

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ÇOK AMAÇLI OPTİMİZASYON TEKNİKLERİYLE
DAYANIKLI OTOMATİK CNN MODEL TASARIMI

DOKTORA TEZİ

Sefa ARAS

EKİM 2024

TRABZON



KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

**ÇOK AMAÇLI OPTİMİZASYON TEKNİKLERİYLE
DAYANIKLI OTOMATİK CNN MODEL TASARIMI**

Sefa ARAS

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

Tezin Enstitüye Verildiği Tarih : 17 / 09 / 2024

Tezin Savunma Tarihi : 25 / 10 / 2024

Tez Danışmanı : Dr. Öğr. Üyesi Eyüp GEDİKLİ

Trabzon 2024

ÖNSÖZ

“Çok Amaçlı Optimizasyon Teknikleriyle Dayanıklı Otomatik CNN Model Tasarımı” isimli bu tez Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Doktora Programı’nda hazırlanmıştır.

Tez çalışmam boyunca hiçbir desteğini esirgemeyen danışman hocam Sayın Dr. Öğr. Üyesi Eyüp GEDİKLİ’ye teşekkürü bir borç bilirim. Ayrıca, yapıcı eleştirileri ve önerileri ile tezime büyük katkıları bulunan saygıdeğer hocam Sayın Prof. Dr. Hamdi Tolga KAHRAMAN’a teşekkür ederim.

Bu tez çalışmasının her aşamasında yanımda olan, bana sevgi, ilgi ve bilgisini esirgemeyen, her zorluğu birlikte aşmamızda büyük rol oynayan sevgili eşim Elif ARAS’a en derin teşekkürlerimi sunarım. Ayrıca, her zaman manevi desteğini hissettiğim, sevgileri ve dualarıyla yanımda olan değerli aileme de sonsuz minnettarlığımı ifade etmek istiyorum. Bu doktora tezini sevgili kızım Asya Sare ARAS'a ithaf ediyorum.

Sefa ARAS

Trabzon 2024

TEZ ETİK BEYANNAMESİ

Doktora Tezi olarak sunduğum “Çok Amaçlı Optimizasyon Teknikleriyle Dayanıklı Otomatik CNN Model Tasarımı” başlıklı bu çalışmayı baştan sona kadar danışmanım Dr.Öğr.Üyesi Eyüp GEDİKLİ’nin sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuvarlarda yaptığımı/yaptırdığımı, 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. 25/10/2024

Sefa ARAS

İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ	III
TEZ ETİK BEYANNAMESİ.....	IV
İÇİNDEKİLER.....	V
ÖZET	VIII
SUMMARY	IX
ŞEKİLLER DİZİNİ.....	X
TABLolar DİZİNİ.....	XI
SEMBOLLER VE KISALTMALAR DİZİNİ	XII
1. GENEL BİLGİLER	1
1.1. Yapay Zeka.....	2
1.2. Derin Öğrenme	4
1.2.1. Yapay Sinir Ağları	5
1.2.2. Evrişimli Sinir Ağları.....	8
1.3. Hiper-Parametre Optimizasyonu	10
1.4. Yaygın ESA Mimarileri	12
1.4.1. İlk Derin Öğrenme: LeNet (1998)	12
1.4.2. AlexNet (2012): Derin Öğrenmenin Yükselişi	14
1.4.3. VGGNet (2014): Daha Derin Daha Basit	15
1.4.4. GoogLeNet/Inception (2014): Hesaplama Verimliliği	17
1.4.5. ResNet (2015): Derinliğin Gücü	17
1.4.6. DenseNet (2017): Yoğun Bağlantılar	18
1.4.7. MobileNet (2017): Hafif ESA.....	19
1.4.8. EfficientNet (2019): Verimlilik	20
1.5. Meta-Sezgisel Optimizasyon.....	21
1.5.1. Rulet Tekerleği Seçimi.....	23
1.5.2. Turnuva Seçimi Yöntemi	24
1.5.3. Aç-göz Algoritması.....	24
1.5.4. Elitizm Seçimi.....	25
1.5.5. Rastgele Seçim	25
1.5.6. Uygunluk-Çeşitlilik Dengesi Seçimi	25

1.6.	Çok Amaçlı Optimizasyon	26
1.6.1.	Pareto Tabanlı Yöntemler	26
1.6.2.	Ağırlıklandırma Yöntemleri.....	30
1.6.3.	Kısıt Tabanlı Yöntemler.....	32
1.7.	ESA Mimarilerinin Çok Amaçlı Optimizasyonu	33
2.	YAPILAN ÇALIŞMALAR.....	35
2.1.	Önerilen ÇESA Yöntemi.....	37
2.2.	Hiper-Parametrelerin ve Arama Uzayının Belirlenmesi	39
2.3.	Birey Kodlama Stratejisi	43
2.4.	Başlangıç Popülasyonun Oluşturulması.....	44
2.5.	Uygunluk Değerlendirme	46
2.6.	ÇAOA: Çok Amaçlı Optimizasyon Algoritması.....	47
2.6.1.	Baskın Olmayan Sıralama.....	49
2.6.2.	Kalabalık Mesafesi.....	51
2.7.	Önerilen ÇESA Modelleri	53
2.7.1.	ÇESA-1 Modeli.....	54
2.7.2.	ÇESA-2 Modeli.....	56
2.7.3.	ÇESA-3 Modeli.....	57
2.7.4.	ÇESA-4 Modeli.....	58
2.8.	Kullanılan Veri Setleri.....	59
2.8.1.	CIFAR-10.....	59
2.8.2.	CIFAR-100.....	60
2.8.3.	MNIST	60
2.8.4.	FRUITS-360.....	61
3.	BULGULAR VE TARTIŞMA.....	63
3.1.	DeneySEL Ayarlar.....	63
3.2.	MNIST Veri Setiyle Yapılan Deneyler	64
3.2.1.	MNIST İstatistiksel Analiz	64
3.2.2.	Çevrim Sayısının Performans Üzerine Etkisi	66
3.3.	CIFAR-10 Veri Setiyle Yapılan Deneyler	67
3.3.1.	CIFAR-10 İstatistiksel Analiz.....	67
3.3.2.	CIFAR-10 Yakınsama Analizi	71
3.4.	CIFAR-100 Veri Setiyle Yapılan Deneyler	73
3.4.1.	CIFAR-100 İstatistiksel Analiz.....	73

3.4.2. CIFAR-100 Yakınasama Analizi	76
3.5. FRUITS-360 Veri Setiyle Yapılan Deneyler	79
3.5.1. FRUITS-360 İstatistiksel Analiz	79
3.5.2. FRUITS-360 Yakınasama Analizi	82
3.6. Genel Değerlendirme	84
4. SONUÇLAR	89
5. ÖNERİLER	91
6. KAYNAKLAR	93
ÖZGEÇMİŞ	



Doktora Tezi

ÖZET

ÇOK AMAÇLI OPTİMİZASYON TEKNİKLERİYLE DAYANIKLI OTOMATİK CNN MODEL TASARIMI

Sefa ARAS

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr.Öğr.Üyesi Eyüp GEDİKLİ
2024, 104 Sayfa

Bu tezde, çok amaçlı optimizasyon teknikleriyle dayanıklı ve otomatik evrişimli sinir ağı (ESA) modellerinin tasarımı için yenilikçi bir yöntem geliştirilmiştir. Derin öğrenme modellerinin tasarımı, genellikle uzman bilgisi ve zaman alıcı manuel süreçler gerektirirken, önerilen Çok Amaçlı Evrişimli Sinir Ağları (ÇESA) yöntemi, bu süreci otomatikleştirerek modelin doğruluk oranını artırırken cevap süresini optimize etmeyi başarmıştır. Çalışma, farklı çok amaçlı optimizasyon algoritmalarını (örneğin NSGA-II, FDB-NSGA-II) kullanarak pareto optimal çözümler üretmiş, doğruluk ve hız arasındaki dengeyi başarılı bir şekilde sağlamıştır. Dört farklı veri seti üzerinde gerçekleştirilen deneyler, ÇESA yönteminin, mevcut ESA mimarilerine kıyasla %1.93 daha yüksek doğruluk ve %69.85 daha hızlı test süresi sağladığını göstermiştir. Özellikle düşük kapasiteli cihazlarda hafif ve hızlı modellerin üretilmesi, yöntemin mobil ve gömülü sistemlerde uygulanabilirliğini kanıtlamaktadır. Ayrıca, hiper-parametre optimizasyonunu tamamen otomatik hale getirerek, kullanıcıdan bağımsız bir şekilde esnek ve etkili modeller üretebilmesi, ÇESA'nın kullanım alanını genişletmektedir. Çalışma, ESA modellerinin tasarımında çok amaçlı optimizasyon yaklaşımlarının gücünü ortaya koyarak, hem akademik hem de endüstriyel uygulamalar için önemli bir rehber olma niteliği taşımaktadır.

Anahtar Kelimeler: Derin öğrenme, Evrişimli sinir ağları, Hiper-parametre optimizasyonu, Çok amaçlı optimizasyon

PhD. Thesis

SUMMARY

MULTI-OBJECTIVE OPTIMIZATION BASED ROBUST AUTOMATIC CNN MODEL DESIGN

Sefa ARAS

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Computer Engineering Graduate Program
Supervisor: Asst. Prof. Dr. Eyüp GEDİKLİ
2024, 104 Pages

This dissertation presents an innovative method for the design of robust and automated convolutional neural network (CNN) models using multi-objective optimization techniques. While the design of deep learning models often requires expert knowledge and time-intensive manual processes, the proposed CESA method automates this process, optimizing both model accuracy and response time. The study leverages various multi-objective optimization algorithms (e.g., NSGA-II, FDB-NSGA-II) to generate Pareto-optimal solutions, achieving a successful balance between accuracy and speed. Experiments conducted on four different datasets demonstrate that the CESA method delivers 1.93% higher accuracy and 69.85% faster test times compared to existing CNN architectures. The ability to produce lightweight and fast models, particularly for low-capacity devices, validates the method's applicability to mobile and embedded systems. Furthermore, by fully automating hyper-parameter optimization, CESA enables the generation of flexible and effective models independent of user intervention, broadening its range of applications. This study highlights the power of multi-objective optimization approaches in CNN model design, serving as a significant guide for both academic research and industrial applications.

Key Words: Deep learning, Convolutional neural networks, Hyper-parameter optimization, Multi-objective optimization

ŞEKİLLER DİZİNİ

	<u>Sayfa No</u>
Şekil 1. Yapay sinir ağı genel ağ modeli	6
Şekil 2. Evrişimli sinir ağı temel mimarisi	8
Şekil 3. LeNet-5 derin ağ modeli (LeCun vd., 1998)	13
Şekil 4. AlexNet derin ağ modeli (Krizhevsky vd., 2012)	14
Şekil 5. VGG-16 derin ağ modeli (Kamilaris ve Prenafeta-Boldú, 2018)	16
Şekil 6. Inception derin ağ modeli (Colaco vd., 2023)	17
Şekil 7. ResNet derin ağ modeli (Colaco vd., 2023)	18
Şekil 8. DenseNet derin ağ modeli (Jia vd., 2021)	19
Şekil 9. MobileNetV1 derin ağ modeli (Saini ve Devi, 2023)	20
Şekil 10. EfficientNetB0 derin ağ modeli (Tan ve Le, 2019)	21
Şekil 11. Pareto cephesi ve popülasyon dağılımı	27
Şekil 12. ÇESA yöntemiyle otomatik DÖ model tasarımı blok diyagramı	38
Şekil 13. CIFAR-10 veri seti üzerinde yapılan yakınsama analizi	72
Şekil 14. CIFAR-100 veri seti üzerinde yapılan yakınsama analizi	77
Şekil 15. FRUITS-360 veri seti üzerinde yapılan yakınsama analizi	83
Şekil 16. Test verisi doğruluk oranı genel değerlendirme	87
Şekil 17. Test verisi cevap süresi genel değerlendirme	87

TABLÖLAR DİZİNİ

	<u>Sayfa No</u>
Tablo 1. Evriřimli sinir aęlarında kullanılan hiper-parametreler	10
Tablo 2. Hiper-parametreler ve önerilen arama uzayı	41
Tablo 3. ÇESA modellerindeki katman sayıları	53
Tablo 4. ÇESA-1 modeli sistem mimarisi	55
Tablo 5. ÇESA-2 modeli sistem mimarisi	56
Tablo 6. ÇESA-3 modeli sistem mimarisi	57
Tablo 7. ÇESA-4 modeli sistem mimarisi	58
Tablo 8. MNIST veri setinde 50 çevrim sayısı ile yapılan istatistiksel analiz	65
Tablo 9. MNIST veri setinde 100 çevrim sayısı ile yapılan istatistiksel analiz	66
Tablo 10. CIFAR-10 veri seti test sonuçları	68
Tablo 11. CIFAR-100 veri seti test sonuçları	74
Tablo 12. FRUITTS-360 veri seti test sonuçları	80
Tablo 13. Yapılan deneylerin genel değeriendirilmesi	85

SEMBOLLER VE KISALTMALAR DİZİNİ

AMMY	: Ağırlıklandırılmış Minimum-Maksimum Yöntemi
AF	: Aktivasyon Fonksiyonu
ATY	: Ağırlıklı Toplam Yöntemi
BÖGTY	: Büyük Ölçekli Görsel Tanıma Yarışması
CIFAR	: Canadian Institute for Advanced Research (Kanada İleri Araştırmalar Enstitüsü)
ÇAO	: Çok Amaçlı Optimizasyon
ÇAOA	: Çok Amaçlı Optimizasyon Algoritması
DBN	: Deep Belief Network
DÖ	: Derin Öğrenme
EB	: Evrişim Bloğu
EK	: Evrişim Katmanı
EKY	: Epsilon Kısıt Yöntemi
ESA	: Evrişimli Sinir Ağları (CNN - Convolutional Neural Network)
GAN	: Generative Adversarial Networks
GİB	: Grafik İşlemci Birimi
HK	: Havuzlama Katmanı
HPO	: Hiper-Parametre Optimizasyonu
HUY	: Hedefe Ulaşma Yöntemi
İHA	: İnsansız Hava Aracı
KM	: Kalabalık Mesafesi
LSTM	: Long Short Term Memory
MLPNN	: Multilayer Perceptron Neural Network
MNIST	: Modified National Institute of Standards and Technology (Düzenlenmiş Ulusal Standartlar ve Teknoloji Enstitüsü)
MOEAs/D	: Multi-Objective Evolutionary Algorithms based on Decomposition (Ayrıştırmaya Dayalı Çok Amaçlı Evrimsel Algoritmalar)
MÖ	: Makine Öğrenmesi
MSA	: Meta-Sezgisel Algoritma
MSO	: Meta-Sezgisel Optimizasyon

NSGA-II	: Non-dominated Sorting Genetic Algorithm-II (Baskın Olmayan Sıralı Genetik Algoritma-II)
NSGAs	: Non-dominated Sorting Genetic Algorithms (Baskın Olmayan Sıralı Genetik Algoritmalar)
ÖÖB	: Özellik Öğrenme Bloğu
PSO	: Parçacık Sürü Optimizasyonu
RBM	: Restricted Boltzmann Machine
RNN	: Recurrent Neural Network
SB	: Sınıflandırma Bloğu
SMA	: Sinirsel Mimari Arama
SPEAs	: Strength Pareto Evolutionary Algorithms (Güçlü Pareto Evrimsel Algoritmalar)
TBK	: Tam Bağlı Katman
UÇD	: Uygunluk-Çeşitlilik Dengesi
YAR	: Yapay Açıklıklı Radar
YSA	: Yapay Sinir Ağları

1. GENEL BİLGİLER

Günümüzde yapay zeka (YZ), derin öğrenme (DÖ) teknikleri sayesinde birçok sektörde önemli yeniliklere imza atmaktadır. DÖ tekniklerinden özellikle Evrişimli Sinir Ağları (ESA), görüntü işleme (Shin vd., 2016), ses tanıma (Bahmei vd., 2022) ve video analizi (Li vd., 2022) gibi alanlarda üstün performans göstermektedir. ESA farklı katmanlarındaki filtreleri sayesinde verileri ayrıntılı bir şekilde analiz ederek, önemli özellikleri ortaya çıkartmaktadır. Modellerin doğruluğu ve etkinliği, kullanım alanlarında büyük bir öneme sahip olup farklı uygulamalarda başarının temelini oluşturmaktadır (Li vd., 2021).

