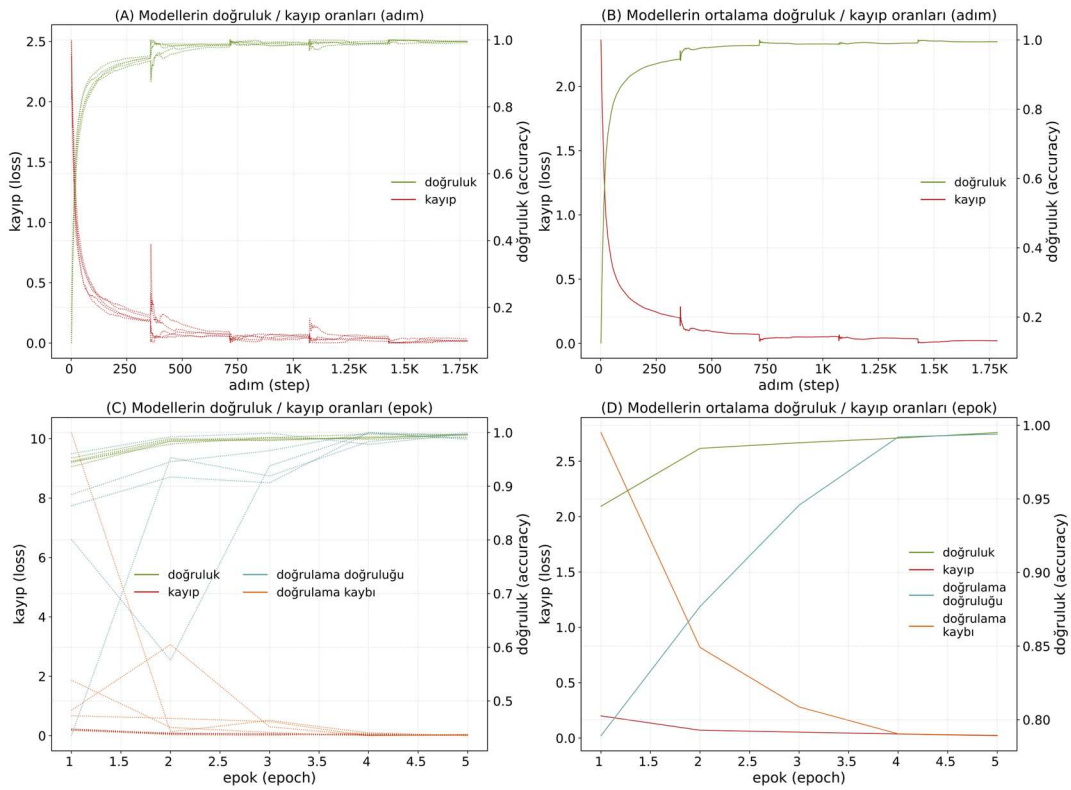
Şekil 98. YOLO-8n 4 sınıflı modelin eğitim verileri ve etiketlerinden örnekler



Şekil 99. YOLO-8n ağının 4 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)

3.2.3. Inception-V3 ile Hareket Sınıflandırma

8 Sınıflı modellerde, kamera veri setinin 8 sınıflı olması sebebiyle, doğrudan sınıflandırma sonuçları elde edebilmiştir. Kamera veri setindeki tüm görüntüler, birer test verisi gibi değerlendirilmiş ve sentetik veri setiyle eğitilmiş Inception-V3 modellerine verilmiştir. Beş kat çapraz doğrulama modellerinin kamera veri setinde elde ettiği sonuçlara ait ortalama hata matrisi Şekil 102'deki gibi olmuştur. Burada elde edilen doğruluk oranı %47,5 olmuştur. Beş kat çapraz doğrulama modellerinin sentetik veri seti üzerindeki eğitim grafikleri Şekil 100'de, sentetik verilerdeki test verilerinden elde edilen ortalama hata matrisi Şekil 101'de verilmiştir. Sentetik veri setindeki sınıflandırmada doğruluk oranı %99,75 olmuştur. Kamera veri seti görüntülerinde, doğruluk oranı en yüksek olarak tespit edilen hareket 1. hareket, en düşük olarak sınıflandırılan hareket ise 8. hareket olmuştur.



Şekil 100. Inception-V3 8 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri

1	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
2	0 (0.00)	194 (0.99)	0 (0.00)	0 (0.00)	2 (0.01)	0 (0.00)	0 (0.00)	0 (0.00)
3	0 (0.00)	0 (0.00)	197 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
4	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
5	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	197 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)
6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (0.99)	0 (0.00)	0 (0.00)
7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)
8	0 (0.00)	0 (0.00)	0 (0.00)	2 (0.01)	0 (0.00)	0 (0.00)	0 (0.00)	194 (0.99)
	1	2	3	4	5	6	7	8

Tahmin

Şekil 101. Inception-V3 8 sınıflı modelin sentetik veri seti üzerindeki hata matrisi

1	16682 (0.81)	213 (0.01)	1267 (0.06)	435 (0.02)	775 (0.04)	626 (0.03)	553 (0.03)	82 (0.00)
2	2071 (0.10)	12863 (0.65)	142 (0.01)	2010 (0.10)	1820 (0.09)	354 (0.02)	430 (0.02)	139 (0.01)
3	353 (0.02)	167 (0.01)	9402 (0.49)	3559 (0.18)	826 (0.04)	898 (0.05)	2334 (0.12)	1823 (0.09)
4	15 (0.00)	106 (0.01)	3978 (0.21)	9710 (0.51)	250 (0.01)	1748 (0.09)	826 (0.04)	2452 (0.13)
5	591 (0.03)	3892 (0.20)	1326 (0.07)	2688 (0.14)	6575 (0.34)	425 (0.02)	1314 (0.07)	2309 (0.12)
6	1157 (0.06)	503 (0.03)	1916 (0.10)	1849 (0.10)	478 (0.03)	11430 (0.60)	492 (0.03)	1277 (0.07)
7	3531 (0.19)	5080 (0.27)	804 (0.04)	1048 (0.06)	1280 (0.07)	1278 (0.07)	5093 (0.27)	912 (0.05)
8	1234 (0.06)	5548 (0.29)	675 (0.04)	1329 (0.07)	2910 (0.15)	3884 (0.20)	1089 (0.06)	2388 (0.13)
	1	2	3	4	5	6	7	8

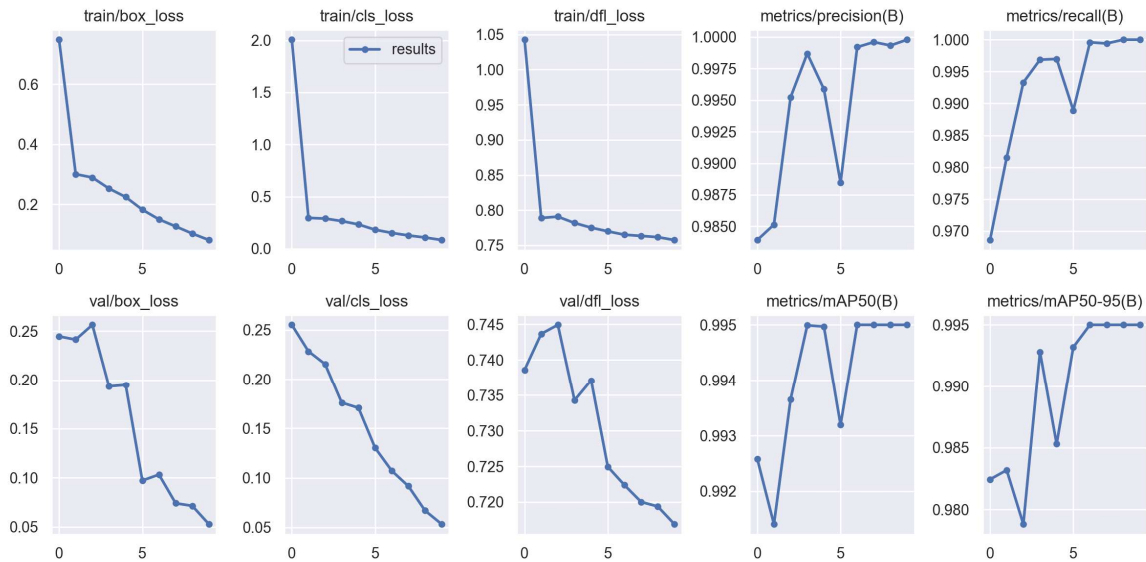
Tahmin

Şekil 102. Inception-V3 8 sınıflı modelin kamera veri seti üzerindeki hata matrisi

3.2.4. YOLO-8n ile Hareket Sınıflandırma

YOLO-8n modelinin 8 sınıflı değerlendirmesinde, kamera veri setindeki kırılmamış RGB görüntüleri üzerinde, beş kat çapraz doğrulama modellerinin tamamının el bölgesi tespiti yaptığı durumdaki görüntüler sınıflandırma için kullanılmıştır. Ayrıca, daha önceki modellerde elde edilen birleşik görüntüler bu model için de oluşturulmuştur. Birleşik görüntülerde tüm hareketlere ait her bir oturum içerisindeki görüntüler üzerinde beş kat çapraz doğrulama modellerinin verdiği sonuçlar ilgili çerçeve üzerine yazdırılmıştır. Şekil 106'da sınıflandırma birleşik görüntülerinden bir örnek verilmiştir.