ESA modellerinin başarısı, büyük ölçüde model tasarımının ve hiper-parametrelerin dikkatlice ayarlanmasına dayanmaktadır. Öğrenme oranı, yığın boyutu, çevrim sayısı, katman sayısı ve her katmandaki nöron sayısı gibi hiper-parametreler, modelin performansını doğrudan etkilemektedir (Zhang vd., 2024). Yanlış parametre ayarları, düşük performans veya aşırı öğrenme gibi sorunlara neden olabilmektedir (Qian vd., 2020). Bu sebeple ESA'nın performansını en üst düzeye çıkarmak için uygun ve etkili optimizasyon stratejilerinin kullanılması kritik öneme sahiptir (Ankalaki ve Thippeswamy, 2024).

Optimum hiper-parametrelerin ve model yapısının belirlenmesi, genellikle deneme-yanılma yöntemine dayandığı için yüksek hesaplama gücü ve zaman gerekmektedir. Arama uzayının genişliği nedeniyle optimizasyon problemi daha da karmaşık hale gelebilmektedir (Liu vd., 2022). Özellikle gerçek zamanlı uygulamalar ve düşük kapasiteli cihazlar için kullanılacak modellerde, zaman ve başarı dengesini sağlamak gerekmektedir. Bu nedenle efektif model optimizasyonu yapacak optimizasyon teknikleri önemlidir. (Zhang vd., 2021).

Model mimarisinin ve hiper-parametrelerin doğru seçimi modelin performansını önemli ölçüde artırmaktadır. İyi optimize edilmiş ESA modelleri, tıbbi görüntüleme, otonom sürüş, güvenlik sistemleri gibi kritik alanlarda kullanılmaktadır. Bu tür uygulamalarda yüksek doğruluk ve güvenilirlik esastır (Varughese ve Sridevi, 2024). Ayrıca uygun model yapıları ve hiper-parametreler, hesaplama kaynaklarının daha etkin kullanılmasına imkan tanımaktadır. Bu durum özellikle büyük veri setleri ve karmaşık modeller üzerinde çalışıldığında avantaj sağlamaktadır (Xin vd., 2020).

Bu tez çalışmasında, ESA'nın model tasarımını ve hiper-parametre optimizasyonunu (HPO), çok amaçlı optimizasyon (ÇAO) teknikleriyle gerçekleştiren yenilikçi bir yöntem

olan ÇESA tanıtılmaktadır. ÇESA birden fazla amacı eş zamanlı optimize ederek model doğruluğu ile yanıt süresi arasında denge kurmayı amaçlamaktadır. Derin öğrenme modellerinin hem doğruluk oranlarını yükselterek hem de yanıt sürelerini kısaltarak modelleri daha hızlı ve verimli hale getirmeye çalışmaktadır. Bu sayede model performansı iyileştirilirken aynı zamanda geniş bir uygulama alanında kullanılabilirliği artırılmış olacaktır. Hem yüksek doğruluk hem de düşük yanıt süreleri ile önerilen modeller, mevcut yaklaşımlara göre daha hızlı ve güvenilir sonuçlar sunmayı hedeflemektedir.

1.1. Yapay Zeka

YZ, insan beyninin işleyişini taklit ederek, öğrenme, düşünme, problem çözme ve dil anlama gibi görevleri yapabilen sistemler ve algoritmalar geliştiren bir bilgisayar bilimi alanıdır (Hurbans, 2020). YZ teknolojisi makinelerin ve yazılımların belirli görevleri kendi başlarına, insan müdahalesi olmadan gerçekleştirebilmesini sağlar. Bu teknolojinin amacı makinelerle insan gibi düşünme ve karar verme yeteneği kazandırmaktır. Böylece karmaşık problemler daha etkili ve kısa sürede çözülebilecektir (Eremia vd., 2020).

Alan Turing “Bilgisayar Makineleri ve Zeka” adlı makalesinde (1950), bir makinenin zeki olup olmadığını belirlemek için Turing Testi’ni önermiştir (Turing, 2009). Dartmouth Konferansı’nda, John McCarthy ve diğerleri (1956) tarafından “yapay zeka” terimini ilk kez kullanılmışlardır. Bu konferans YZ araştırmalarının başlangıç noktası olarak kabul edilmektedir (Rajaraman, 2014). YZ’nin gelişimi birkaç dalgalanma dönemi geçirmiştir. İlk dönemlerde, mantık temelli sistemler ve sembolik yöntemler üzerinde yoğunlaşmıştır. Ancak hesaplama gücünün sınırlamaları ve karmaşık problemlerin çözümündeki zorluklar, YZ araştırmalarının yavaşlamasına neden olmuştur (Gurjar vd., 2024). 1980’lerde uzman sistemlerin popülerlik kazanmasıyla YZ, ticari uygulamalarda kullanılmaya başlandı. Uzman sistemlerin yetersiz kalması 1990’lara gelinmeden finansman kesintisine sebep olmuştur. 2000’li yıllarda ise büyük veri, artan hesaplama gücü ve derin öğrenme gibi alanlardaki ilerlemeler YZ’nin yeniden popüler hale gelmesini sağlamıştır (Nazar vd., 2021).

YZ günümüzde birçok alanda kullanılmaktadır. Bazıları şunlardır:

- Sağlık: YZ hastalık teşhisi, ilaç keşfi ve hasta bakımında kullanılmaktadır. Örneğin, radyoloji görüntülerini analiz ederek erken teşhis koyabilen sistemler geliştirilmiştir (Wang ve Summers, 2012).

- Finans: YZ finansal dolandırıcılık tespiti, algoritmik ticaret ve müşteri hizmetlerinde kullanılmaktadır (Vetrivel vd., 2022). Bankalar kredi riskini değerlendirmek ve müşteri davranışlarını analiz etmek için YZ tercih etmektedir (Moradi ve Mokhatab, 2019).
- Otonom Araçlar: Otonom araçlar çevrelerini algılayarak güvenli bir şekilde hareket edebilmek için YZ teknolojilerini kullanmaktadır. Bu araçlar sürücüsüz taşıma sistemlerinde devrim yaratmaktadır (Bathla vd., 2022; Khayyam vd., 2020).
- Perakende: YZ müşteri davranışlarını analiz ederek kişiselleştirilmiş öneriler sunmakta ve envanter yönetimini optimize etmektedir (Babatunde vd., 2024). Çevrimiçi perakendeciler, müşteri deneyimini iyileştirmek için YZ tabanlı sohbet robotları kullanmaktadır (Chen vd., 2021).
- Eğitim: YZ öğrenci performanslarını izlemek ve kişiselleştirilmiş eğitim materyalleri sunmak için kullanılmaktadır. Akıllı öğretim sistemleri öğrencilerin öğrenme süreçlerini optimize etmektedir (Murtaza vd., 2022).
- Tarım: YZ mahsul verimini artırmak, zararlı canlıları tespit etmek ve tarımsal faaliyetleri otomatikleştirmek için kullanılmaktadır. Akıllı tarım sistemlerinde kaynak kullanımını optimize ederek sürdürülebilir tarım uygulamalarını desteklemektedir (AlZubi ve Galyna, 2023).

Yapay Zeka (YZ) teknolojisi, yalnızca insan becerilerini taklit etmekle kalmayıp, karmaşık problemlerin çözümünde benzersiz bir verimlilik sunarak modern dünyada dönüşüm sağlamaktadır. İlk ortaya çıkışından bu yana mantıksal ve sembolik yöntemlerden derin öğrenmeye kadar birçok farklı yaklaşımla gelişen YZ, grafik işlemci birimi (GİB) ve bulut sistemlerin sağladığı yüksek kapasiteli veri işleme yetenekleri sayesinde daha da ilerlemiştir. Bu gelişmeler, yeni algoritmik yaklaşımların ortaya çıkmasına zemin hazırlamış ve derin ağların çok daha büyük ölçekte çalışabilmesine imkan tanımıştır. DÖ sayesinde YZ problemleri yeniden modellenmiş ve daha iyi sonuçlar elde edilmiştir. Sağlık, finans, otonom araçlar, perakende, eğitim ve tarım gibi alanlarda çığır açan çözümler sunan YZ sadece mevcut sistemleri optimize etmekle kalmayıp, insan hayatını kolaylaştıran yeni inovasyonların da önünü açmıştır. YZ geleceğin teknolojik ekosistemini şekillendiren temel bir yapı taşı olarak sınırlarını her geçen gün genişletmektedir (Jarrahi, 2018).

1.2. Derin Öğrenme

Derin öğrenme (DÖ) büyük veri setlerinden anlamlı bilgiler çıkarma ve karmaşık sorunlara çözümler üretme amacıyla kullanılan etkili bir makine öğrenmesi (MÖ) yöntemidir (Mahmud vd., 2018). Son yıllarda veri miktarının artması ve hesaplama kapasitelerinin gelişmesiyle birlikte, derin öğrenme teknikleri hızla ilerlemiş ve geniş bir uygulama alanı bulmuştur (Najafabadi vd., 2015). DÖ’de, insan beyninin yapısını ve işlevini modellemeye çalışan yaklaşımlar kullanılmaktadır. Bu modeller çok katmanlı derin sinir ağlar aracılığıyla veriyi işlemektedir. Her katman önceki katmandan gelen bilgiyi daha soyut bir hale dönüştürmektedir. Bu katmanlı yapı sayesinde, derin öğrenme modelleri karmaşık veri setlerindeki gizli örüntüleri ve özellikleri keşfetme becerisine sahiptir (LeCun vd., 2015).

DÖ başarısının arkasındaki en önemli etkenlerden biri çok katmanlı derin sinir ağlarının kullanılmasıdır (Mao vd., 2018). Bu ağlar, bir giriş katmanı ile başlar, ardından birden fazla gizli katman ile devam eder ve bir çıkış katmanı ile tamamlanır. Her bir katman girdileri belirli bir fonksiyonla (genellikle doğrusal olmayan bir aktivasyon fonksiyonu) işleyerek karmaşık ilişkileri öğrenbilir (Cinar, 2020). DÖ modellerinde katman sayısının artması, modelin öğrenme kapasitesini ve karmaşık veri yapıları arasındaki ilişkileri anlama yeteneğini güçlendirmektedir (Ahmed vd., 2023; Sarker, 2021). Ancak katman sayısının artması, modelin hesaplama maliyetini ve eğitim süresini de önemli ölçüde artırmaktadır (Basodi vd., 2020; Lewis vd., 2021; Menghani, 2023).

DÖ optimizasyonu, modelin sahip olduğu büyük parametre sayısı ve karmaşık yapı yüzeyleri nedeniyle oldukça zordur (Abdolrasol vd., 2021). Gradyan inişi gibi optimizasyon algoritmaları yerel minimumlara takıldığında en iyi çözümü bulmakta güçlük çekebilmektedir (Wierichs vd., 2020). Bunun yanı sıra, öğrenme hızı ve momentum gibi hiper-parametrelerin doğru ayarlanması büyük önem taşımaktadır. Bu parametrelerin yanlış ayarlanması, modelin aşırı öğrenmesine veya yetersiz öğrenmesine yol açmaktadır (Bejani ve Ghatee, 2021). Büyük veri setlerinin işlenmesi yüksek hesaplama maliyetleri gerektirmektedir. Veri setlerindeki dengesizlikler optimizasyon sürecini daha da karmaşık hale getirir. Bu sebeple başarılı bir optimizasyon süreci için dikkatli bir planlama ve gelişmiş tekniklerin kullanılması gereklidir (Shlezinger vd., 2022; Zhang vd., 2022).

Derin öğrenme modelleri, farklı yapı ve işlevlere sahip çeşitli sinir ağlarından oluşur. Başlıca derin öğrenme sınıflandırması şunlardır:

- Çok katmanlı algılayıcı (multilayer perceptron neural network, MLPNN)
- Derin inanç ağları (deep belief networks, DBN)
- Evrişimsel sinir ağları (ESA - convolutional neural networks, CNN)
- Yinelemeli sinir ağları (recurrent neural network, RNN)
- Uzun kısa vadeli bellek (long short term memory, LSTM)
- Üretken çekişmeli ağlar (generative adversarial networks, GAN)
- Kısıtlanmış Boltzmann makinesi (restricted Boltzmann machine, RBM)

1.2.1. Yapay Sinir Ağları

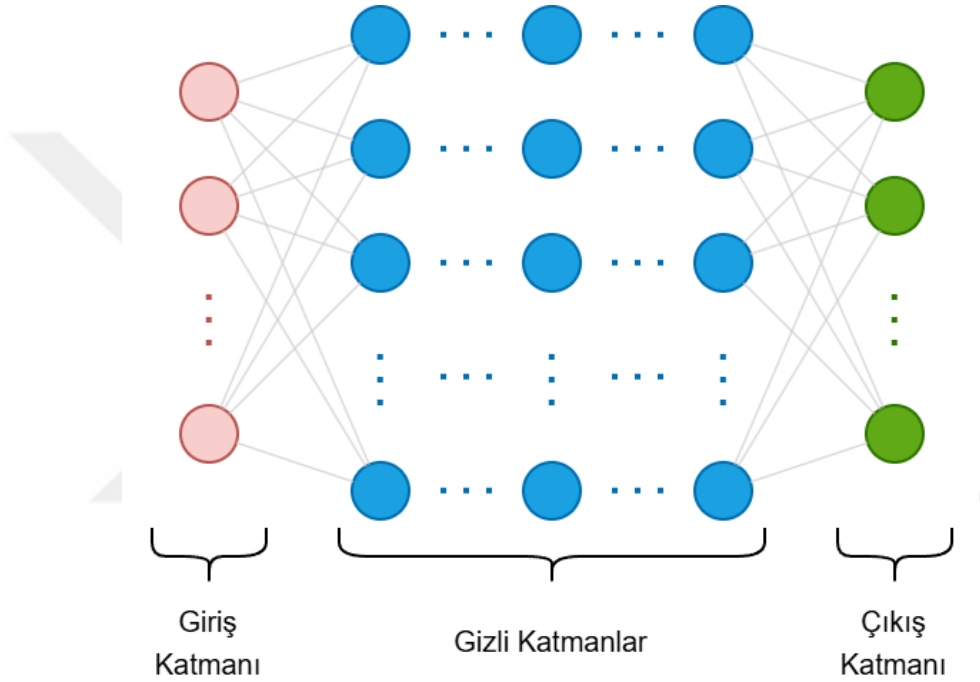
Yapay sinir ağları (YSA), insan beyninin biyolojik sinir sistemini temel alarak geliştirilen ve çeşitli karmaşık görevlerde yüksek başarı elde etmeyi amaçlayan matematiksel modellerdir. Bu ağlar, biyolojik nöronların birbirleriyle bağlantı kurarak bilgi işleme yeteneklerinden esinlenmiştir (Bacanin vd., 2020). YSA'lar çok sayıda nöronun katmanlar halinde düzenlenmesiyle oluşturulur. Bu nöronlar arasındaki bağlantıların ağırlıkları, öğrenme süreci boyunca güncellenerek ayarlanır. YSA'lar, veri analizi (Wang vd., 2024), desen tanıma (Wang vd., 2021), görüntü (Huixian, 2020) ve ses tanıma (Alam vd., 2020), metin işleme (Naiemi vd., 2020), sınıflandırma (Baldominos vd., 2020), tahmin (Tedjopurnomo vd., 2020) ve optimizasyon (Habib ve Qureshi, 2022) gibi çeşitli alanlarda yaygın olarak kullanılmaktadır. Bu geniş kullanım alanları YSA'ların esnek yapısı ve güçlü öğrenme yeteneklerinden kaynaklanmaktadır. YSA, karmaşık veri setlerindeki gizli ilişkileri ve kalıpları öğrenerek bu bilgileri yeni ve bilinmeyen verilere uyarlayabilme yeteneğine sahiptir (Samek vd., 2021).

YSA'nın kökeni 1940'lı yıllara kadar uzanmaktadır. Bu alandaki ilk adım, Warren McCulloch ve Walter Pitts tarafından geliştirilen basit bir sinir modeliyle atılmıştır (Makarenko, 2020). 1980'ler ve 1990'lar boyunca, geri yayılım algoritması gibi tekniklerin geliştirilmesiyle, sinir ağları alanında önemli ilerlemeler kaydedilmiştir (Liu vd., 2021). Son yıllarda ise, büyük veri kümelerinin çoğalması ve hesaplama kapasitesinin artması, derin öğrenme modellerinin popülerleşmesine ve yapay sinir ağlarına olan ilginin yeniden canlanmasına neden olmuştur (Adadi, 2021).

Şekil 1'de gösterilen YSA, genellikle üç ana bileşenden oluşur:

- Giriş Katmanı: Dış dünyadan alınan verilerin ağa girdiği ilk katmandır. Bu katman veriyi, ağırlıklandırma işlemi öncesinde uygun bir formatta sunmaktadır.

- **Gizli Katmanlar:** Ağın işlem gücünü belirleyen katmanlardır. Birden fazla gizli katman kullanılarak veri üzerinde daha karmaşık işlemler gerçekleştirilebilir. Her gizli katman, bir önceki katmandan aldığı veriyi belirli bir aktivasyon fonksiyonu aracılığıyla işleyerek sonraki katmana iletmektedir.
- **Çıkış Katmanı:** Ağın nihai sonucunu üreten son katmandır. Çıkış katmanının yapısı, modelin çözmeye çalıştığı problem türüne (örneğin, sınıflandırma veya regresyon) bağlı olarak değişiklik göstermektedir.



Şekil 1. Yapay sinir ağı genel ağ modeli

YSA'ların temel yapı taşları nöronlar olup bu nöronlar, giriş verilerini alır, bu verileri işler ve bir çıktı üretir. Her bir nöronun net çıkışı, tüm girişlerin ağırlıklarla çarpılıp toplanmasıyla hesaplanır.

$$z_j = \sum_{i=1}^n w_{ij} * x_i + b_j \quad (1)$$

Denklem 1'de her nöron kendisine gelen girdileri (x_i) ilgili ağırlıklarla (w_{ij}) çarparak toplar. Ağırlıklı toplam değerine nöronun bias (b_j) değeri de eklenerek nöronun net (z_j) değeri hesaplanır.

$$a_j = f(z_j) \quad (2)$$

Denklem 2’de net değeri bir aktivasyon fonksiyonu (f) aracılığıyla, bir çıktıya (a_j) türetilir. Aktivasyon fonksiyonları genellikle doğrusal olmayan işlevlerdir. En yaygın olanları arasında sigmoid, tanh ve ReLU bulunur.

$$\hat{y}_k = \text{softmax}(z_k) \quad (3)$$

Nöronlar arasındaki bağlantılar, bilgiyi bir katmandan diğerine taşımaktadır. Denklem 3’te sınıflandırma problemlerinde çıktı katmanında ağırlıklı toplam değerinden (z_k), çıktı ($\hat{y}_k$) değeri *softmax* fonksiyonu ile hesaplanır.

$$L(y, \hat{y}) = - \sum_{i=1}^n y_i \log \hat{y}_i \quad (4)$$

Nöronlar arasındaki her bağlantı bir ağırlıkla ilişkilendirilir. Bu ağırlıklar öğrenme sürecinde güncellenerek ağıın performansı artırılır. Giriş verisi ağıın katmanları boyunca ileriye doğru taşınır, her katmanda işlenerek çıkış katmanına ulaşır. Denklem 4’te çıkış katmanında üretilen sonuç ($\hat{y}$), hedef değerle (y) karşılaştırılır ve bir hata değeri ($L(y, \hat{y})$) hesaplanır.

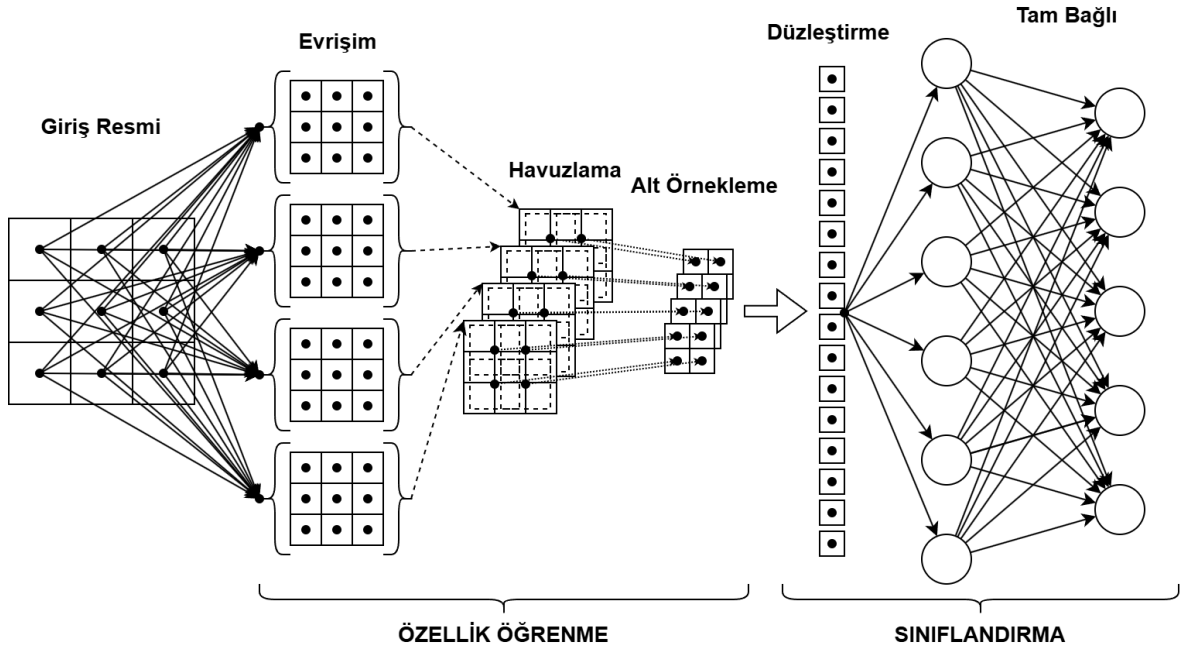
$$w_{ij} = w_{ij} - \mu * \frac{\delta L}{\delta w_{ij}} \quad (5)$$

Denklem 4’de hesaplanan hata değeri (L), ağıın ağırlıklarının (w_{ij}) güncellenmesi için geriye doğru yayılır. Geri yayılım algoritması, hatayı geriye doğru izleyerek her bir ağırlığın hataya katkısını hesaplar. Denklem 5’de hata fonksiyonu türevlenerek (gradyan hesaplaması), ağıın ağırlıkları geri yayılım algoritmasıyla güncellenir. Öğrenme oranı (μ) ağırlıkların güncellenme hızını kontrol etmektedir.

1.2.2. Evrişimli Sinir Ağları

Derin öğrenme genel olarak çok gelişmiş yapay sinir ağları olarak düşünebilir. Çalışmada evrişimli sinir ağları kullanılacağından çok geniş bir konu olan derin ağlardan diğerlerine değinilmemiştir.

Evrişimli sinir ağları (ESA) görüntü tanıma (Won, 2020), ses analizi (Yousefi ve Hansen, 2020), doğal dil işleme (Alnashwan vd., 2023) ve sinyal işleme (Yu vd., 2021) gibi geniş bir yelpazede uygulamalara sahip güçlü bir derin öğrenme modelidir. ESA özellikle matris yapısındaki veriler üzerinde etkili çalışmak üzere geliştirilmiştir. Bu tür veriler üzerinde gösterdiği üstün performansla öne çıkan başarılı modellerden biri olarak kabul edilir (Zhao vd., 2024). ESA en az bir katmanında filtreleme işlemi gerçekleştirir. Evrişim katmanının amacı girdinin özniteliklerini belirlemektir. Havuzlama katmanında ise öznitelikleri indirger (Chen vd., 2024).



Şekil 2. Evrişimli sinir ağı temel mimarisi

Şekil 2’de temel bir ESA mimarisi gösterilmiştir. Bu mimari giriş, evrişim, havuzlama ve tam bağlı katmanlardan oluşur. Girdiler genellikle sinyal (vektör) ya da görüntü (matris) formatındadır. ESA’da görüntüler genellikle genişlik, yükseklik ve renk kanallarını içeren üç boyutlu bir matris şeklinde modele giriş olarak verilir. Evrişim işlemi sonrasında elde

edilen doğrusal veri, bir transfer fonksiyonu ile doğrusal olmayan bir forma dönüştürülür. Havuzlama katmanı, evrişim katmanının ardından gelerek verinin boyutunu küçültür. Evrişim katmanları görüntüden özellikler çıkarırken, havuzlama katmanları bu özelliklerin boyutunu azaltarak modelin karmaşıklığını ve hesaplama yükünü düşürmektedir. Sonraki aşamada düzleştirme katmanı, çok boyutlu özellikleri tek bir vektöre indirger. Vektörü girdi olarak alan tam bağlı katmanlar modelin hesaplama işlemini tamamlar. Modelin performansı değerlendirilirken hata değerleri hesaplanır ve bu hatalara göre ağırlıklar güncellenir (Cong ve Zhou, 2024).

Denklem 6’da evrişim katmanında kullanılan filtereme işlemi verilmektedir. Burada I ; giriş görüntüsünü, K , filtreyi, i, j , görüntüdeki piksel konumlarını, m, n , filtre boyutlarını ifade etmektedir.

$$(I * K)(i, j) = \sum_m \sum_n I(i - m, j - n) \cdot K(m, n) \quad (6)$$

Denklem 7’de sırasıyla maksimum ve ortalama havuzlama yöntemi verilmektedir. X ; giriş özellik haritasını, Y , çıkış özellik haritasını, R , alt örneği, i, j , alt örnek pozisyonu ifade etmektedir.

$$Y(i, j) = \max\{X(m, n) | (m, n) \in R(i, j)\}$$

$$Y(i, j) = \frac{1}{|R(i, j)|} \sum_{(m, n) \in R(i, j)} X(m, n) \quad (7)$$

ESA’lar, geleneksel sinir ağlarının geri yayılım ve kayıp fonksiyonları gibi bazı genel özelliklerini paylaşmakla birlikte mimari olarak oldukça farklılık göstermektedir. Geleneksel sinir ağlarında giriş ile çıkış katmanı arasında, her nöronun önceki katmandaki tüm nöronlara bağlı olduğu tam bağlı gizli katmanlar bulunmaktadır (Doi vd., 2019). ESA ise yerel alıcı alan, ağırlık paylaşımı ve alt örnekleme olmak üzere üç temel özelliğe sahiptir:

- **Yerel Alıcı Alanlar:** ESA’da her nöron, girdi görüntüsünün yalnızca “yerel alıcı alan” denilen küçük bir bölgesiyle bağlantılıdır. Bu strateji her nöronun sadece bağlantılı olduğu bu küçük alanın özelliklerini öğrenmesini sağlayarak karmaşık desenlerin algılanmasını kolaylaştırır (Jiang vd., 2024).
- **Ağırlık Paylaşımı:** ESA’da aynı evrişim katmanındaki nöronlar, aynı ağırlık setini (filtre veya çekirdek) paylaşır. Bu yaklaşım modelin daha az parametreye sahip

olmasını sağlayarak işlem hızını artırır. Ağırlık paylaşımı, girdinin farklı bölgelerinde aynı özellikleri tanıyabilme yeteneğini geliştirir (Goay vd., 2021).

- Alt Örnekleme: ESA’da havuzlama katmanları genellikle evrişim katmanlarının ardından gelerek özellik haritalarının boyutunu küçültür. Havuzlama işlemi genellikle maksimum veya ortalama değerler alınarak gerçekleştirilir. Bu katman, modelin hesaplama yükünü azaltmakla kalmaz, aynı zamanda uzamsal değişikliklere karşı dayanıklılık sağlar (Naulia vd., 2024).

1.3. Hiper-Parametre Optimizasyonu

ESA ağlarında hiper-parametre optimizasyonu (HPO), modelin performansını artırmak amacıyla ESA’nın eğitim ve yapılandırma ayarlarının en uygun kombinasyonlarını belirlemeye yönelik bir süreçtir (Atteia vd., 2022). Hiper-parametreler modelin mimarisi ve eğitim sürecini kontrol eden, ancak eğitim sırasında öğrenilemeyen ayarlardır (Raziani ve Azimbagirad, 2022). HPO’nun amacı bu ayarları en iyi şekilde belirleyerek modelin genelleme kapasitesini maksimize etmektir (Mostafa vd., 2020). Tablo 1’de, her bir ESA katmanı için ayarlanabilen hiper-parametreler ayrıntılı olarak sunulmaktadır.

Tablo 1. Evrişimli sinir ağlarında kullanılan hiper-parametreler

Evrişim Katmanı	Havuzlama Katmanı	Tam Bağlı Katman
Katman Sayısı	Katman Sayısı	Katman Sayısı
Filtre Sayısı	Filtre Boyutu	Nöron Sayısı
Filtre Boyutu	Filtre Adımı	
Filtre Adımı	Dolgu Oranı	
Dolgu Oranı	Havuzlama Yöntemi	
Eğitim Süreci	Düzenleme	Ağırlıklar
Eğitim Döngü Sayısı	Budama Oranı	Başlangıç Ağırlıkları
Yığın Boyutu	Düzenleştirme Yöntemi	Normalizasyon Ağırlıkları
Öğrenme Oranı	L_1 - L_2 Katsayıları	Bias Başlangıç Değeri
Aktivasyon Fonksiyonu		
Optimizasyon Yöntemi		

Evrişim katmanında, katman sayısının ve filtrelerin sayısının artırılması, modelin daha karmaşık ve detaylı özellikleri öğrenmesini sağlar. Ancak çok sayıda katman eklemek

hesaplama maliyetini artırabilir ve modeli aşırı uyuma yatkın hale getirebilir. Filtre boyutları bu süreçte önemli bir rol oynar. Küçük boyutlu filtreler (örneğin 2×2) daha ince ve ayrıntılı özelliklerin yakalanmasına olanak tanırken, büyük boyutlu filtreler (örneğin 5×5) daha geniş ve genel özellikleri çıkarır (Mohakud ve Dash, 2022). Filtre adım boyutunun artırılması, özellik haritasının boyutunu küçülterek hesaplama maliyetini düşürse de bilgi kaybına yol açabilir. Dolgu türü çıktı boyutunu etkiler, “aynı” dolgu çıktı boyutunu girişle aynı tutarken, “geçerli” dolgu kullanımı çıktının boyutunu küçültür (Naseri ve Mehrdad). Her katmandan sonra özellik haritasının boyutu, katmanın derinliğine ve filtre sayısına bağlı olarak değişir. Daha derin katmanlarda genellikle daha fazla sayıda filtre kullanılır, bu da modelin hem öğrenme kapasitesini hem de genelleme yeteneğini artırmaya yönelik bir stratejidir (Raiaan vd., 2024).