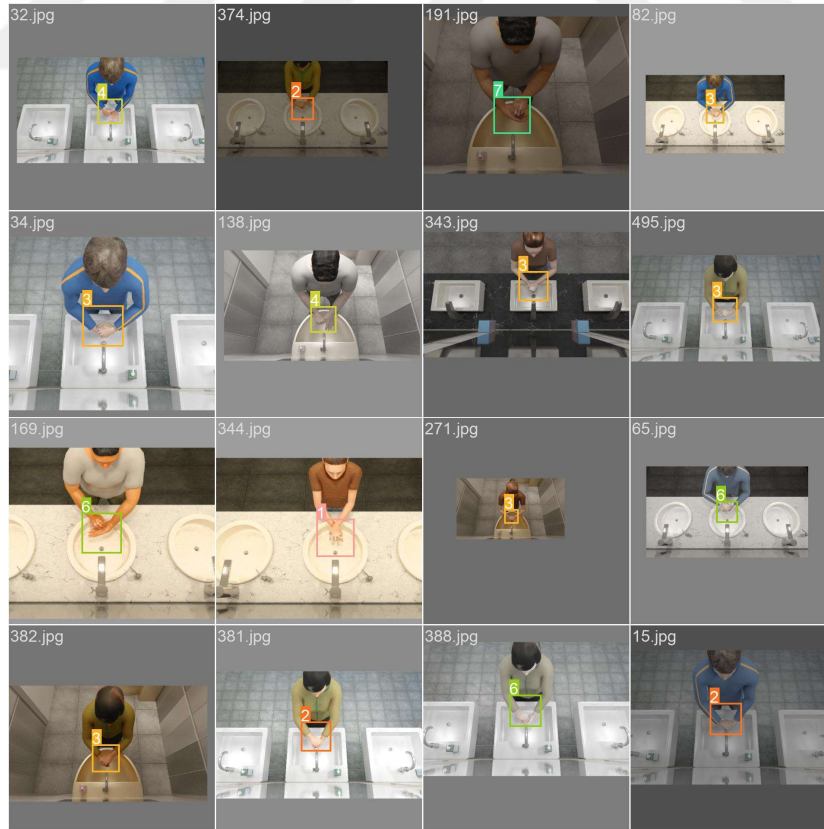
Şekil 103'te, beş modelden birine ait eğitim grafikleri görülmektedir. Grafikte x eksenini epok miktarını, y eksenini ise grafiklerin üzerinde bulunan parametre değerini belirtmektedir. Şekil 104'te, sentetik veri seti üzerindeki test verilerinde beş kat çapraz doğrulama modellerinin ortalama hata matrisi görülmektedir. Burada doğruluk oranı %100 olmuştur. Şekil 105'te, etiketli eğitim verilerinin eğitimden önce YOLO ağı tarafından otomatik hazırlandığı bir görüntü yer almaktadır. Görüntüde önceden etiketlenmiş el bölgeleri ve sınıfları görülmektedir. YOLO-8n 8 sınıflı modelin kamera verileri üzerinde elde ettiği doğruluk oranı %56,5 olmuştur. Kamera veri setindeki sınıflandırmaya ait ortalama hata matrisi Şekil 107'de verilmiştir. Doğruluk oranı en yüksek olarak tespit edilen hareket 1. hareket, en düşük olarak sınıflandırılan hareket ise 8. hareket olmuştur.



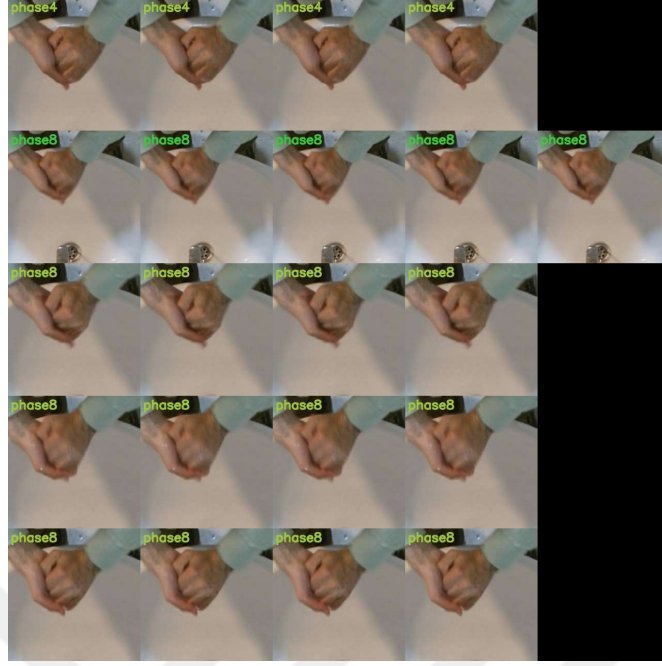
Şekil 103. YOLO-8n 8 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikler,

Hareket	1	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
	2	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
	3	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
	4	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
	5	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)	0 (0.00)
	6	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)	0 (0.00)
	7	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)	0 (0.00)
	8	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	196 (1.00)
arka plan		0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
		1	2	3	4	5	6	7	8
		arka plan							
		Tahmin							

Şekil 104. YOLO-8n 8 sınıflı modelin sentetik veri seti üzerindeki hata matrisi



Şekil 105. YOLO-8n 8 sınıflı modelin eğitim verileri ve etiketlerinden örnekler



Şekil 106. YOLO-8n ağıının 8 sınıflı modelinin kamera veri setindeki sınıflandırmasından örnekler (birleşik görüntü)

1	10078 (0.79)	48 (0.00)	51 (0.00)	474 (0.04)	243 (0.02)	1646 (0.13)	1 (0.00)	148 (0.01)
2	9 (0.00)	8744 (0.87)	65 (0.01)	762 (0.08)	84 (0.01)	294 (0.03)	69 (0.01)	4 (0.00)
3	25 (0.00)	56 (0.01)	4425 (0.46)	519 (0.05)	348 (0.04)	102 (0.01)	4153 (0.43)	4 (0.00)
4	6 (0.00)	14 (0.00)	300 (0.03)	6352 (0.69)	41 (0.00)	274 (0.03)	7 (0.00)	2251 (0.24)
5	90 (0.01)	80 (0.01)	258 (0.02)	1814 (0.16)	8575 (0.75)	31 (0.00)	542 (0.05)	19 (0.00)
6	24 (0.00)	219 (0.03)	293 (0.03)	1248 (0.15)	162 (0.02)	6232 (0.74)	43 (0.01)	237 (0.03)
7	1259 (0.10)	167 (0.01)	393 (0.03)	651 (0.05)	7933 (0.64)	123 (0.01)	1768 (0.14)	136 (0.01)
8	7 (0.00)	569 (0.11)	49 (0.01)	2671 (0.50)	739 (0.14)	678 (0.13)	185 (0.03)	435 (0.08)
	1	2	3	4	5	6	7	8

Tahmin

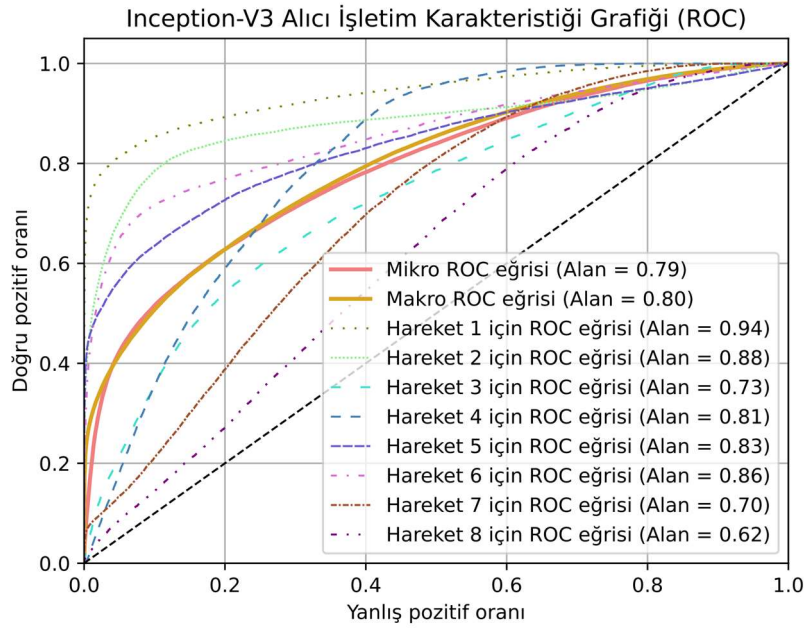
Şekil 107. YOLO-8n 8 sınıflı modelin kamera veri seti üzerindeki hata matrisi