Havuzlama katmanı, modelin boyutunu küçülterek hesaplama maliyetini düşürmeyi ve modelin daha hızlı çalışmasını sağlamayı amaçlar. Bu katman özellik haritasının boyutunu azaltarak modelin işlem süresini hızlandırır. Havuzlama katmanları modelin konum değişikliklerine karşı daha dayanıklı olmasına katkıda bulunur. Bu durum modelin genelleme yeteneğini güçlendirir (İnik, 2023). Havuzlama katmanlarında kullanılan filtre boyutu, özellik haritasının boyutunu küçültmede önemli bir faktördür. Çoğunlukla 2×2 boyutunda filtreler tercih edilir ve bu da özellik haritasının boyutunu yarıya indirir. Filtre adımı havuzlamanın yatay ve dikey eksenlerde ne kadar ilerleyeceğini belirler. Örneğin, adım boyutu 2 olarak ayarlandığında, filtre her adımda 2 piksel hareket eder ve bu durum hem yatay hem de dikey yönde özellik haritasının boyutunu küçültür (Andonie ve Florea, 2020). Havuzlama yöntemi olarak en sık kullanılan maksimum ve ortalama havuzlamadır. Maksimum havuzlama, en yüksek değerleri seçerek daha belirgin ve keskin özellikleri öne çıkarırken, ortalama havuzlama, değerlerin ortalamasını alarak daha yumuşak ve genel bir temsil sunmaktadır (Loussaief ve Abdelkrim, 2018).

Tam bağlı katmanların sayısı ve her bir katmandaki nöron sayısı, bir ESA modelinin performansını ve genelleme yeteneğini önemli ölçüde etkiler (Zatarain vd., 2020). Daha fazla katman eklemek, modelin öğrenme kapasitesini artırarak daha karmaşık ilişkileri öğrenmesini sağlar. Bu durum özellikle büyük ve karmaşık veri setleri üzerinde çalışırken faydalıdır. Ancak aşırı sayıda katman kullanmak modelin aşırı öğrenme riskini de beraberinde getirir. Bu durumda model eğitim verisine mükemmel uyum sağlayabilir, ancak test verisi üzerinde düşük performans sergileyebilir (Kilichev ve Kim, 2023).

Eğitim sürecinde, yığın boyutu, modelin her bir eğitim adımında işlediği veri örneklerinin sayısını belirtir. Küçük yığın boyutları daha sık model güncellemeleri sağlar, ancak bu durum gradyan varyansını artırabilir. Büyük yığın boyutları ise daha doğru model güncellemeleri yapar, ancak hesaplama maliyetini yükseltir (Talaat ve Gamel, 2023). Yüksek öğrenme oranları, modeli hızlı bir şekilde yakınsamaya yönlendirebilir, ancak bu durum modelin yerel minimum tuzaklarına takılma riskini de artırır. Öte yandan, düşük öğrenme oranları daha stabil bir öğrenme süreci sunarken, yakınsama hızını yavaşlatabilir (Florea ve Andonie, 2020).

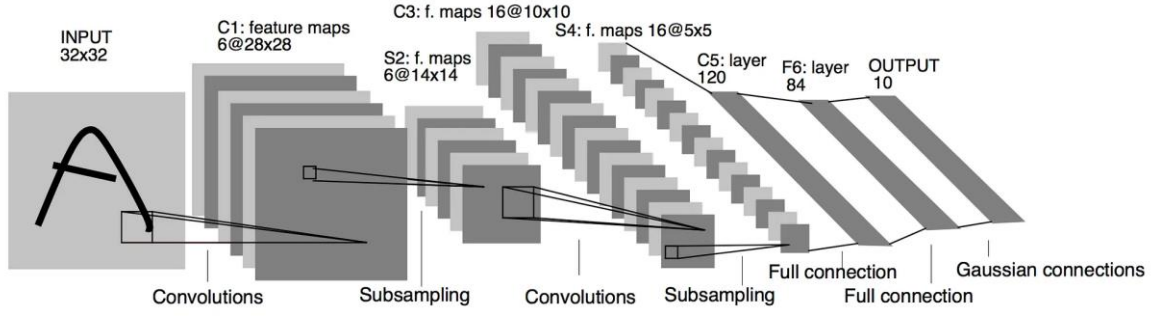
Düzenleme sürecinde budama oranı, aşırı uyumu önlemek amacıyla kullanılır ve genellikle %20 ile %50 arasında bir değer seçilir. L_1 ve L_2 düzenleştirme yöntemleri de aşırı uyumu önlemek için kullanılır. L_2 düzenleştirme yöntemi daha yaygın olarak tercih edilir (Hong ve Chen, 2023). ESA'da, öğrenme hızını etkileyen faktörler arasında başlangıç ağırlıkları, bias başlangıç değerleri ve normalizasyon ağırlıkları yer alır, bu değerler genellikle rastgele olarak başlatılır (Choudhary vd., 2024).

1.4. Yaygın ESA Mimarileri

1980'lerin sonlarından itibaren teorik temelleri atılan ve 1990'larda ilk pratik uygulamaları geliştirilen ESA, büyük veri setlerinin ve güçlü hesaplama kaynaklarının artışıyla birlikte 2010'lu yılların başında modern derin öğrenme araştırmalarının merkezine oturmuştur. ESA, görüntüdeki öznelilikleri otomatik olarak öğrenme kapasitesi sayesinde el yazısı tanımadan otonom sürüş sistemlerine kadar geniş bir yelpazede kullanılmıştır. Bu modeller evrim, havuzlama, doğrusal olmayan aktivasyonlar ve tam bağlı katmanlar gibi yapılarla öne çıkmaktadır. Her bir yeni model, hem teorik yenilikler hem de pratik uygulamalar sunarak görüntü işleme görevlerinde daha önce elde edilmemiş doğruluk seviyelerine ulaşmayı başarmıştır.

1.4.1. İlk Derin Öğrenme: LeNet (1998)

LeNet, Yann LeCun ve arkadaşları tarafından 1998 yılında geliştirilmiş, ilk derin öğrenme tabanlı görüntü tanıma modellerinden biridir. Modelin amacı, el yazısı rakam tanıma ve posta kodlarını otomatik olarak okumak gibi görevlerdi. LeNet bu bağlamda, MNIST gibi veri setleri üzerinde oldukça başarılı olmuş ve yapay sinir ağlarının potansiyelini göstermiştir (LeCun vd., 1998).



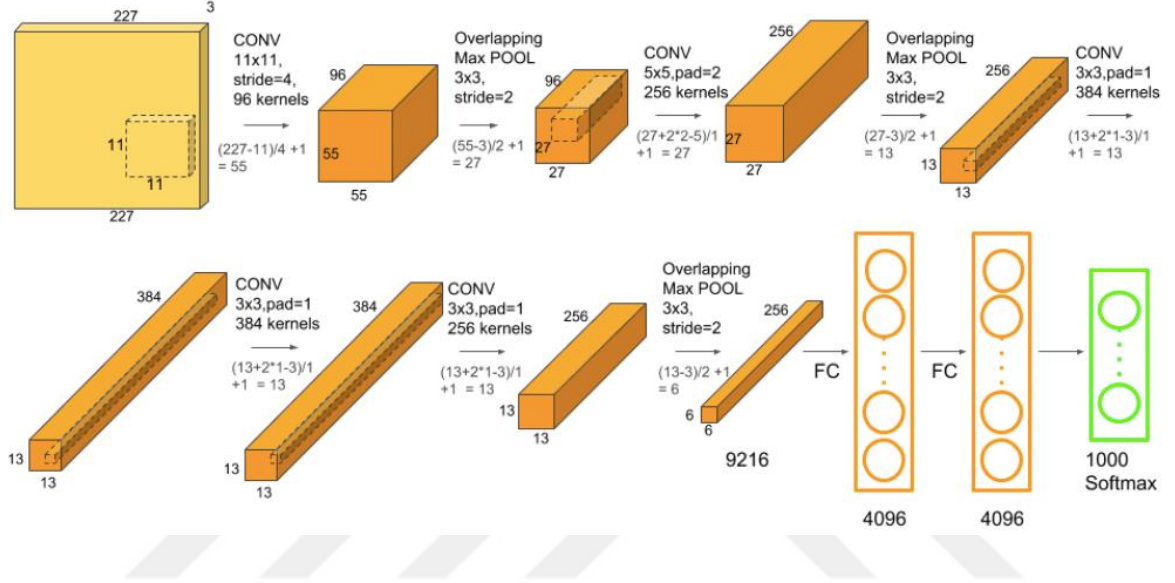
Şekil 3. LeNet-5 derin ağ modeli (LeCun vd., 1998)

Şekil 3'te gösterilen LeNet'in en popüler versiyonu olan LeNet-5 derin ağ modeli verilmiştir. LeNet-5 derin ağ modeli, giriş katmanını, üç evrişim katmanını (C_1 , C_3 ve C_5), iki havuzlama katmanını (S_2 ve S_4), iki tam bağlı katman (F_5 ve F_6) ve bir çıkış katmanından oluşmaktadır. C_1 her biri 5×5 boyutunda 6 filtrelili bir evrişim katmanıdır. Giriş görüntüsünden farklı öznitelikler çıkarmaktadır. S_2 ortalama havuzlama katmanı, her 2×2 bölgeyi alarak çıktı boyutunu yarıya indirmektedir. İkinci evrişim (C_3) ve havuzlama (S_4) katmanları, daha derin öznitelik çıkarımını sağlamaktadır. Tam bağlı katmanlar sınıflandırma için özellikleri birleştirmektedir. LeNet modeli aktivasyon fonksiyonu olarak *sigmoid* veya *tanh* kullanarak veri temsillerini doğrusal olmayan bir hale getirmektedir. Ayrıca ESA mimarisinin temel yapı taşlarını tanıtarak, daha sonraki modellerin gelişimine katkı sağlamıştır. C_5 katmanı, 5×5 boyutundaki filtreler ile çalışır ve 120 özellik haritası üretir. Bu katman önceki evrişim ve havuzlama katmanlarından gelen öznitelikleri daha yüksek seviyede temsil eder ve son sınıflandırma katmanlarına bilgi sağlar. Bu katmandaki nöronların her biri, S_4 'teki özellik haritalarına bağlıdır. C_5 katmanı modelin karmaşık özellikleri öğrenebilmesini sağlar (LeCun vd., 1998; Wei vd., 2019).

Geleneksel sinir ağlara kıyasla LeNet-5 evrişim filtre paylaşımı sayesinde parametre sayısı oldukça azdır. Bu sayede daha az bellek kullanımı ve daha az işlem gücü gerektirir. Görüntüdeki düşük seviyeli özelliklerden başlayarak daha yüksek seviyeli özellikleri öğrenmesi, görüntü işleme problemlerine uyguladığını göstermektedir. Kenar, köşe gibi düşük seviyeli özellikler ilk evrişim katmanlarında öğrenilirken, daha soyut özellikler ilerleyen katmanlarda temsil edilir. LeNet-5, evrensel özellik öğrenimi sağlayan bir mimari sunduğu için, yeni veri setlerine uyarlanması görece kolaydır ve farklı sınıflandırma problemlerine transfer edilebilir. LeNet-5'in başarılı performansı, görüntü işleme ve tanıma alanlarında derin öğrenme algoritmalarının kullanılabilirliğini göstermiştir (Wei vd., 2024).

1.4.2. AlexNet (2012): Derin Öğrenmenin Yükselişi

AlexNet, 2012 yılında Alex Krizhevsky ve ekibi tarafından geliştirilmiştir. Bu model Grafik İşlemci Birim'lerinin (GİB) yoğun kullanımını içeren ilk büyük ölçekli sinir ağı modelidir. AlexNet, derin öğrenme araştırmalarında bir dönüm noktası olarak kabul edilir (Krizhevsky vd., 2012).



Şekil 4. AlexNet derin ağ modeli (Krizhevsky vd., 2012)

Şekil 4'te gösterilen AlexNet derin ağ modeli, 5 evrişim ve 3 tam bağlı katman olmak üzere 8 katmandan oluşmaktadır. İlk katmanda 96 adet 11×11 filtre kullanılarak kapsamlı bir filtreleme uygulamaktadır. *ReLU* aktivasyon fonksiyonu kullanıldıktan sonra 3×3 boyutunda maksimum havuzlama işlemi gelmektedir. Ara katmanlarda daha küçük (3×3 ve 5×5) filtrelerle derin özelliklerin çıkarımı yapılmaktadır. Bu katmanlar, kenar ve doku gibi daha soyut özellikleri tanımaktadır. Tam bağlı katmanlar özellikleri sınıflandırmak için kullanılmakta ve *softmax* çıkışı ile sonlanmaktadır (Krizhevsky vd., 2012). AlexNet, *ReLU* aktivasyon fonksiyonunu tanıtarak, derin ağların daha hızlı eğitilmesini sağlamıştır. Ayrıca veri artırma ve budama teknikleri ile aşırı uyum sorunlarını azaltarak performansı artırmıştır (Ullah vd., 2022).

AlexNet, *tanh* veya *sigmoid* gibi geleneksel aktivasyon fonksiyonları yerine *ReLU*'yu kullanarak eğitim hızını önemli ölçüde artırmıştır. *ReLU* öğrenmeyi hızlandırırken, gradyan sönümlenmesi problemini de azaltır. Ayrıca AlexNet'te, tam bağlı katmanlarda aşırı uyum problemini azaltmak için budama tekniğini kullanılmıştır. Budama tekniği her öğrenme

çevriminde bazı nöronları rastgele devre dışı bırakarak modelin genelleme kabiliyetini artırmaktadır. Bu teknik, modelin öğrenme kapasitesini artırır ve performansını geliştirir. Özellikle ilk katmanlarda özelliklerin daha keskin bir şekilde ayırt edilmesine katkı sağlar. AlexNet'in 2012 yılında ImageNet LSVRC-2012 yarışmasında gösterdiği başarı, derin öğrenme ve ESA'nın yüksek boyutlu görsel veriler üzerindeki performansını kanıtlamış ve bu alanın büyük bir ivme kazanmasını sağlamıştır (Jana vd., 2024).