3.2.5. Hareket Sınıflandırma Modellerinin Değerlendirme Metrikleri

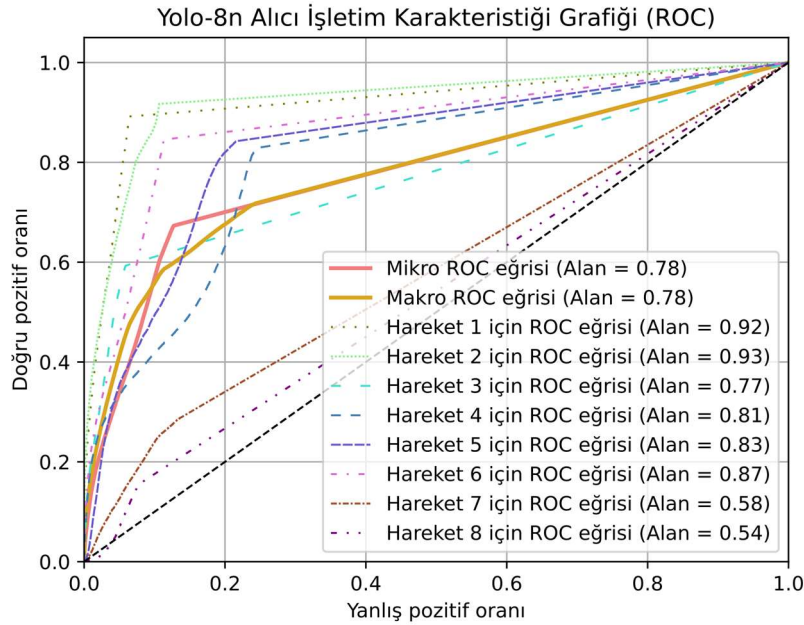
Şekil 108’de Inception-V3 8 sınıflı modelin alıcı işletim karakteristiği grafiği (ROC) ve alanları verilmiştir. Şekil 109’da ise Yolo-8n 8 sınıflı modelin alıcı işletim karakteristiği grafiği ve alanları verilmiştir. Grafiklerde, doğruluk oranı yüksek sınıfların ROC eğrileri ideale daha çabuk yakınsamış ve alanları daha yüksek çıkmıştır. Tablo 6’da, sentetik veri setindeki 8 sınıflı modellerin değerlendirme metrikleri verilmiştir.

Tablo 6. Sentetik veri seti 8 sınıflı modellerin değerlendirme metrikleri

Model	Duyarlılık (Sensitivity)	Özgüllük (Specificity)	F1		ROC AUC
			Mikro	Makro	
Inception-V3	0,45463	0,92632	0,47746	0,45594	0,79601
Yolo-8n	0,51765	0,93610	0,54158	0,51046	0,78047



Şekil 108. Inception-V3 sentetik veri seti hareket sınıflandırması alıcı işletim karakteristiği grafiği

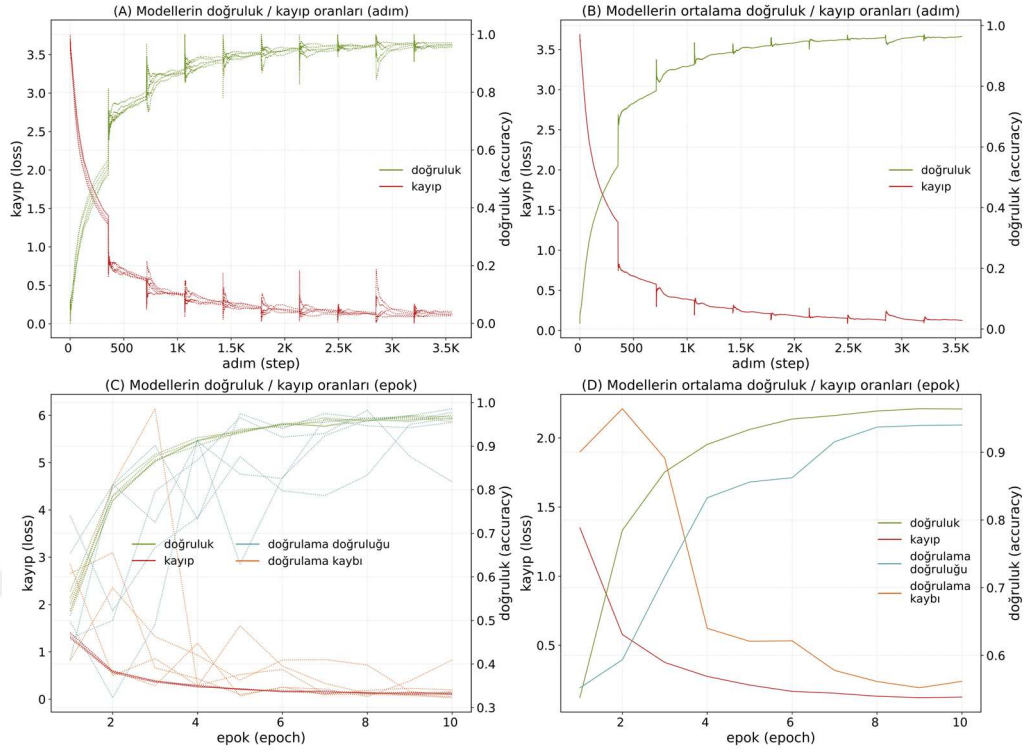


Şekil 109. Yolo-8n sentetik veri seti hareket sınıflandırması alıcı işletim karakteristiği grafiği

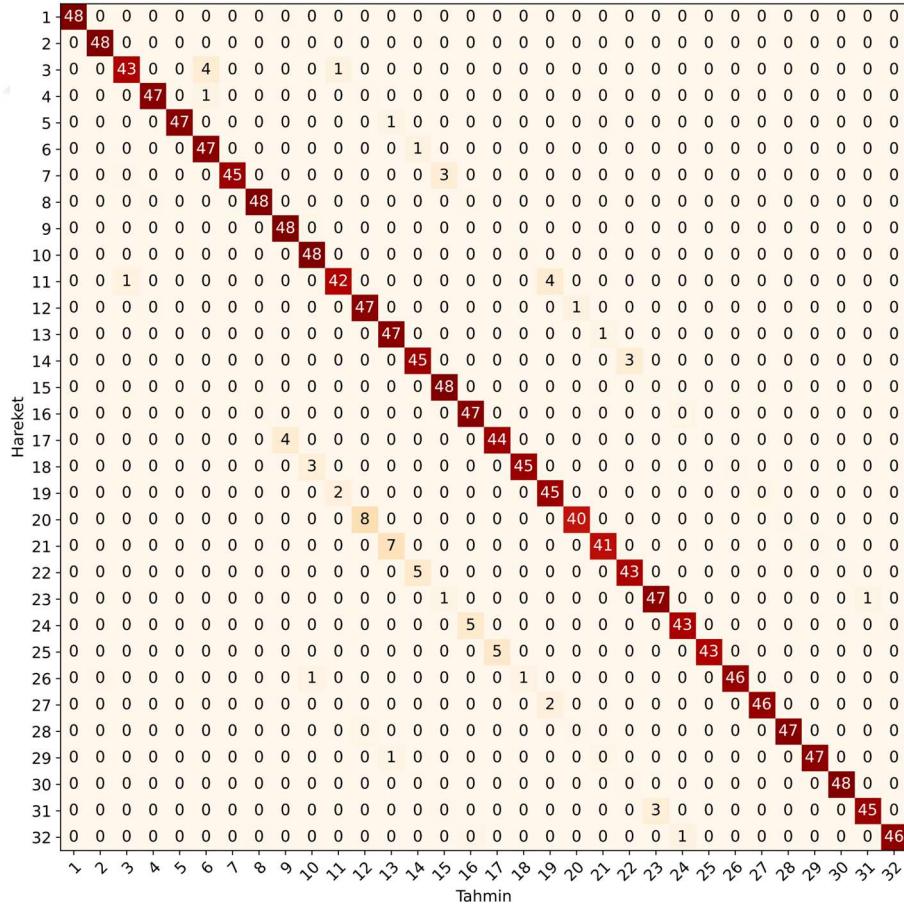
3.2.6. Inception-V3 ile Sabun Miktarı ve Hareket Sınıflandırma

32 sınıflı Inception-V3 modeli için, tıpkı 4 sınıflı modelde olduğu gibi, kamera veri setindeki RGB her bir çerçeve için, beş kat çapraz doğrulama modelinin verdiği sonuçlar aynı çerçeve üzerine yazılıp, tüm hareketlerin tüm yıkama oturumları için ayrı birleşik görüntüler oluşturulmuştur. Şekil 112’de bu görüntülerden bir örnek verilmektedir. Modellerin verdiği çıktılar çerçeveler üzerine yazılmıştır. Bu çıktılarda ilk karakter rakam olarak 1’den 8’e hareketi ifade etmekte, ikinci karakter ise sabun miktarını ifade etmektedir; n, no soap (sabunsuz), d; deficient soap (az sabunlu), m; normal soap (normal sabunlu), a; adequate soap (çok sabunlu).