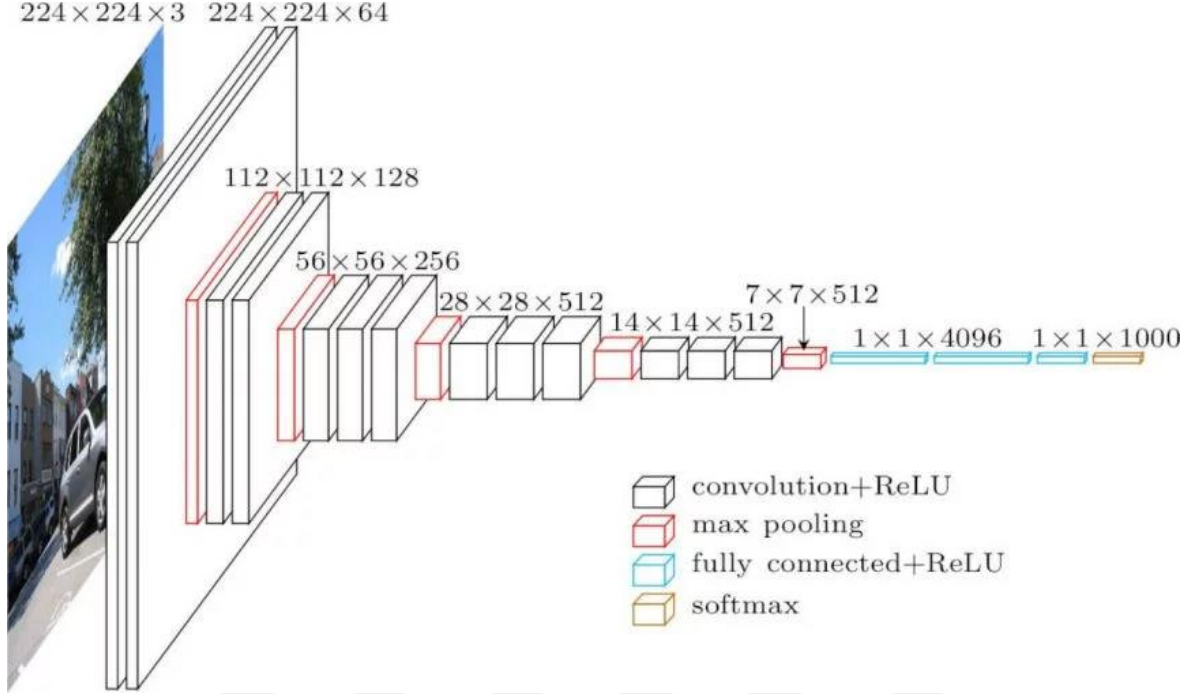
1.4.3. VGGNet (2014): Daha Derin Daha Basit

VGGNet, Oxford Üniversitesi'ndeki Visual Geometry Group tarafından geliştirilmiş ve 2014 ImageNet LSVRC-2014 yarışmasında en iyi performanslardan birini sergilemiştir. Model daha derin ağ yapılarının küçük boyutlu evrişim filtreleriyle (3x3) daha iyi performans gösterebileceğini kanıtlamıştır (Shinde vd., 2021).

VGG-16 ve VGG-19 olmak üzere iki temel VGGNet versiyonu bulunmaktadır. Her iki model de 3x3 evrişim filtresi ve 2x2 maksimum havuzlama işlemi kullanarak derin yapılar oluşturmaktadır. Katman yapısı giriş katmanı, bir dizi evrişim ve maksimum havuzlama katmanları, ardından tam bağlı katmanlar ve çıkış katmanı şeklindedir. VGG-16, 13 evrişim ve 3 tam bağlı katman, VGG-19 ise 16 evrişim ve 3 tam bağlı katman içermektedir. VGGNet fazla sayıda ve küçük boyutlu filtreler kullanarak model derinliğini artırmış ve yüksek performans elde etmiştir. Bu yaklaşım derin öğrenme çalışmalarında geniş çapta benimsenerek birçok modelin temelini oluşturmuştur (Feng vd., 2023).

Şekil 5'te VGGNet versiyonlarında olan VGG-16 derin ağ modeli gösterilmektedir. Birinci evrişim katmanına giren veri 224x224 boyutlarında renkli bir görüntüdür. Görüntü, 3x3 boyutunda çok küçük bir alana sahip filtreler kullanılarak bir dizi evrişim katmanından geçirilir. Bu boyut, sol/sağ, yukarı/aşağı ve merkez gibi yön kavramlarını yakalayabilecek en küçük boyuttur. Bazı konfigürasyonlarda 1x1 boyutunda evrişim filtreleri de kullanılır. Bu, giriş kanallarının doğrusal bir dönüşümü olarak düşünülebilir. Evrişim kaydırması bir piksel olarak sabitlenmiştir. Evrişim katmanlarının girişine yapılan dolgu, evrişim sonrasında çözünürlüğün korunmasını sağlar. Yani 3x3 evrişim katmanları için dolgu 1 piksel olarak ayarlanmıştır. "Uzaysal havuzlama" bazı evrişim katmanlarının ardından gelen beş adet maksimum havuzlama katmanı ile gerçekleştirilir (tüm evrişim katmanlarından sonra maksimum havuzlama yapılmaz). Maksimum havuzlama, 2x2 piksel filtre boyutu üzerinde 2 piksel kaydırma ile uygulanır. Evrişim ve havuzlama katmanlarından oluşan bu

yapının ardından (farklı mimarilerde derinlik değişkenlik gösterir), üç tane tam bağlı katman gelir. Son katman *softmax* çıktı katmanıdır (Tang vd., 2024).

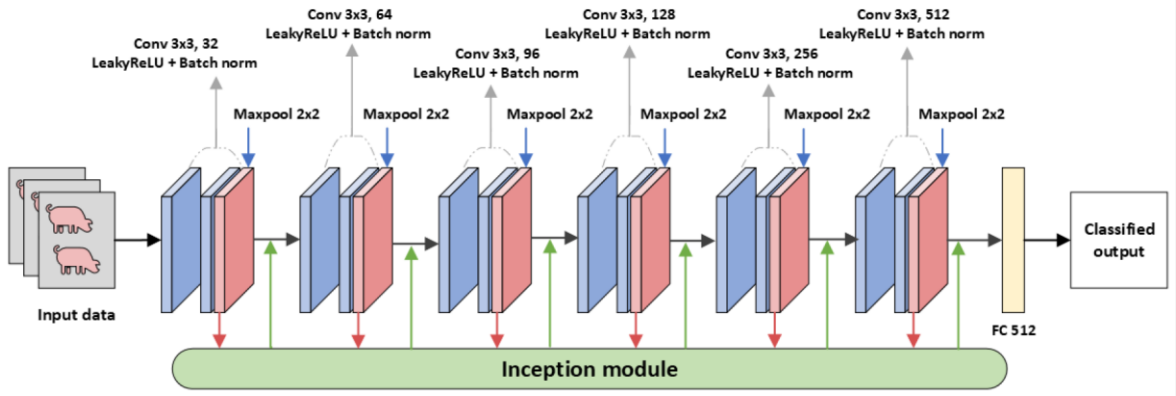


Şekil 5. VGG-16 derin ağ modeli (Kamilaris ve Prenafeta-Boldú, 2018)

VGGNet'in başarısı derin öğrenme topluluğunda, derin ve küçük boyutlu filtreli ağların etkinliğini ortaya koymuştur. Özellikle küçük boyutlu filtreler kullanarak model derinliğinin artırılmasının performansı iyileştirdiğini göstererek, diğer derin ağ mimarilerine ilham vermiştir. Pek çok derin öğrenme mimarisi, VGGNet'in yapı taşlarını (örneğin, 3x3 filtreli evrişim katmanları ve maksimum havuzlama) temel alarak gelişmiştir. VGGNet transfer öğrenme için önceden eğitilmiş modeller arasında popüler bir seçenek haline gelmiştir. Özellikle daha büyük veri kümelerinde eğitilmiş olması, sınırlı veri kümeleri üzerinde çalışan araştırmacılar için hazır bir başlangıç noktası sunmaktadır. VGGNet'in yüksek hafıza ve hesaplama gereksinimleri, daha sonra gelen ResNet ve MobileNet gibi daha verimli mimarilerin geliştirilmesine katkı sağlamıştır (Kamilaris ve Prenafeta-Boldú, 2018; Zhang vd., 2023).

1.4.4. GoogLeNet/Inception (2014): Hesaplama Verimliliği

Google araştırma ekibi tarafından geliştirilen Inception (GoogLeNet) daha verimli ve hesaplama açısından uygun bir derin ağ yapısı oluşturma amacıyla tasarlanmıştır. ImageNet LSVRC-2014 yarışmasını kazanarak büyük bir başarı elde etmiştir (Yuesheng vd., 2021).



Şekil 6. Inception derin ağ modeli (Colaco vd., 2023)

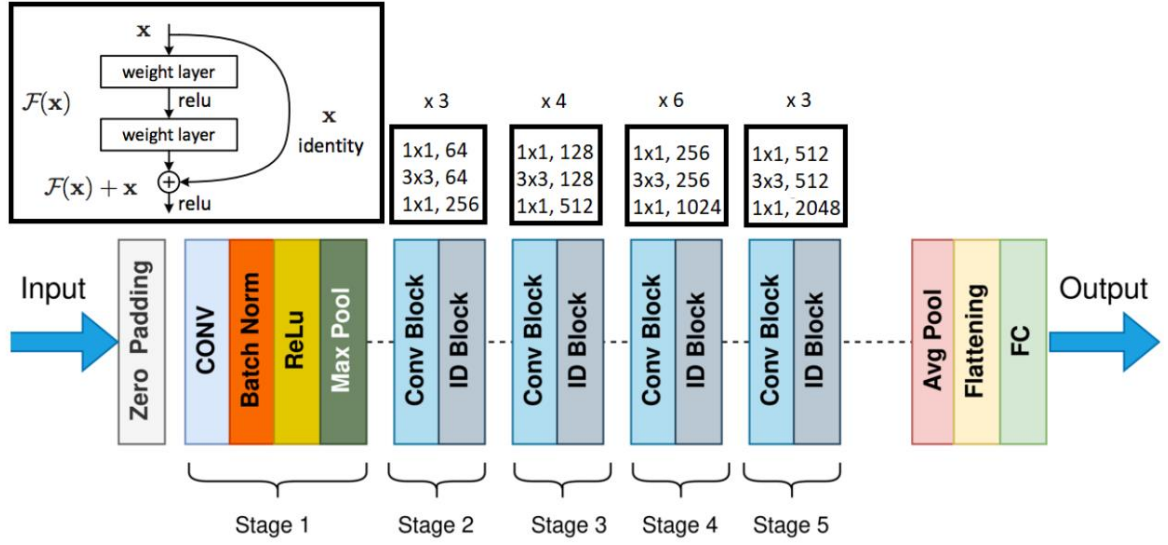
Şekil 6’da gösterilen Inception derin ağ modeli, “kavrama modülleri” adı verilen benzersiz bir yapı kullanmaktadır. Bu modüller çeşitli boyutlardaki filtreleri (1×1 , 3×3 , 5×5) ve maksimum havuzlama işlemlerini birleştirir. Her modül farklı boyutlardaki evrişim ve havuzlama işlemlerini birleştirerek farklı seviyelerde bilgi çıkarımı yapmaktadır. Bu modüller aynı katmanda paralel olarak çalışarak hesaplama maliyetini düşürmektedir. Orta katmanlarda “yardımcı sınıflandırıcılar” kullanılarak gradyan yayılımını iyileştirilir ve ezberlemeyi azaltılır. Inception mimarisi hesaplama maliyetini düşürürken derin ağların performansını artırmak için optimize edilmiştir. Ayrıca çeşitli boyutlardaki filtrelerin birleşimi, modelin genelleme gücünü artırmaktadır (Colaco vd., 2023; Szegedy vd., 2017).

1.4.5. ResNet (2015): Derinliğin Gücü

ResNet, Microsoft Research tarafından 2015 yılında geliştirilmiş ve çok derin ağların eğitimi sırasında ortaya çıkan “gradyan sönümlenmesi” sorununu çözmeyi amaçlamıştır. ImageNet LSVRC-2015 yarışmasını kazanarak bu alanda başarısını ispatlamıştır (He vd., 2016).

ResNet, “artık blok” adı verilen bir yapı kullanmaktadır. Bu bloklar girişteki sinyali çıktı ile birleştirerek gradyan sönümlenmesini önlemeyi amaçlamaktadır. Her blok iki veya

daha fazla evrişim katmanı ve bu katmanlar arasında bir atlama bağlantısından oluşur. Bu yapı gradyanların geriye doğru daha kolay yayılmasını sağlar. ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152 gibi farklı derinliklerde versiyonları bulunmaktadır. Her bir versiyon artan sayıda artık blok içermektedir. ResNet’in en büyük yeniliği, derin ağların gradyan sönümlenmesi sorunu olmadan eğitilebilmesini sağlayan “artık öğrenme” tekniğidir. Bu teknik derin öğrenme araştırmalarında yeni bir standart haline gelmiştir (Wu vd., 2019).



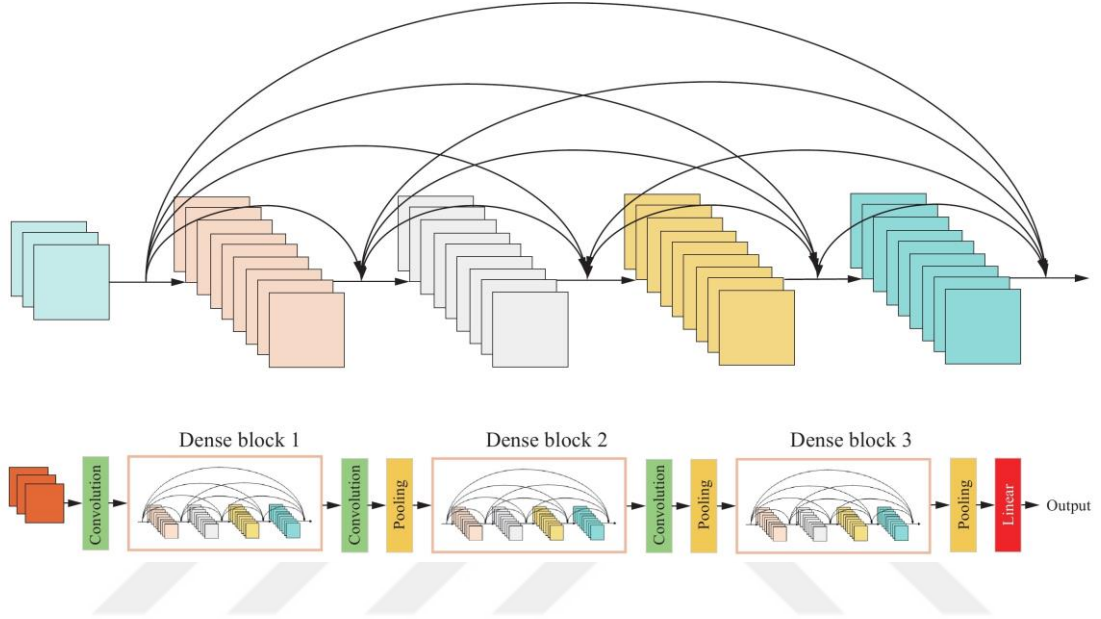
Şekil 7. ResNet derin ağ modeli (Colaco vd., 2023)

Şekil 7’de gösterilen ResNet derin ağ modeli, derin öğrenme literatürüne önemli bir katkı sağlayarak derin yapılarla çalışmanın zorluklarını aşmada önemli bir çözüm sunmuştur. Geleneksel olarak ağ derinliği arttıkça gradyan sönümlenmesi gibi sorunlar nedeniyle model performansı düşerken, ResNet’in getirdiği “atlamalı bağlantılar” sayesinde çok daha derin ağlar oluşturulabilmiştir. Bu mimari, ağırlık güncellemelerinde kalan hataların öğrenilmesine olanak tanıyarak modelin verimli bir şekilde derin özellikleri öğrenmesini sağlamıştır (Colaco vd., 2023; Song vd., 2020).

1.4.6. DenseNet (2017): Yoğun Bağlantılar

DenseNet 2017 yılında geliştirilmiş ve her katmanın tüm önceki katmanların çıkışlarıyla bağlantılı olduğu bir yapı sunmuştur. Amaç bilgi akışını ve gradyan yayılımını iyileştirerek daha verimli bir ağ yapısı oluşturmaktır. DenseNet, “yoğun bloklar” adı verilen

ve her katmanın tüm önceki katmanlardan gelen girdileri aldığı bloklar kullanmaktadır. Her blok tüm önceki katmanların çıktısını birleştirerek özelliklerin zengin bir temsiliyi sağlamaktadır. Yoğun bloklar arasındaki geçiş katmanları, boyut azaltma ve düzenleme işlevi görmektedir. DenseNet daha az parametre ile daha iyi performans elde ederek gradyanların daha etkin bir şekilde yayılmasını sağlamaktadır (Zhang vd., 2019).

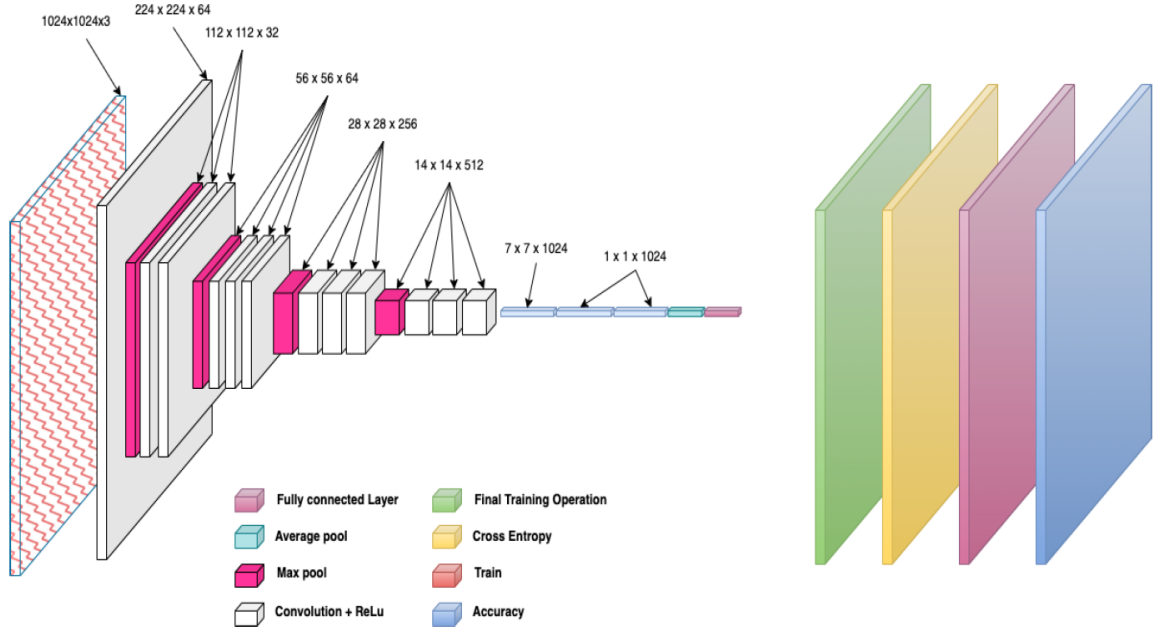


Şekil 8. DenseNet derin ağ modeli (Jia vd., 2021)

Şekil 8’de DenseNet derin ağ modeli gösterilmektedir. Her bir katman kendisinden önce gelen tüm katmanların çıkışı doğrudan kullanır. Veri iletimi gerçekleştirilen bu yapı sayesinde bilgi akışını önemli ölçüde artırır. Böylece derin ağlarda sıkça karşılaşılan gradyan sönümlenmesi problemini azaltarak öğrenme sürecini hızlandırır. Bu yoğun bağlantılı yapı, parametre sayısını azaltırken modeli daha hafif hale getirir. DenseNet ağları özellikle veri verimliliği ve hızlı öğrenme gerektiği durumlar için tercih edilmektedir (Jia vd., 2021; Zeng ve Xiao, 2019).

1.4.7. MobileNet (2017): Hafif ESA

MobileNet, Google tarafından mobil ve gömülü cihazlar için geliştirilmiş bir ESA modelidir. Amaç, sınırlı hesaplama ve bellek kaynaklarına sahip cihazlarda yüksek doğrulukta görüntü sınıflandırma ve diğer görevleri yapabilmektir. MobileNet, “ayrık derin evrişim” yapısını kullanarak hesaplama maliyetini düşürmektedir. Bu yapı standart evrişim katmanlarına göre parametre sayısını önemli ölçüde azaltmaktadır (Elfatimi vd., 2022).



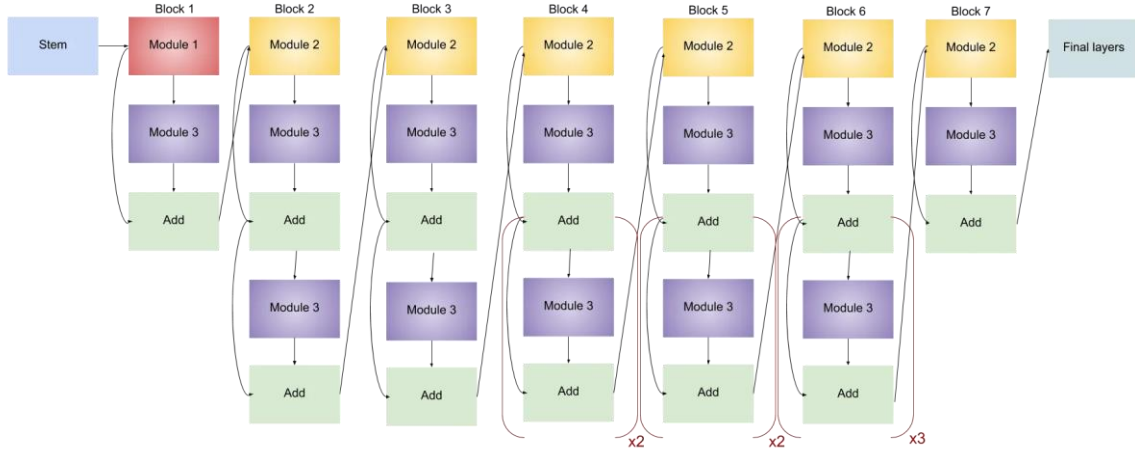
Şekil 9. MobileNetV1 derin ağ modeli (Saini ve Devi, 2023)

Şekil 9’da gösterilen MobileNetV1, ayrık derin evrişim kullanarak modelin hafif olmasını sağlamaktadır. MobileNetV2, “ters artık blok” yapıları ve doğrusal daralma katmanları kullanarak performansı artırmaktadır. MobileNetV3, Sinirsel Mimari Arama (SMA) ile optimize edilmiş ve model boyutunu daha da küçültürken doğruluğu da artıran gelişmiş bir versiyondur. MobileNet mimarileri mobil ve gömülü cihazlarda gerçek zamanlı uygulamalar için optimize edilmiş hafif modellerdir (Saini ve Devi, 2023; Yu vd., 2023).

1.4.8. EfficientNet (2019): Verimlilik

EfficientNet, Google AI ekibi tarafından 2019 yılında geliştirilmiş ve model ölçeklendirme konusunda yeni bir yaklaşım sunmuştur. Bu yaklaşımın hedefi, genişlik, derinlik ve çözünürlüğü optimize eden bileşik bir ölçeklendirme yöntemiyle, daha az parametre ve daha yüksek doğruluk sağlamaktır.

Şekil 10’da EfficientNetB0 temel derin ağ modeli gösterilmektedir. EfficientNet temel model (B0) ve bu modelin ölçeklendirilmiş versiyonları (B1-B7) olmak üzere çeşitli boyutlarda modellerden oluşmaktadır. Bu modelde genişlik, derinlik (katman sayısı) ve çözünürlük (giriş boyutu) optimizasyonu birlikte yapılmaktadır. Bu birleşik ölçeklendirme stratejisi hesaplama maliyetlerini düşürürken performansı da artırmaktadır (Tan ve Le, 2019; Koonce ve Koonce, 2021).



Şekil 10. EfficientNetB0 derin ağ modeli (Tan ve Le, 2019)

1.5. Meta-Sezgisel Optimizasyon

Optimizasyon diğer adıyla eniyileme, bir problemin çözüm uzayındaki en iyi sonucu veren çözümün bulunmasıdır. Günümüzde optimizasyon problemlerinin çözümünde matematiksel ve sezgisel teknikler kullanılmaktadır. Çözüm uzayı geniş olan problemlerde, deterministik yöntemler tüm arama uzayını taradığı için daha maliyetli olmakta ve/veya kabul edilebilir sürede çözümü üretememektedir. Bu tip problemlerde arama uzayını sezgisel olarak tarayarak daha kısa sürede çözüme ulaşan veya çözüme yakınsayan sezgisel (heuristic) algoritmalar kullanılabilir. Günümüzde temel sezgisel metotlar birleştirilerek ya da iyileştirilerek meta-sezgisel (meta-heuristic) algoritmalar geliştirilmiştir. Meta-sezgisel optimizasyon (MSO) yöntemleri, geleneksel eniyileme yöntemlerinin kabul edilebilir bir çözüm üretemediği karmaşık eniyileme problemleri için kabul edilebilir bir zaman diliminde etkin ve uygun çözümler üretebilen yaklaşık algoritmalarlardır.

MSO çok boyutlu arama uzayında en uygun çözümü bulmayı hedefleyen genel amaçlı optimizasyondur. Sezgisel yaklaşımların ötesine geçen ve geniş bir yelpazede farklı problem türlerini çözmeyi amaçlayan bir dizi strateji ve yöntemden oluşur. Bu teknikler geleneksel optimizasyon yöntemlerinin yetersiz kaldığı veya verimliliğini kaybettiği durumlarda, özellikle büyük, karmaşık veya çok boyutlu veri setleriyle çalışırken global optimuma yakın çözümler bulma konusunda etkili olurlar (Kahraman vd., 2023). Meta-sezgisel algoritmalar (MSA) doğadan ilham alır ve biyolojik, sosyal veya fiziksel süreçlerin davranışlarını simüle ederler. Örneğin, biyolojik evrim süreçleri, sosyal sistemlerdeki işbirlikçi davranışlar veya

fiziksel sistemlerdeki enerji minimizasyonu gibi fenomenler bu algoritmaların altında yatan temel prensipleri yani esinlenilen kaynakları oluşturur (Aras vd., 2021).

Algoritma 1. MSO için genel algoritma sözde kodu

- | | |
|----------------|---|
| Adım 1. | <i>Başlat:</i> ilklendirme parametrelerini ayarla, popülasyonu rastgele çözüm adayları ile başlat ve her çözümün uygunluğunu değerlendir, |
| Adım 2. | <i>Döngü:</i> sonlandırma kriteri sağlanana kadar arama sürecine devam et, |
| Adım 3. | <i>Seçim:</i> bir seçim yöntemi kullanarak ebeveynleri seç, |
| Adım 4. | <i>Komşuluk Araması:</i> seçilen ebeveynlerden yeni çözümler oluştur, |
| Adım 5. | <i>Çeşitlilik:</i> çözümler üzerinde rastgele değişiklikler yap, |
| Adım 6. | <i>Değerlendirme:</i> yeni çözümlerin uygunluklarını hesapla, |
| Adım 7. | <i>Güncelleme:</i> yeni çözümler ile popülasyonu güncelle, |
| Adım 8. | <i>Sonuç:</i> en iyi çözüm setini döndür, |
| Adım 9. | Sonlandırma kriterleri sağlanana kadar döngüye devam et (adım 2). |
-

MSO'nun güçlü yönlerinden birisi belli bir problem alanına özgü olmamasıdır. Özellikle karmaşık, doğrusal olmayan veya belirsiz problem uzayında da kullanılmak üzere tasarlanmıştır (Sonmez vd., 2024). Geleneksel optimizasyon tekniklerinin genellikle yerel minimumda takılıp kaldığı veya çözüm uzayının karmaşıklığı nedeniyle etkin bir şekilde arama yapamadığı durumlarda, meta-sezgisel algoritmalar (MSA) yaklaşımlar bu engelleri aşarak daha iyi çözümler bulabilmektedir (Gürgen vd., 2022; Duman vd., 2021).

Günümüzde pek çok MSA geliştirilmiş olmakla birlikte genel adımlara sahiptirler. Bu adımlar, farklı özelleştirilmiş MSA varyasyonları için temel oluşturmaktadır (Guvenc vd., 2021). Her algoritma bu adımları farklı şekillerde uygulayarak kendi optimizasyon stratejilerine göre adapte etmektedir. MSA genel adımları şu şekilde verilebilir:

1. **Başlat:** Algoritma, başlangıçta rastgele çözüm adaylarından oluşan bir popülasyon oluşturur. Bu çözümler genellikle problemin arama uzayının rastgele bölgelerinde başlatılır ve her bir çözümün uygunluk değeri hesaplanır.
2. **Seçim:** Algoritma, yeni çözümler oluşturmak için mevcut popülasyondan ebeveyn bireyler seçer. Bu seçim süreci çeşitli yöntemlerle (örn. Rulet Tekerleği, Turnuva, UÇD) yapılabilir.

3. Komşuluk Araması: Seçilen bireylerden yeni çözümler oluşturulur. Bu süreç genellikle iki ebeveyn çözümün bilgisinden yeni çözümler üretmektir.
4. Çeşitlilik: Yeni çözümler üzerinde, arama uzayının farklı bölgelerini keşfetmek ve popülasyonun çeşitliliğini artırmak için rastgele değişiklikler yapılır
5. Değerlendirme: Yeni çözümlerin uygunluk değeri/leri hesaplanır. Bu değer/ler çözümlerin problemin amaçlarına ne kadar uygun olduğunu gösterir.
6. Güncelleme: Mevcut popülasyon, yeni çözümlerin de popülasyona dahil edilip en uygun bireylerin seçilmesiyle güncellenir. Bu adımda en iyi, rastgele, turnuva gibi çeşitli stratejiler kullanılabilir.
7. Sonuç: Önceden belirlenmiş bir durdurma kriterine ulaşıldığında (örn. belirli bir iterasyon sayısı, zaman sınırı veya çözüm uygunluk değeri) en iyi çözüm/ler döndürülür.

MSA'lar genellikle rastgelelik içeren stokastik (değişken, rastlantısal) ve iteratif yöntemlerdir. Stokastik süreç, algoritmanın farklı çözümleri keşfetmesini ve yerel minimumlardan kaçınmasını sağlar. İteratif süreç, mevcut çözümler üzerinde küçük değişimler yapılarak belirli bir durdurma kriteri sağlanıncaya kadar devam eder. MSA'lar, genellikle tek bir çözüm yerine, aynı anda birden fazla çözüm adayı ile çalışır. Böylece çözümlerin paralel olarak araştırılması ve daha geniş bir arama alanının taranması sağlanır (Duman vd., 2022; Kahraman vd., 2022). İteratif süreç, çözüm adayları içinden yeni çözümler üreterek en iyi çözüme yakınsamayı sağlar. Yani başka bir deyişle çözüm, iterasyonlar sonucu evrimleşir/yakınsar.