Modellerin sentetik veri seti üzerindeki eğitim grafikleri Şekil 110’da görülmektedir. Şekil 111’de ise, beş kat çapraz doğrulamaya ait modellerin sentetik veri setindeki test verilerinde elde ettiği ortalama hata matrisi görülmektedir. Burada doğruluk oranı %95,11 olmuştur. Hata matrisinde sınıflar 1-32 arasında gösterilmiştir. Sınıflara atanan sayılar, 1: 1. hareket sabunsuz, 2: 1. hareket az sabunlu, 3: 1. hareket normal sabunlu, 4: 1. hareket çok sabunlu şeklinde başlayıp, 5: 2. hareket sabunsuz, 6: 2. hareket az sabunlu, 2. hareket normal sabunlu, 2. hareket çok sabunlu şeklinde tüm hareketlerde devam etmektedir.



Şekil 110. Inception-V3 32 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri



Şekil 111. Inception-V3 32 sınıflı modelin sentetik veri seti üzerinde hata matrisi

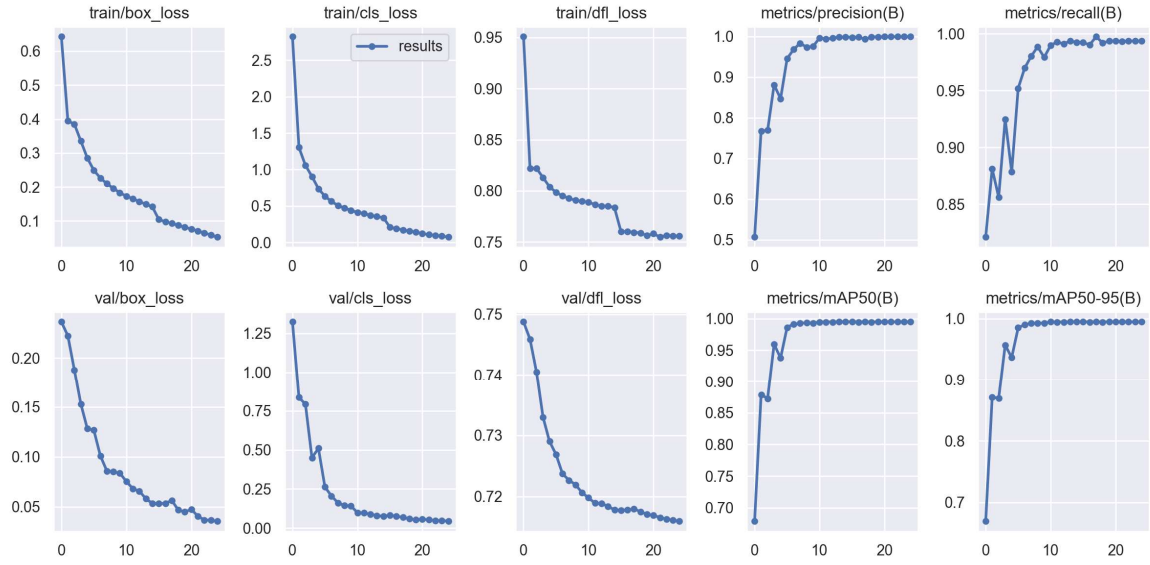


Şekil 112. Inception-V3 ağı 32 sınıflı modelinin sınıflandırmasından örnekler (birleşik görüntü)

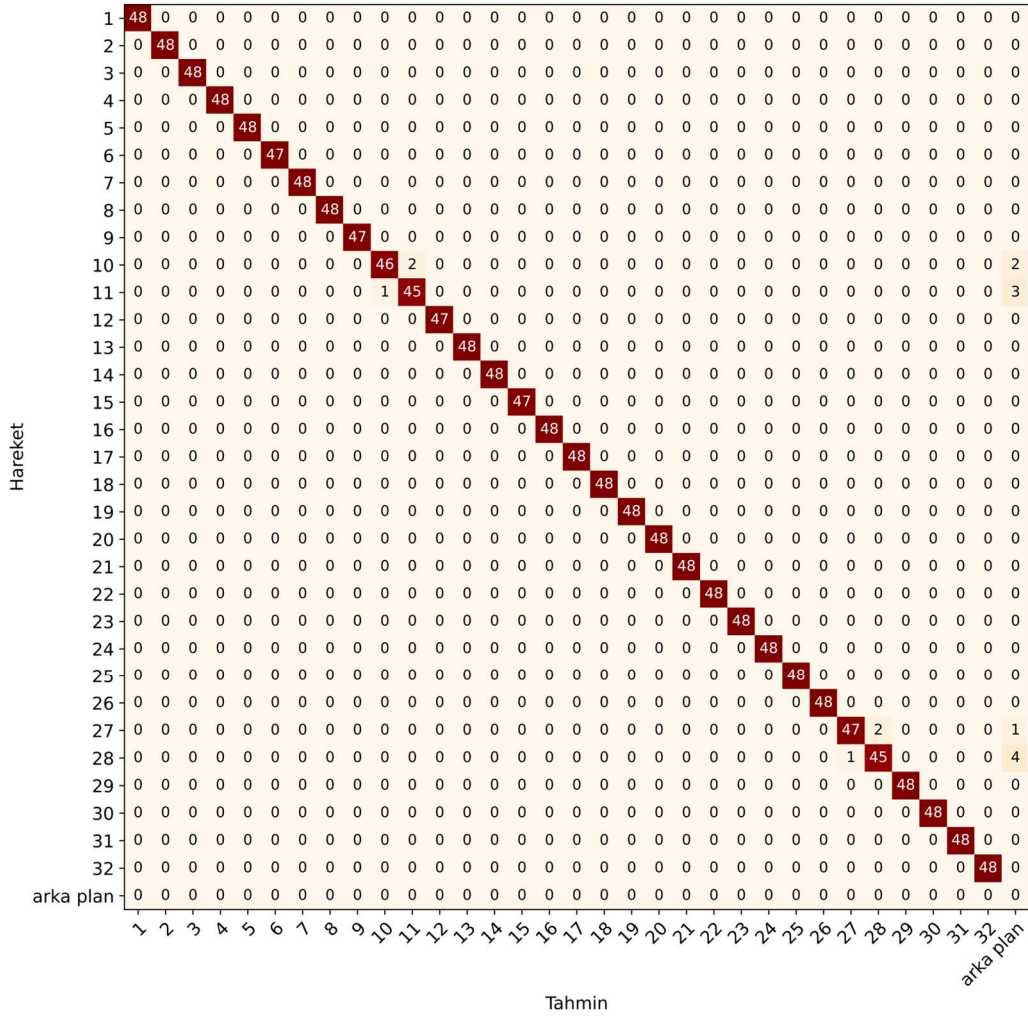
3.2.7. YOLO-8n ile Sabun Miktarı ve Hareket Sınıflandırma

YOLO-8n 32 sınıflı modelde de diğer modellerdeki gibi, kamera veri setindeki tüm hareketlere ait her bir oturumdaki kırılmamış RGB çerçeveler için, beş kat çapraz doğrulama modellerinin el bölgesi tespiti ve sınıflandırma sonucu çerçeveler üzerine yazılmış, bu çerçeveler yan yana birleştirilerek görüntüler elde edilmiştir. Şekil 116’da bu birleşik görüntülerden bir örnek görülmektedir.

Şekil 113’te, beş modelden birine ait eğitim grafikleri görülmektedir. x eksenı epok miktarını, y eksenı ise grafiklerin üzerinde bulunan parametre değerini belirtmektedir. Şekil 114’te sentetik veri seti üzerindeki test verilerinde beş kat çapraz doğrulama modellerinin ortalama hata matrisi görülmektedir. Burada doğruluk oranı %99,15 olmuştur. Hata matrisinde sınıflar 1-32 arasında gösterilmiştir. Sınıflara atanan sayılar, 1: 1. hareket sabunsuz, 2: 1. hareket az sabunlu, 3: 1. hareket normal sabunlu, 4: 1. hareket çok sabunlu şeklinde başlayıp, 5: 2. hareket sabunsuz, 6: 2. hareket az sabunlu, 7: 2. hareket normal sabunlu, 8: 2. hareket çok sabunlu şeklinde tüm hareketlerde devam etmektedir. Şekil 115’de etiketli eğitim verilerinin eğitimden önce hazırlandığı bir görüntü yer almaktadır.



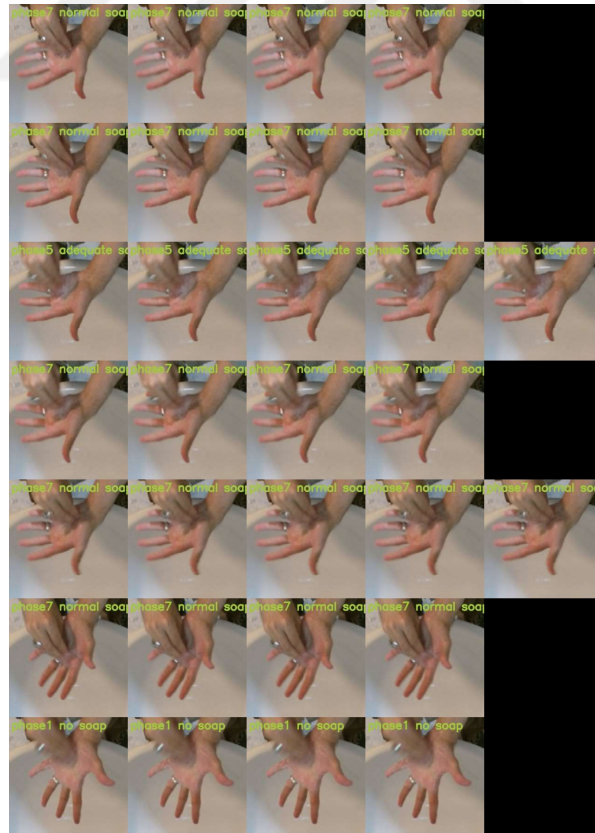
Şekil 113. YOLO-8n 32 sınıflı modelin sentetik veri seti üzerindeki eğitim grafikleri



Şekil 114. YOLO-8n 32 sınıflı modelin sentetik veri seti üzerinde hata matrisi



Şekil 115. YOLO-8n 32 sınıflı modelin eğitim verileri ve etiketlerinden örnekler



Şekil 116. YOLO-8n ağı 32 sınıflı modelinin sınıflandırmasından örnekler (birleşik görüntü)

3.3. Literatür ile Karşılaştırma

Literatürde bulunan el yıkama hareketi sınıflandırma çalışmaları kategorik olarak Tablo 7’de özetlenmiştir. Çalışmalarda veri seti, hareket miktarı ve yöntem (görüntü / sensör) farklılıkları olduğu için, doğrudan bir karşılaştırma yapmak mümkün değildir. Çalışmalara ait doğruluk oranlarında, kullanılan yöntemler içerisinde en yüksek doğruluk değeri ya da bir yöntem içerisinde en doğru hareket sınıflandırmasının oranı verilmiştir.

Tablo 7. Literatürde bulunan el yıkama tanıma çalışmalarının genel özeti

Çalışma	Hareket Sayısı	Yöntem	Doğruluk Oranı (%)
Llorca vd., 2011	6	Görüntü bazlı	97,81
Xia vd., 2015	12	Görüntü bazlı	100,00
Ivanovs vd., 2020	7	Görüntü bazlı	67,00
Prakasa ve Sugiarto, 2020	6	Görüntü bazlı	98,81
Vo vd., 2021	7	Görüntü bazlı	84,00
Zhong vd., 2021	4	Görüntü bazlı	98,10
Xie vd., 2022	7	Görüntü bazlı	93,75
Özakar & Gedikli, 2023	8	Görüntü bazlı	100,00
Mondol & Stankovic, 2015	7	Sensör	~60,00
Wang vd., 2020	10	Sensör	90,90
Samyoun vd., 2021	10	Sensör	~99,00
Santos-Gago vd., 2021	-	Sensör	100,00
Wang vd., 2021	9	Sensör	92,27
Zhang vd., 2021	11	Sensör	98,00

Literatürdeki çalışmalarda, sensör bazlı yaklaşımların genelinde kullanıcıların bir tür cihaz giymeleri gerekmektedir. Günlük hayatta el hijyeni takibi gerektiren durumlarda bu kısıtlayıcı ya da rahatsızlık verici olabilir. Bu bağlamda, tez çalışmasında görüntü bazlı bir yöntem benimsenmesi sensör bazlı yaklaşımlara göre avantajlı olacaktır. Görüntü bazlı çalışmaların bir kısmında ise, kamera yerleşim yerinin ellerin ikisini de eşit miktarda görmeyen veya her lavaboya uygun olmayan şekilde olduğu görülmüştür. Tez çalışmasında

önerilen kamera konumu avantajlı ve uygulanabilir olmaktadır. Görüntü bazlı çalışmalar içerisinde veri tipi çeşitliliği ve sınıflandırıcı çeşitliliği göz önünde bulundurulduğunda, tez çalışması en kapsamlı çalışmalardan biri olmuştur. Doğruluk oranı açısından değerlendirildiğinde, PointNet-LSTM modelinde sağlanan %100 doğruluk oranıyla çalışmalar arasında en yüksek doğruluk oranlarından birine sahip olmuştur. Derinlik görüntülerinden elde edilen noktalar kümesi verilerinin sağladığı yüksek başarı, renkli görüntülerde ışık, deri rengi ve arka plan değişimlerinden oluşan renk farklılığından etkilenmemesi ve PointNet ağının mimarisiyle ilişkilendirilebilir.

Elde edilen bulgular sonucunda, derinlik görüntüsü ile lavabo ortamında ellerin tespiti kolayca yapılabilmiştir. Burada lavabonun ve kameranın sabit konumu, el ile sensör arasında görüntü yakalamayı engelleyici unsurların olmaması rol oynamaktadır. Sınıflandırmada elde edilen doğruluk oranlarına bakıldığında, bütün veri ve sınıflandırma modellerin kabul edilebilir düzeyde başarı sağladığı görülmektedir.

Projeksiyon görüntüleriyle eğitimin daha önce bahsedildiği gibi özel bir algoritmayla daha az miktarlara bölünmüş doğrulama verisiyle yapılmış olması, doğruluk oranını etkileyen faktörler arasında sayılabilir. Yeterli bellek miktarı olan bir sistemde diğer ağlardaki gibi yüksek miktarda doğrulama verisiyle eğitimin daha iyi sonuç elde edilebileceği öngörülebilir.

4. SONUÇLAR

Bu tez çalışmasında, kişisel el hijyeninin denetlenmesinin zorunlu olduğu durumlarda kullanılabilmesi amacıyla, bilgisayarlı görme ile DSÖ tarafından önerilen hareketleri içeren 8 el yıkama hareketini tespit edebilecek bir sistemin çalışmaları yapılmıştır. Bu sistem, Kinect 2 kamerasıyla oluşturulan, renkli ve derinlik görüntüleri içeren, iki farklı mekanda iki kişiyle elde edilmiş bir veri setiyle eğitilmiş yapay sinir ağlarından oluşmaktadır. Ayrıca, sentetik görüntüler kullanılarak hazırlanmış yeni bir veri setinin sağlayabileceği katkılar ölçülmeye çalışılmıştır. Tez çalışmasında elde edilen sonuçlar ve sunulan katkılar aşağıdaki şekilde özetlenebilir;

- İki özgün veri seti üretilmiştir.
- Renkli ve derinlik görüntülerini aynı anda işleyebilen iş parçacığı bazlı gerçek zamanlı bir sistem üretilmiştir.
- Literatürde yer alan diğer çalışmalarda, dezavantajlı kamera konumları, giyilmesi gereken sensörler ve yetersiz veri modeli problemleri bulunmaktadır. Bunlar kullanıcı açısından rahatsızlık verici olabilir. Bu durumlara bilgisayarlı görme bazlı kullanıcıyı rahatsız etmeyecek bir çözüm üretilmiştir.
- Problem için uygun bir kamera pozisyonu önerilmiştir. Bu kamera pozisyonuna bağlı olarak, düşük maliyetli ve pratik bir el takip sistemi önerilmiştir.
- İki elin birleşik biçimde bulunduğu el hareketlerini tanımlamak için iki el tek bir obje gibi kabul edilip sınıflandırılmıştır. Bu yaklaşımın birbirinin simetriği olan el hareketleri için dahi başarılı sonuçlar elde ettiği gözlenmiştir.
- 16 yapay sinir ağı modelinin doğruluk oranları ölçülmüştür.
- 2 yenilikçi veri tipiyle beraber 4 farklı veri modelinin kullanım senaryoları ve başarıları ölçülmüştür (RGB, Projeksiyon, Noktalar Kümesi, Birleşik Noktalar Kümesi).
- Sentetik bir veri setinin elde edebileceği başarılar ölçülmüştür. Bu veri setinde, ellerin farklı sabun miktarları sınıflandırılmaya çalışılmıştır.