Mevcut çözüm adayları arasından yeni çözümler oluşturmak için yapılan birey seçimi, popülasyon tabanlı optimizasyon tekniklerinde önemli bir rol oynar. Yaygın kullanılan seçim yöntemleri, rulet tekerleği, turnuva seçimi yöntemi, aç-göz algoritması, elitizm, rastgele ve uygunluk-çeşitlilik dengesi (UÇD) olarak bilinmektedir (Kahraman vd., 2020).

1.5.1. Rulet Tekerleği Seçimi

Rulet tekerleği seçimi yöntemi, bireylerin uygunluk değerlerine orantılı olasılık değerine göre rastgele birey seçimi yapar. Rulet tekerleği seçimi, bireylerin uygunluk değerlerine göre rastgele seçildiği bir yöntemdir. Bu yöntemde yüksek uygunluk değerine sahip bireylerin seçilme şansı daha yüksektir. Rulet tekerleği yöntemi popülasyonda

çeşitliliği korurken, daha iyi çözümlerin popülasyon içinde kalmasına daha çok olanak tanır. Böylece farklı çözümlerin keşfedilmesi sağlanır (Guo vd., 2020). Her bireyin uygunluk değeri f_i olarak tanımlandığında, birey i için seçilme olasılığı p_i şu şekilde hesaplanmaktadır:

$$p_i = \frac{f_i}{\sum_{j=1}^n f_j} \quad (8)$$

Denklem 8’de n , popülasyondaki toplam birey sayısını temsil eder. Denkleme göre, uygunluk değeri yüksek olan bireyin seçilme ihtimali daha yüksek olacaktır.

1.5.2. Turnuva Seçimi Yöntemi

Turnuva seçimi, rastgele seçilen küçük bir grup birey arasında yapılan rekabeti temel almaktadır. Bu grupta en yüksek uygunluk değerine sahip birey, bir sonraki nesile geçmek üzere seçilir. Bu yöntem uygunluk değeri yüksek bireylerin popülasyon içinde hızla yayılmasını sağlar. Aynı zamanda daha az uygun bireylerin tamamen elenmesini engelleyerek popülasyonun genetik çeşitliliğini korur (Wang vd., 2019). Bir turnuvada k -adet birey olduğunda ve bu bireylerden birinin uygunluğu f_i diğer tüm bireylerin uygunluklarından büyükse, birey i için seçilme olasılığı şu şekilde hesaplanmaktadır:

$$p_i = \left(\frac{1}{n}\right) \left(1 - \left(1 - \frac{f_i}{\sum_{j=1}^k f_j}\right)^k\right) \quad (9)$$

Denklem 9’da n , popülasyondaki toplam birey sayısını temsil eder. Popülasyondan seçilen k -adet bireyden j . bireyin uygunluk değeri f_j ile gösterilmektedir. Bu formül turnuvada bir bireyin kazanma olasılığını ifade etmektedir.

1.5.3. Aç-göz Algoritması

Aç-gözlü seçim, popülasyondan en yüksek uygunluk değerine sahip bireylerin doğrudan seçildiği bir yöntemdir. Bu yaklaşımda en iyi çözümler hızlı bir şekilde bir sonraki nesile aktarılır. Aç-gözlü seçim popülasyonun uygunluğunu hızla artırma potansiyeline sahiptir ve genellikle optimizasyon sürecinin son aşamalarında kullanılır. Ancak bu yöntem,

popülasyonun genetik çeşitliliğini hızla azaltabilir ve arama sürecinin yerel bir optimuma sıkışmasına yol açabilir. Aç-gözlü seçim diğer seçim stratejileriyle birlikte kullanıldığında global en iyi çözüme ulaşılmasına yardımcı olmaktadır (Nadeem vd., 2022).

1.5.4. Elitizm Seçimi

Elitizm, popülasyonun en uygun bireylerini doğrudan bir sonraki nesile taşıyan bir seçim yöntemidir. Bu strateji popülasyonun genel uygunluğunu artırarak global en iyi çözüme ulaşma olasılığını yükseltir. Ancak aşırı elitizm, popülasyonun çeşitliliğini azaltabilir ve yerel optimumlarda sıkışma riskini artırabilir. Bu sebeple elitizm genellikle diğer seçim yöntemleriyle dengeli bir şekilde kullanılmaktadır (Akay vd., 2022).

1.5.5. Rastgele Seçim

Rastgele seçim popülasyondaki bireylerin tamamen rastgele bir şekilde seçildiği basit bir yöntemdir. Her bireyin seçilme olasılığı eşittir ve uygunluk değerine bakılmaksızın seçim yapılır. Bu yöntem genellikle popülasyonun çeşitliliğini korumak ve arama sürecinin erken aşamalarında farklı bölgelerde keşif yapmak için kullanılır. Ancak uygunluk değerlerine göre ayırım yapmadığı için genellikle tek başına kullanılmamakta ve diğer seçim yöntemleriyle birleştirilmektedir (Abd Elaziz vd., 2021).

1.5.6. Uygunluk-Çeşitlilik Dengesi Seçimi

Uygunluk-çeşitlilik dengesi (UÇD) yöntemi, bireylerin uygunluk değerlerini ve en iyi bireye olan mesafelerini birlikte değerlendirerek uygunluk ve çeşitlilik arasında dengeleme yapmaktadır. Bu yaklaşım, yüksek uygunluk değerine sahip bireylerin seçilmesi yanında popülasyonun çeşitliliğini korumayı da hedeflemektedir. UÇD seçiminde, arama uzayının daha geniş bir kısmı keşfedilebilir ve yerel optimum tuzaklarına takılma olasılığı azalır. UÇD seçimi uygunluk ve çeşitliliği dengede tutarak daha etkili bir arama sağlamaktadır (Aras vd., 2021).

$$FDB_i = w * f_i + (1 - w) * d_i \quad (10)$$

Denklem 10'da d_i çözümlerin en iyi çözüme olan uzaklığını f_i uygunluk değerini w ise ağırlığı ifade eden bir sabittir. FDB , çözümün skorunu göstermektedir ve bu skor bilgisine göre aç gözlü veya rulet tekerleği seçimi yapılmaktadır. Bu yöntem popülasyonun çeşitliliğini korumak ve arama alanını daha geniş bir ölçekte keşfetmek için kullanılmaktadır (Kahraman vd., 2020).

1.6. Çok Amaçlı Optimizasyon

Çok amaçlı optimizasyon (ÇAO), birden fazla amaç fonksiyonunu aynı şartlar altında aynı anda optimize etmeyi hedefleyen bir yaklaşımdır. Gerçek dünya problemleri genellikle birbiriyle çatışan birden fazla amacı içerir. Örneğin, aynı anda hem maliyeti azaltma ve hem performansı artırma genellikle çatışan amaçlardır. Bu tür problemlerde tek bir ideal çözüm bulunması zordur. Bunun yerine farklı amaçlar arasında bir denge kurmaya çalışan çözümler kümesi bulunur (Oh ve Kim, 2021). ÇAO problemlerinin genel tanımı Denklem 11'de verilmektedir. ÇAO bir problemde birden fazla amaç fonksiyonunun aynı anda optimize edilmesini gerektirir.

$$\begin{aligned} \min_x f(x) &= (f_1(x), f_2(x), \dots, f_n(x)) \\ g_i(x) &\leq 0, i = 1, 2, \dots, m \\ h_j(x) &= 0, j = 1, 2, \dots, k \\ x &= (x_1, x_2, \dots, x_d) \end{aligned} \quad (11)$$

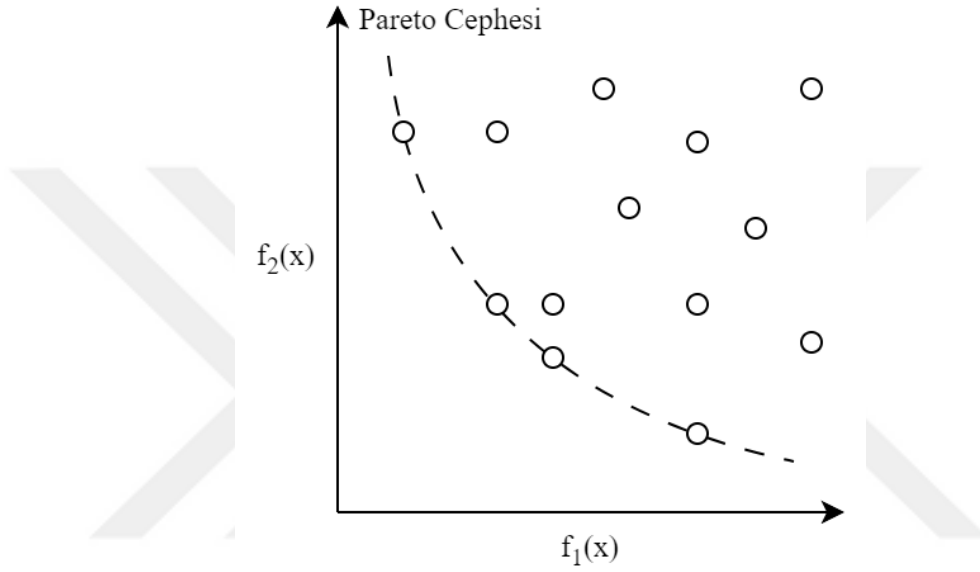
Denklem 11'de popülasyondaki p . çözüm $x^p = (x_1^p, x_2^p, \dots, x_d^p)$ X arama uzayındaki çözümün d -boyutlu karar vektörünü, $f(x)$ n tane amaç fonksiyonlarını, g ve h fonksiyonları modelin kısıtlarını göstermektedir. Çok amaçlı optimizasyon yöntemlerini, pareto tabanlı, ağırlıklandırma ve kısıt tabanlı olarak üç grupta inceleyebiliriz.

1.6.1. Pareto Tabanlı Yöntemler

ÇAO problemleri, amaç fonksiyonlarının çeşitliliği ve çelişkisi nedeniyle zor problemlerdir. Bu tür problemlerin çözümünde pareto optimal kavramı kullanılır. Pareto optimal, tüm amaçlar açısından bir çözümün başka bir çözüm tarafından domine edilemediği durumu ifade eder. Yani bir çözüm, bir amaç fonksiyonu için daha iyi sonuç verirken diğer

bir amaç fonksiyonları için daha kötü sonuç vermezse pareto cephesine uygun olarak kabul edilir (Wei vd., 2022).

ÇAO'nun amacı pareto cephesine uygun çözümler kümesini belirlemektir. Bu cephe karar vericilere çeşitli çözümler sunarak farklı hedefler arasında denge kurmalarını sağlamaktadır. Bir çözüm doğruluğu maksimize ederken, başka bir çözüm maliyeti minimize edebilmektedir (Liang vd., 2019).



Şekil 11. Pareto cephesi ve popülasyon dağılımı

Şekil 11’de ÇAO problemlerinde pareto cephesi ve popülasyonun dağılımı, iki amaç fonksiyonuna sahip problem için temsil edilmektedir. Grafikteki yatay eksen $f_1(x)$, dikey eksen ise $f_2(x)$ amaç fonksiyonlarının değerlerini temsil eder. Bu fonksiyonlar optimizasyon probleminin çözümü sırasında aynı anda optimize edilmeye çalışılan iki farklı hedeflerdir. Kesikli çizgiyle gösterilen eğri pareto cephesi olarak adlandırılır. Pareto cephesi, baskılamamayan çözümler kümesini temsil eder. Baskılamamayan çözümler ise, en az bir amaç fonksiyonu açısından daha iyi uygunluk değerine sahip olup, diğer amaç fonksiyonları açısından en azından mevcut çözümlerle eşit uygunluk değerini koruyan çözümlerdir.

ÇAO problemlerinde, amaçlar genellikle birbiriyle çatıştığından tüm hedefleri en iyi şekilde karşılayan tek bir çözüm bulunması zordur. Bir hedefin iyileştirilmesi çoğu durumda diğer hedeflerin olumsuz etkilenmesi anlamına gelebilmektedir (Loni vd., 2020). Bu sebeple ÇAO’nun temel hedefi pareto cephesine yakınsayan ve çeşitlilik gösteren bir dizi çözüm

bulmaktır. Pareto cephesine başarılı bir yakınsama, çözümlerin pareto cepheye uygun çözümlere yakın olduğunu göstermektedir. İyi bir çeşitlilik ise çözümlerin amaç uzayında geniş bir alana yayıldığını ve farklı çözüm yollarını kapsadığını ifade etmektedir. Bu durum karar vericilere çeşitli seçenekler sunarak farklı amaçlar arasındaki dengeyi daha iyi kurmalarını sağlamaktadır (da Silva vd., 2024).

Pareto tabanlı yöntemler ÇAO problemlerinde çözümler cephe uygunluğuna göre sıralanır. Bu yaklaşım birden fazla hedefi aynı anda optimize etmeye çalışırken çözümlerin pareto cephesine yakınsamasını amaçlar. Cephe uygunluğu bir çözümün başka bir çözümle kıyaslandığında tüm amaçlar açısından daha üstün veya eşit olacağı durumunu tanımlar (Gülmez, 2024).

Baskın Olmayan Sıralı Genetik Algoritmalar

Baskın Olmayan Sıralı Genetik Algoritmalar (Non-dominated Sorting Genetic Algorithms, NSGAs), en yaygın kullanılan pareto tabanlı yöntemlerden biridir. Bu algoritmalarda popülasyon, “baskın olmayan sıralama” tekniğini kullanılarak sıralanır. Sıralama işleminde, her bir çözümün pareto cephesindeki pozisyonu belirlenerek, çözümler öncelik sırasına göre gruplandırılır. NSGAs, pareto cephesinde yer alan çözümleri bulurken çeşitliliği korumak için de “kalabalık mesafesi” metriğini kullanmaktadır (He vd., 2021).

Baskın olmayan sıralama, bir dizi çözümü, pareto baskınlığına göre sıralayarak birkaç öncelik seviyesine ayırmaktadır. Bir çözüm diğer çözümler tarafından domine edilemiyorsa, bu çözüm birinci öncelik seviyesinde gösterilmektedir. Birinci öncelik seviyesindeki çözümler çıkarıldıktan sonra geriye kalan çözümler arasında aynı işlem tekrarlanır. Bu işlemler tüm çözümler sıralanana kadar devam eder (Elghazi vd., 2024).

$$\begin{aligned} rank_i &= \{x_i \in P \mid n_i = 0\} \\ P &\leftarrow P \setminus rank_i \\ rank_{i+1} &= \{x_{i+1} \in P \mid n_{i+1} = 0\} \end{aligned} \tag{12}$$

Denklem 12’de baskın olmayan sıralama hesaplama adımları verilmektedir. Denklemde P popülasyonu, x_i çözüm adayını, n_i , x_i ’yi domine eden çözüm sayısını ifade etmektedir. Baskın olmayan sıralama yapılırken önce $rank_1$ çözümleri bulunur. Sonra $rank_1$ çözümleri popülasyondan çıkarılır ve aynı işlem tekrarlanarak $rank_2$ çözümleri

bulunur. Bu işlemler tüm çözümlere sıralana kadar devam eder. Ranklar öncelik seviyelerini ifade etmektedir.