Kamera veri setiyle yapılan alıřmалardaki aęların yksek doęruluk oranları elde etmiřtir. Aęlardan PointNet-LSTM modeli %100 doęruluk oranıyla sınıflandırma gerekleřtirebilmiřtir. Bu da tez senaryosundaki problemin bilgisayarlı grme yntemleriyle zlebileceęini gstermiřtir.

Sentetik veri setiyle yapılan alıřmada, 4, 8 ve 32 sınıflara ayrılmıř veri setleri ile eęitilen iki farklı aę trnn gerek veriler zerindeki doęruluk oranları llmřtir. Etiketli olması sebebiyle kesin olarak doęruluk oranı llebilen 8 sınıflı modellerin doęruluk oranları Inception-V3 iin %47,5, YOLO-8n iin %56,5 olarak llmřtir. Inception-V3 modellerinde, el blgelerinin tespitinin nceden yapıldıęı varsayılmaktadır. YOLO modellerinde ise el tespiti de aę ile yapılmaktadır. 4 sınıflı YOLO aęında el tespit oranı %70, 8 sınıflı YOLO aęında %63, 32 sınıflı YOLO aęında ise %65 olmuřtur. Bu doęruluk oranlarına bakılarak, tamamen sentetik verilerle eęitilmiř aęın gerek verileri sınıflandırma zerinde yardımcı etkisi olduęu sylenebilir. İki farklı aę trnn de hareket sınıflandırmada daha bařarılı olduęu, ancak ellerdeki sabun miktarını tespit etmede veya hem hareketi hem sabun miktarını aynı anda doęru biimde tespit etmede yetersiz kaldıęı gzlenmiřtir. Sentetik veri setinde oluřturulan verilerin sayısının, eřitlilięinin ve gerekilięinin artırılması sonucunda, daha bařarılı sonular elde edilebileceęi ngrlebilir. alıřmada kamera veri seti ile elde edilen sonular makale olarak yayınlanmıřtır (zakar & Gedikli, 2023).

5. ÖNERİLER

Bu tez çalışmasında, gerçek hayattan veriler toplanarak prosedürel el yıkama hareketlerinin doğru uygulanıp uygulanmadığı denetlenmiştir. Bunun için derinlik sensörü ile nokta bulutu verileri ve RGB görüntüler kullanılmıştır. Ayrıca, sentetik veriler üzerinden sabun kullanımı araştırılmıştır. Çalışmada 4 farklı örüntü modeli önerilmiş ve 16 farklı ağ ile sınıflandırma yapılmıştır. 8 el hareketinin farklı zamanlarda gerçekleştirilme ihtimaline karşılık, çerçeve bazlı hareket tanıma yapılmış ve el yıkama süreci incelenmiştir. Çalışmada mevcut yaklaşımla başarılı sonuçlar elde edilmiştir. Çalışmanın devamında geliştirilebilecek kısımlar aşağıdaki gibi sıralanabilir;

- Çalışmada, el yıkama sürecinin DSÖ'nün önerilen el yıkama hareketlerine uygun bir biçimde ve sürede gerçekleştirilmesine odaklanılmıştır. El hijyeni, sabun veya dezenfektan gibi kimyasal maddelerle sağlanabilmektedir. Çalışmada sentetik veri üretilerek, sabun miktarına göre 4 ve 32 sınıflı bir veri seti üretilmiştir. Gerçek ve sentetik veriler; farklı sabun miktarları, çeşitli dezenfektanlar kullanılarak daha da genişletilebilir.
- Farklı sınıflandırma modelleri kullanılarak hesaplama maliyeti düşürülebilir.
- Mevcut çalışma, lavabo üzerine entegre edilen Kinect kamera ile yapılmıştır. Bu şekilde yoğun bakım gibi alanlarda kullanımı mümkündür. Ancak farklı sensörler kullanılarak mevcut yaklaşım daha modüler hale getirilebilir. Sistem gömülü hale getirilebilir.
- Çalışma, ellerin yeterince temas edip etmediğini tespit edecek ek sensörler veya kameralarla genişletilebilir. Görüntü işlemede bulunan çerçeve farkı ya da optik akış yöntemi gibi yöntemlerle ellerin hareketinin denetlemesi sağlanabilir.
- Çalışma, sahada görevli aktif kişiler için geliştirildiğinden, veri setlerinde fazla kişiye gerek duyulmamıştır. Ancak toplu alanlarda, yaşlı, çocuk veya engelli bireylerin olduğu yerlerde kişisel hijyen için veri setindeki kişi sayısının artırılması gerekebilir.

6. KAYNAKLAR

- Abavisani, M., Joze, H. R. V., & Patel, V. M. (2019). Improving the performance of unimodal dynamic hand-gesture recognition with multimodal training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1165-1174).
- Alcantarilla, P. F., Bartoli, A., & Davison, A. J. (2012). KAZE features. In *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12* (pp. 214-227). Springer Berlin Heidelberg.
- Alcantarilla, P. F., Nuevo, J., & Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces british machine vision conference (BMVC).
- Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.
- Amin, M. G., Zeng, Z., & Shan, T. (2019, April). Hand gesture recognition based on radar micro-Doppler signature envelopes. In *2019 IEEE Radar Conference (RadarConf)* (pp. 1-6). IEEE.
- Azad, R., Asadi-Aghbolaghi, M., Kasaei, S., & Escalera, S. (2018). Dynamic 3D hand gesture recognition by learning weighted depth motion maps. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(6), 1729-1740.
- Bambach, S., Lee, S., Crandall, D. J., & Yu, C. (2015). Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In *Proceedings of the IEEE international conference on computer vision* (pp. 1949-1957).
- Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. *Lecture notes in computer science*, 3951, 404-417.
- Ben Jmaa, A., Mahdi, W., Ben Jemaa, Y., & Ben Hamadou, A. (2016). A new approach for hand gestures recognition based on depth map captured by rgb-d camera. *Computación y Sistemas*, 20(4), 709-721.
- Benmoussa, M., & Mahmoudi, A. (2018, April). Machine learning for hand gesture recognition using bag-of-words. In *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)* (pp. 1-7). IEEE.
- Birdal, A., & Hassanpour, R. (2008). Region based hand gesture recognition.
- Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.
- Black, M. J., & Jepson, A. D. (1998). Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26, 63-84.

- Boulahia, S. Y., Anquetil, E., Multon, F., & Kulpa, R. (2017, November). Dynamic hand gesture recognition based on 3D pattern assembled trajectories. In *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)* (pp. 1-6). IEEE.
- Bretzner, L., Laptev, I., & Lindeberg, T. (2002, May). Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In *Proceedings of fifth IEEE international conference on automatic face gesture recognition* (pp. 423-428). IEEE.
- Buehler, P., Everingham, M., Huttenlocher, D. P., & Zisserman, A. (2008). Long term arm and hand tracking for continuous sign language TV broadcasts. In *Proceedings of the 19th British Machine Vision Conference* (pp. 1105-1114). BMVA Press.
- Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *Computer Vision—ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11* (pp. 778-792). Springer Berlin Heidelberg.
- Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6), 679-698.
- Chen, F. S., Fu, C. M., & Huang, C. L. (2003). Hand gesture recognition using a real-time tracking method and hidden Markov models. *Image and vision computing*, 21(8), 745-758.
- Chen, X., Wang, G., Guo, H., Zhang, C., Wang, H., & Zhang, L. (2019). MFA-Net: Motion Feature Augmented Network for Dynamic Hand Gesture Recognition from Skeletal Data. *Sensors*, 19(2), 239.
- Chen, Z. H., Kim, J. T., Liang, J., Zhang, J., & Yuan, Y. B. (2014). Real-time hand gesture recognition using finger segmentation. *The scientific world journal*, 2014.
- Cheng, S. T., Hsu, C. W., & Li, J. P. (2013). Combined hand gesture—speech model for human action recognition. *Sensors*, 13(12), 17098-17129.
- Coleman, G. B., & Andrews, H. C. (1979). Image segmentation by clustering. *Proceedings of the IEEE*, 67(5), 773-785.
- Coogan, T., Awad, G., Han, J., & Sutherland, A. (2006, November). Real time hand gesture recognition including hand segmentation and tracking. In *International Symposium on Visual Computing* (pp. 495-504). Springer, Berlin, Heidelberg.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20, 273-297.
- Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (Vol. 1, pp. 886-893). Ieee.

- Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7), 1160-1169.
- de La Gorce, M., Paragios, N., & Fleet, D. J. (2008, June). Model-based hand tracking with texture, shading and self-occlusions. In *2008 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1-8). IEEE.
- Dechter, R. (1986). Learning while searching in constraint-satisfaction problems.
- Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.
- Elmezain, M., Al-Hamadi, A., & Michaelis, B. (2009, December). Improving hand gesture recognition using 3D combined features. In *2009 Second International Conference on Machine Vision* (pp. 128-132). IEEE.
- Fang, Y., Wang, K., Cheng, J., & Lu, H. (2007, July). A real-time hand gesture recognition method. In *2007 IEEE International Conference on Multimedia and Expo* (pp. 995-998). IEEE.
- Felzenszwalb, P. F., & Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International journal of computer vision*, 59, 167-181.
- Fix, E. and Hodges, J.L. (1951) Discriminatory Analysis, Nonparametric Discrimination: Consistency Properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field.
- Gabor, D. (1946). Theory of communication. Part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-part III: radio and communication engineering*, 93(26), 429-441.
- Garcia-Garcia, B., Bouwmans, T., & Silva, A. J. R. (2020). Background subtraction in real applications: Challenges, current models and future directions. *Computer Science Review*, 35, 100204.
- Gardner, H. E. (2000). *Intelligence reframed: Multiple intelligences for the 21st century*. Hachette UK.
- Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).
- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).
- Gonzales, R. C., & Woods, R. E. (2018). Digital image processing 4th edition.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.
- Horn, B. K., & Schunck, B. G. (1981). Determining optical flow. *Artificial intelligence*, 17(1-3), 185-203.
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.
- Ivanovs, M., Kadikis, R., Lulla, M., Rutkovskis, A., & Elsts, A. (2020). Automated quality assessment of hand washing using deep learning. *arXiv preprint arXiv:2011.11383*.
- Kao, C. Y., & Fahn, C. S. (2011). A human-machine interaction technique: hand gesture recognition based on hidden Markov models with trajectory of hand motion. *Procedia Engineering*, 15, 3739-3743.
- Kapuscinski, T., Oszust, M., Wysocki, M., & Warchol, D. (2015). Recognition of hand gestures observed by depth cameras. *International Journal of Advanced Robotic Systems*, 12(4), 36.
- Kim, S. Y., Han, H. G., Kim, J. W., Lee, S., & Kim, T. W. (2017). A hand gesture recognition sensor using reflected impulses. *IEEE Sensors Journal*, 17(10), 2975-2976.
- Kim, Y., & Toomajian, B. (2016). Hand gesture recognition using micro-Doppler signatures with convolutional neural network. *IEEE Access*, 4, 7125-7130.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kishore, P. V. V., Prasad, M. V. D., Kumar, D. A., & Sastry, A. S. C. S. (2016, February). Optical flow hand tracking and active contour hand shape features for continuous sign language recognition with artificial neural networks. In 2016 IEEE 6th international conference on advanced computing (IACC) (pp. 346-351). IEEE.
- Kolkur, S., Kalbande, D., Shimpi, P., Bapat, C., & Jatakia, J. (2017). Human skin detection using RGB, HSV and YCbCr color models. *arXiv preprint arXiv:1708.02694*.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84-90.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436-444.

- Legg, S., & Hutter, M. (2007). A collection of definitions of intelligence. *Frontiers in Artificial Intelligence and applications*, 157, 17.
- Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011, November). BRISK: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision* (pp. 2548-2555). Ieee.
- Li, C., Zhang, X., & Jin, L. (2017). LPSNet: A novel log path signature feature based hand gesture recognition framework. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 631-639).
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14* (pp. 21-37). Springer International Publishing.
- Liu, X., & Fujimura, K. (2004, May). Hand gesture recognition using depth data. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.* (pp. 529-534). IEEE.
- Liu, Z., Chai, X., Liu, Z., & Chen, X. (2017). Continuous gesture recognition with hand-oriented spatiotemporal feature. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (pp. 3056-3064).
- Llorca, D. F., Parra, I., Sotelo, M. Á., & Lacey, G. (2011). A vision-based system for automatic hand washing quality assessment. *Machine Vision and Applications*, 22, 219-234.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60, 91-110.
- Ma, Z., & Wu, E. (2014). Real-time and robust hand tracking with a single depth camera. *The Visual Computer*, 30(10), 1133-1144.
- MacCormick, J., & Isard, M. (2000). Partitioned sampling, articulated objects, and interface-quality hand tracking. In *Computer Vision—ECCV 2000: 6th European Conference on Computer Vision Dublin, Ireland, June 26–July 1, 2000 Proceedings, Part II 6* (pp. 3-19). Springer Berlin Heidelberg.
- MacQueen, J. (1967, June). Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability* (pp. 281-297). Los Angeles LA USA: University of California.
- McConnell, R. K. (1986). *U.S. Patent No. 4,567,610*. Washington, DC: U.S. Patent and Trademark Office.
- Mitra, S., & Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3), 311-324.

- Mohamed, N., Mustafa, M. B., & Jomhari, N. (2021). A review of the hand gesture recognition system: Current progress and future directions. *IEEE Access*, 9, 157422-157436.
- Molchanov, P., Gupta, S., Kim, K., & Pulli, K. (2015, May). Multi-sensor system for driver's hand-gesture recognition. In 2015 11th IEEE international conference and workshops on automatic face and gesture recognition (FG) (Vol. 1, pp. 1-8). IEEE.
- Mondol, M. A. S., & Stankovic, J. A. (2015). Harmony: A hand wash monitoring and reminder system using smart watches. *EAI Endorsed Transactions on Ambient Systems*, 2(6), 11-20.
- Mueller, F., Davis, M., Bernard, F., Sotnychenko, O., Verschoor, M., Otaduy, M. A., ... & Theobalt, C. (2019). Real-time pose and shape reconstruction of two interacting hands with a single depth camera. *ACM Transactions on Graphics (ToG)*, 38(4), 1-13.
- Muja, M., & Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340), 2.
- Nabiyev, V. V. (2012). Yapay zeka: insan-bilgisayar etkileşimi. Seçkin Yayıncılık.
- Ohn-Bar, E., & Trivedi, M. M. (2014). Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE transactions on intelligent transportation systems*, 15(6), 2368-2377.
- Ong, E. J., & Bowden, R. (2004, May). A boosted classifier tree for hand shape detection. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.* (pp. 889-894). IEEE.
- Owoyemi, J., & Hashimoto, K. (2018, May). Spatiotemporal learning of dynamic gestures from 3d point cloud data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5929-5934). IEEE.
- Özakar, R., & Gedikli, E. (2023). Evaluation of Hand Washing Procedure Using Vision-Based Frame Level and Spatio-Temporal Level Data Models. *Electronics*, 12(9), 2024.
- Patwardhan, K. S., & Roy, S. D. (2004). Dynamic hand gesture recognition using predictive eigentracker. *GESTURE*, 32(38), 48.
- Prakasa, E., & Sugiarto, B. (2020, November). Video analysis on handwashing movement for the completeness evaluation. In *2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)* (pp. 296-301). IEEE.
- Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).
- Qian, C., Sun, X., Wei, Y., Tang, X., & Sun, J. (2014). Realtime and robust hand tracking from depth. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1106-1113).

- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.
- Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6), 386.
- Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9* (pp. 430-443). Springer Berlin Heidelberg.
- Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011, November). ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision* (pp. 2564-2571). Ieee.
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533-536.
- Samyoun, S., Shubha, S. S., Mondol, M. A. S., & Stankovic, J. A. (2021). iWash: A smartwatch handwashing quality assessment and reminder system with real-time feedback in the context of infectious disease. *Smart Health*, 19, 100171.
- Santos-Gago, J. M., Ramos-Merino, M., & Álvarez-Sabucedo, L. M. (2021). Identification of free and WHO-compliant handwashing moments using low cost wrist-worn wearables. *IEEE Access*, 9, 133574-133593.
- Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5, 197-227.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61, 85-117.
- Sefat, M. S., & Shahjahan, M. (2015, May). A hand gesture recognition technique from real time video. In *2015 International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)* (pp. 1-4). IEEE.
- Sezgin, M., & Sankur, B. L. (2004). Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic imaging*, 13(1), 146-168.
- Sharp, T., Keskin, C., Robertson, D., Taylor, J., Shotton, J., Kim, D., ... & Izadi, S. (2015, April). Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems* (pp. 3633-3642).
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

- Song, Y., Demirdjian, D., & Davis, R. (2012). Continuous body and hand gesture recognition for natural human-computer interaction. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(1), 5.
- Starner, T., Weaver, J., & Pentland, A. (1998). Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on pattern analysis and machine intelligence*, 20(12), 1371-1375.
- Stauffer, C., & Grimson, W. E. L. (1999, June). Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149)* (Vol. 2, pp. 246-252). IEEE.
- Stenger, B., Mendonça, P. R., & Cipolla, R. (2001, December). Model-based 3D tracking of an articulated hand. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* (Vol. 2, pp. II-II). IEEE.
- Stenger, B., Thayananthan, A., Torr, P. H., & Cipolla, R. (2006). Model-based hand tracking using a hierarchical bayesian filter. *IEEE transactions on pattern analysis and machine intelligence*, 28(9), 1372-1384.
- Suk, H. I., Sin, B. K., & Lee, S. W. (2010). Hand gesture recognition based on dynamic Bayesian network framework. *Pattern recognition*, 43(9), 3059-3072.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826).
- Szeliski, R. (2022). *Computer vision: algorithms and applications*. Springer Nature.
- Tang, D., Jin Chang, H., Tejani, A., & Kim, T. K. (2014). Latent regression forest: Structured estimation of 3d articulated hand posture. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3786-3793).
- Tompson, J., Stein, M., Lecun, Y., & Perlin, K. (2014). Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5), 169.
- Turing, A. M. (2009). *Computing machinery and intelligence* (pp. 23-65). Springer Netherlands.
- Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1), 71-86.
- URL-1. (2023, Eylül 28) Optik karakter okuyucu tarihi: https://en.wikipedia.org/wiki/Timeline_of_optical_character_recognition
- URL-2 (2023, Eylül 28) DSÖ'nün önerilen el yıkama prosedürleri: <https://www.who.int/multi-media/details/how-to-handwash-and-handrub>

- URL-3. (2023, 28 Eylül) Microsoft Kinect 2 Kamera: <https://developer.microsoft.com/en-us/windows/kinect/>
- URL-4. (2023, 28 Eylül) Derinlik kameraları karşılaştırması: https://youtu.be/NrIgjk_PeQU/
- URL-5. (2023, 28 Eylül) Derinlik kameraları: <https://rosindustrial.org/3d-camera-survey/>
- URL-6. (2023, 28 Eylül) Gaussian Karışım Modelleriyle arka plan çıkarımı: https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html
- URL-7. (2023, 28 Eylül) Şablon eşleştirme: https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html/
- URL-8. (2023, Eylül 28) YOLO-V8: <https://github.com/ultralytics/ultralytics>
- URL-9. (2023, Eylül 28) YOLO-V8 ağının yapısı: <https://github.com/ultralytics/ultralytics/issues/189>
- URL-10. (2023, Eylül 28) Python: <https://www.python.org/>
- URL-11. (2023, Eylül 28) OpenCV: <https://opencv.org/>
- URL-12. (2023, Eylül 28) PyKinect2: <https://github.com/Kinect/PyKinect2>
- URL-13. (2023, Eylül 28) Tensorflow: <https://www.tensorflow.org/>
- URL-14. (2023, Eylül 28) Keras: <https://keras.io/about/>
- URL-15. (2023, Eylül 28) Pytorch: <https://pytorch.org/>
- URL-16. (2023, Eylül 28) Blender: <https://www.blender.org/>
- URL-17. (2023, Eylül 28) GIMP: <https://www.gimp.org/>
- URL-18. (2023, Eylül 28) Makehuman: <https://www.makehumancommunity.org/>
- URL-19. (2024, Ocak, 15) Scikit-Learn: <https://scikit-learn.org/>
- URL-20. (2023, Eylül 28) Önceden eğitilmiş Haar basamakları <https://github.com/Aravindliewire/Opencv/tree/master/haarcascade>
- URL-21. (2023, Eylül 28) https://github.com/tensorflow/models/tree/master/research/object_detection
- URL-22. (2023, Eylül 28) Kinect 2.0 kamerası <https://en.wikipedia.org/wiki/Kinect#/media/File:Xbox-One-Kinect.jpg>
- URL-23. (2023, Eylül 28) Open3d: <http://www.open3d.org/>
- URL-24. (2023, Eylül 28) Keras PointNet: <https://keras.io/examples/vision/pointnet/>

- Urooj, A., & Borji, A. (2018). Analysis of hand segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pa*
- Victor, D. (2017). Handtrack: A library for prototyping real-time hand tracking interfaces using convolutional neural networks. *GitHub repository*, 3(4), 6.
- Viola, P., & Jones, M. (2001, December). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (Vol. 1, pp. I-I). Ieee.
- Vo, H. Q., Do, T., Pham, V. C., Nguyen, D., Duong, A. T., & Tran, Q. D. (2021, October). Fine-grained hand gesture recognition in multi-viewpoint hand hygiene. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 1443-1448). IEEE.
- Wang, C., Liu, Z., & Chan, S. C. (2014). Superpixel-based hand gesture recognition with kinect depth camera. *IEEE transactions on multimedia*, 17(1), 29-39.
- Wang, C., Sarsenbayeva, Z., Chen, X., Dingler, T., Goncalves, J., & Kostakos, V. (2020). Accurate measurement of handwash quality using sensor armbands: Instrument validation study. *JMIR mHealth and uHealth*, 8(3), e17001.
- Wang, F., Wu, X., Wang, X., Chi, J., Shi, J., & Huang, D. (2021). You can wash better: Daily handwashing assessment with smartwatches. *arXiv preprint arXiv:2112.06657*.
- Wang, R. Y., & Popović, J. (2009). Real-time hand-tracking with a color glove. *ACM transactions on graphics (TOG)*, 28(3), 1-8.
- Wiener-Well, Y., Galuty, M., Rudensky, B., Schlesinger, Y., Attias, D., & Yinnon, A. M. (2011). Nursing and physician attire as possible source of nosocomial infections. *American journal of infection control*, 39(7), 555-559.
- Xia, B., Dahyot, R., Ruttle, J., Caulfield, D., & Lacey, G. (2015). Hand hygiene poses recognition with rgb-d videos. In *Proceedings of the 17th Irish Machine Vision and Image Processing Conference* (pp. 43-50). Irish Pattern Recognition & Classification Society.
- Xie, T., Tian, J., & Ma, L. (2022). A vision-based hand hygiene monitoring approach using self-attention convolutional neural network. *Biomedical Signal Processing and Control*, 76, 103651.
- Xu, P. (2017). A real-time hand gesture recognition and human-computer interaction system. *arXiv preprint arXiv:1704.07296*.
- Yun, L., Lifeng, Z., & Shujun, Z. (2012). A hand gesture recognition method based on multi-feature fusion and template matching. *Procedia Engineering*, 29, 1678-1684.
- Zabulis, X., Baltzakis, H., & Argyros, A. A. (2009). Vision-Based Hand Gesture Recognition for Human-Computer Interaction. *The universal access handbook*, 34, 30.

- Zhang, X., Chen, X., Li, Y., Lantz, V., Wang, K., & Yang, J. (2011). A framework for hand gesture recognition based on accelerometer and EMG sensors. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(6), 1064-1076.
- Zhang, Y., Xue, T., Liu, Z., Chen, W., & Vanrumste, B. (2021). Detecting hand washing activity among activities of daily living and classification of WHO hand washing techniques using wearable devices and machine learning algorithms. *Healthcare Technology Letters*, 8(6), 148-158.
- Zhang, Z., Tian, Z., & Zhou, M. (2018). Latern: Dynamic continuous hand gesture recognition using FMCW radar sensor. *IEEE Sensors Journal*, 18(8), 3278-3289.
- Zhong, C., Reibman, A. R., Mina, H. A., & Deering, A. J. (2021). Designing a Computer-Vision Application: A Case Study for Hand-Hygiene Assessment in an Open-Room Environment. *Journal of Imaging*, 7(9), 170.
- Zhou, Q., Gao, Y., Wang, X., Liu, R., Du, P., Wang, X., ... & Yang, K. (2020). Nosocomial infections among patients with COVID-19, SARS and MERS: a rapid review and meta-analysis. *Annals of Translational Medicine*, 8(10).
- Zurada, J. (1992). Introduction to artificial neural systems. West Publishing Co..

ÖZGEÇMİŞ

Rüstem ÖZAKAR İlköğrenimini Kültür Kurumu İlköğretim Okulu'nda tamamladı. Lise eğitimini 2003–2007 yılları arasında Erzurum Anadolu Lisesi'nde tamamladı. 2007–2011 yılları arasında Kocaeli Üniversitesi'nde Bilgisayar Mühendisliği bölümünde lisans eğitimini tamamladı. 2012 yılında Ankara'da özel bir şirkette yazılım mühendisi olarak çalıştı. 2013 yılında Erzurum Teknik Üniversitesi Bilgisayar Mühendisliği bölümünde araştırma görevlisi olarak göreve başladı. 2013–2016 yılları arasında Atatürk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda yüksek lisansını tamamladı. 2017 yılında Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda Doktora eğitimine başladı. Halen Erzurum Teknik Üniversitesi'nde araştırma görevlisi olarak görev yapmaktadır. İyi derecede İngilizce bilmektedir.

Özakar, R., & Gedikli, E. (2023). Evaluation of Hand Washing Procedure Using Vision-Based Frame Level and Spatio-Temporal Level Data Models. *Electronics*, 12(9), 2024.