Pareto tabanlı CAO algoritmalarında çözümler arasında çeşitliliği korumak için kalabalık mesafesi uzaklık metriği kullanılır. Bu yöntem baskın olmayan sıralama yöntemiyle sıralanmış popülasyonlar üzerinde uygulanır. Pareto cephesindeki bir çözümün kendisine en yakın iki çözüme (komşularına) olan amaç uzayındaki uzaklığını ölçmektedir. Bu uzaklık her bir amaç fonksiyon için normalize edilmiş mesafeler kullanılarak hesaplanmaktadır (Tian vd. 2020).

$$d_i^k = \frac{f_i^{k+1} - f_i^{k-1}}{f_i^{max} - f_i^{min}} \quad (13)$$

$$CD_k = \sum_{i=1}^n d_i^k$$

Denklem 13’de kalabalık mesafesi hesaplama adımları verilmektedir. Denklemde her amaç fonksiyonu f_i^k için komşu çözümlere (f_i^{k-1}, f_i^{k+1}) olan mesafesi hesaplanmaktadır. Bireyin kalabalık mesafesi (CD_k), her amaç fonksiyonunun hesaplanmış mesafelerinin normalizasyonlarının toplamı olarak hesaplanmaktadır (Lu ve Zhang, 2020).

Güçlü Pareto Evrimsel Algoritmalar

Güçlü Pareto Evrimsel Algoritmalar (Strength Pareto Evolutionary Algorithms, SPEAs), çözümleri “güç” değerlerine dayalı olarak sıralama yapan başka bir pareto temelli evrimsel algoritmalar. Güç değeri bir çözümün diğer çözümler üzerinde ne kadar baskın olduğunu ölçer. Bir çözümün gücü $S(i)$, o çözüm tarafından baskın kılınan çözümlerin sayısı ile ilişkilidir (Zhao vd., 2023). Her çözüm için güç değeri şöyle hesaplanır:

$$S(i) = \{j \mid x_i \gg x_j\} \quad (14)$$

Denklem 14’te, $S(i)$ i . çözümün güç değerini ve $x_i \gg x_j$ ise x_i çözümünün x_j çözümünü baskıladığını gösterir. Her çözümün uygunluk değeri, kendisinden daha iyi veya eşit çözümlerden ne kadar baskın olduğunu ölçen bir fonksiyon kullanılarak hesaplanır.

$$F(i) = S(i) + D(i) \quad (15)$$

Denklem 15'te $D(i)$, yoğunluk ölçüsüdür (en yakın k -adet komşu çözüme olan uzaklık toplamı) ve çözümlerin popülasyon içindeki çeşitliliğini korumak için kullanılır. SPEAs'ın avantajı pareto cephesine iyi yakınsayan ve çeşitliliği koruyan çözümler üretebilmesidir (Gadhvi vd., 2016).

Ayrıştırırmaya Dayalı Çok Amaçlı Evrimsel Algoritmalar

Ayrıştırırmaya Dayalı Çok Amaçlı Evrimsel Algoritmalar (Multi-Objective Evolutionary Algorithms based on Decomposition, MOEAs/D), ÇAO problemini birçok tek amaçlı alt probleme bölen ve her alt problemi bağımsız olarak optimize eden pareto tabanlı bir yaklaşımdır. Bu yöntem, her alt problemin kendi başına pareto cephesine yakınsayan çözümler üretmesine olanak tanır (Zhang vd., 2022). Çok amaçlı problem bir ağırlık vektör seti kullanılarak n adet alt probleme ayrılır.

Denklem 16'da, her alt problem ($f_j(x)$), farklı bir ağırlık vektörü (λ_j) ile ilişkili tek amaçlı bir optimizasyon problemine ($g_i(x)$) dönüştürülür:

$$\min_x g_i(x) = \sum_{j=1}^n \lambda_j * f_j(x) \quad (16)$$

Her alt problem, ilişkili ağırlık vektörüne göre optimize edilir ve çözümler güncellenir. Bu süreçte, alt problemler arasındaki bilgi alışverişi, çözümlerin pareto cephesine yakınsamasını sağlar. MOEAs/D özellikle büyük ölçekli ÇAO problemleri için etkilidir ve çözümler arasındaki çeşitliliği korumada başarılıdır (Zhang vd., 2024).

1.6.2. Ağırlıklandırma Yöntemleri

Ağırlıklandırma yöntemleri, farklı amaç fonksiyonlarını tek bir amaç fonksiyonuna indirgemek için ağırlık katsayıları kullanmaktadır. Bu yöntemde her bir amaca bir ağırlık atanır ve tüm ağırlıklandırılmış amaçlar birleştirilerek tek bir toplam amaç fonksiyonu oluşturulur (Aziz vd., 2018).

Ağırlıklı Toplam Yöntemi

Ağırlıklı Toplam Yöntemi (ATY) ÇAO problemlerini çözmek için en basit ve yaygın kullanılan yöntemlerden biridir. Bu yöntemde, her bir amaç fonksiyonuna belirli bir ağırlık katsayısı atanır ve bu ağırlıklandırılmış fonksiyonlar bir araya getirilerek tek bir bileşik amaç fonksiyonu oluşturulur. Ortaya çıkan bu yeni amaç fonksiyonu klasik optimizasyon teknikleri kullanılarak optimize edilir (Marler ve Arora, 2010). Ağırlıklı toplam yönteminde n adet amaç fonksiyonu $f_1(x), f_2(x), \dots, f_n(x)$ ve bunlara karşılık gelen ağırlıklar $w_1, w_2, \dots, w_n$ olmak üzere, optimize edilecek tek bir amaç fonksiyonu şu şekilde tanımlanır:

$$\min_x F(x) = \sum_{i=1}^n w_i * f_i(x) \quad (17)$$

Denklem 17’de $w_i \geq 0$ ve $\sum_{i=1}^N w_i = 1$ olacak şekilde ağırlıklar belirlenir. Her ağırlık katsayısı ilgili amacın önem derecesini temsil etmektedir. Ağırlıkların toplamının 1 olması, amaçların toplamının bir bütün olarak dikkate alınmasını sağlamaktadır (Kaddani vd., 2017).

Ağırlıklandırılmış Minimum-Maksimum Yöntemi

Ağırlıklandırılmış Minimum-Maksimum Yöntemi (AMMY) ÇAO problemlerini çözmek için kullanılan alternatif bir ağırlıklandırma yöntemidir. Bu yöntem her bir amaç fonksiyonunun ağırlıklandırılmış maksimum değerlerini minimize etmeyi hedefler. AMMY amaçların en kötü performansını iyileştirmeye çalışır. Her bir amaç fonksiyonu için belirli ağırlıklar kullanarak bu amaçların maksimum değerini minimize etmeye çalışmaktadır (Taha, 2020).

$$\min \max_{i=1,2,\dots,N} \left(\frac{w_i * f_i(x) - f_i^{min}}{f_i^{max} - f_i^{min}} \right) \quad (18)$$

Denklem 18’de f_i^{min} ve f_i^{max} i . amaç fonksiyonunun, minimum ve maksimum değerlerini temsil eder. Ağırlıklar $w_i \geq 0$ ve $\sum_{i=1}^N w_i = 1$ koşulunu sağlar. Amaç, ağırlıklı oranların maksimumunu minimize etmektir. Bu yöntem en kötü durumdaki amacın iyileştirilmesine odaklanmaktadır (Li vd., 2021).

1.6.3. Kısıt Tabanlı Yöntemler

Kısıt tabanlı yöntemler ÇAO problemlerinde amaç fonksiyonlarını kısıtlar olarak ele alarak bir veya birkaç amacın optimize edilmesine odaklanır. Bu yöntemlerde birden fazla amaç fonksiyonu olduğu durumlarda, bazı amaçlar optimizasyonun birincil hedefi olurken, diğer amaçlar da kısıtlar olarak kabul edilir. Bu kısıtlar belirli eşik değerinden (genellikle ε) küçük tutulmaya çalışılır. Kısıt tabanlı yöntemler genellikle esneklik ve hassasiyet sağladığı için karmaşık optimizasyon problemlerinde kullanılmaktadır (Rahimi vd., 2023).

Epsilon Kısıt Yöntemi

Epsilon Kısıt Yöntemi (EKY), ÇAO problemlerinde bir veya birkaç amacın optimize edilmesini sağlarken, diğer amaçları kısıt olarak ele alır. Bu yöntemde, optimize edilmeyen her bir amaç fonksiyonuna bir epsilon (ε) değeri atanır. Optimizasyon sürecinde, amaç fonksiyonlarının bu eşik değerini aşmamasını sağlanır (Yang vd., 2020). ÇAO probleminde n adet amaç fonksiyonu $f_1(x), f_2(x), \dots, f_n(x)$ verildiğinde ε -Kısıt Yöntemi denklem 19'daki gibi formüle edilmektedir:

$$\begin{aligned} \min & f_1(x) \\ \forall i = 2, 3, \dots, N & f_i(x) \leq \varepsilon_i \end{aligned} \quad (19)$$

Denklem 19'da ε_i değerleri, optimize edilmeyen her bir amaç fonksiyonu için belirlenen eşik değerleridir. İlk olarak optimize edilecek bir amaç fonksiyonu seçilir. Diğer amaçlar için uygun ε değerleri belirlenir. Bu eşik değerleri problemin gereksinimlerine ve çözüm alanının genişliğine göre ayarlanır. Belirlenen kısıtlar altında seçilen birincil amaç fonksiyonu optimize edilir (Zhou vd., 2021).

Hedefe Ulaşma Yöntemi

Hedefe Ulaşma Yöntemi (HUY), belirli bir hedef değere ulaşmaya çalışan bir kısıt setiyle optimizasyon yapmaktadır. Bu yöntemde her bir amaç fonksiyonu için bir hedef belirlenir ve bu hedeflere ulaşmak için çözümler optimize edilir. Yöntemin amacı hedeflere mümkün olduğunca yakın çözümler bulmaktır (Gunantara, 2018). ÇAO probleminde her bir amaç fonksiyonu için belirli hedef değerler (g_i) belirlenir ve optimizasyon, bu hedeflerden sapmaların minimize edilmesine odaklanır.

$$\min \max_{i=1,2,\dots,n} \left(\frac{f_i(x) - g_i}{g_i} \right) \quad (20)$$

Denklem 20’de $f_i(x)$ i . amaç fonksiyonunu ve g_i ise bu amaç için belirlenen hedef değeri temsil eder. Yöntem bu şekilde her bir amaç fonksiyonunun hedef değere olan sapmasını minimize etmeye çalışmaktadır.

1.7. ESA Mimarilerinin Çok Amaçlı Optimizasyonu

Genellikle yüksek sayıda parametre içeren ESA, derin öğrenme modellerinin önemli bir sınıfını oluşturmaktadır. ESA’da, parametrelerin optimize edilmesi, modelin performansını önemli ölçüde etkilemektedir. ESA’ların kısa cevap verme süresine sahip olmaları ve aynı zamanda yüksek doğruluk oranlarına ulaşmaları arzulanan bir durumdur. Kısa cevap verme süresi, modelin hızlı bir şekilde tahmin yapabilmesi anlamına gelmektedir. Bu durum, modelin daha az parametreye ve daha basit yapıya sahip olmasını gerektirir. Öte yandan yüksek doğruluk oranı, genellikle fazla katmanlı ve parametrelili karmaşık modellerin kullanılmasını gerektirir. Sonuç olarak ideal cevap süresi ve doğruluk oranı sağlayan modellerin geliştirilmesi için, ESA katmanlarının ve parametrelerinin, doğruluk oranını azaltmadan minimizasyonu, ÇAO problemi olarak karşımıza çıkmaktadır.

Liu ve arkadaşları çalışmasında, yüz görüntülerindeki alt bileşenlerin (göz, burun ve ağız gibi) etiketlenmesi için çok amaçlı ESA kullanmıştır. Kullanılan teknikte daha küçük boyutlu ağ ile performans artırılmıştır (Liu vd., 2015). Çok amaçlı gramer evrimi (da Silva vd., 2024), genetik algoritmalar (Doi vd., 2019) ve çok amaçlı ESA algoritmaları (Wang vd., 2019), zengin optimizasyon teknikleri, hata oranını düşürme, hesaplama maliyetini kontrol etme ve nadir özellikleri ayırt etme gibi çeşitli amaçlara ulaşmada önemli rol oynamaktadır (Loni vd., 2020). Bu yöntemler, ESA mimarilerinin inşası ve ayarlanması sürecinde özgül amaçlara hizmet etmekte ve modelin performansını artırmak için geniş bir parametre yelpazesi üzerinde çalışan stratejileri oluşturmaktadır.

ÇAO yöntemi olarak da kullanılan parçacık sürü optimizasyonu (PSO), ESA yapılarının geliştirilmesinde de kullanılmaktadır. Çok amaçlı PSO yaklaşımında, çeşitli performans kriterleri dikkate alınarak daha sofistike modellerin oluşturulması amaçlanmaktadır. Örneğin, uyku apnesi tespitinde kullanılan çok amaçlı PSOCNN modelinde, sınıflandırma performansını artırılmasının yanında farklı girdi büyüklükleri, veri setlerinin özellikleri ve katman yapılarının model üzerindeki etkisi detaylı incelemiştir

(Mostafa vd., 2020). İnsansız Hava Araçları (İHA) kullanımı için sahne tanıma konusunda yürütülen çalışmada, MOPSO gibi yenilikçi yaklaşımlarla geliştirilen derin öğrenme modelleri kullanılmıştır (Rajagopal vd., 2020). Bu alandaki araştırmalar, veri bilimi ve yapay zekanın sinerjisinden faydalanarak karmaşık sorunlara çözüm üretme potansiyelini artırmaktadır.

ÇAO teknikleri ESA mimarilerinin gelişimine önemli katkılar sağlamaktadır (Elghazi vd., 2024). Sinir ağı mimarilerinin otomatik oluşturulması ve hiper-parametre optimizasyonu, Yapay Açıklıklı Radar (YAR) görüntülerinin iyileştirilmesinde kullanılan ESA tabanlı algoritmalara getirdiği yeni bakış açıları, bu alanlardaki araştırma ve uygulamaların sınırlarını zorlamaktadır. Bu gelişmeler, yapay zekanın karmaşık problemleri çözme kapasitesini önemli ölçüde artırmaktadır (Vitale vd., 2020).

2. YAPILAN ÇALIŞMALAR

Evrişimli sinir ağları (ESA), kullanılabilen pek çok alanda performans artışı sağlamıştır. Ancak büyük işlem hacmi kapasitelerine sahip olmayan gömülü sisteme dayalı alanlarda, gerçek zamanlı işlem gerektiren ve hatta çevrimiçi uygulamalarda, zaman kısıtı sorunuyla karşılaşmaktadır. Çoğu çalışmada, modelin doğruluk ve genelleme yeteneğine odaklanılmış, çıktıların üretilme hızı ve sistemin tepki süresi gibi faktörler dikkate alınmamıştır. Bu durum interaktif uygulamalar ve düşük işlem hacmine sahip sistemlerde ESA mimarilerinin uygulanamamasına yol açmaktadır. Bu nedenle daha optimal ESA mimarilerine ihtiyaç duyulmaktadır. “Başarı performansından taviz vermeden ESA modelleri tasarlanabilir mi?” sorusu ezeli bir araştırma sorusu olacaktır.

Optimal ESA için çok amaçlı optimizasyon kullanılan çalışmalar yapılmıştır (Liu vd., 2015; da Silva vd., 2024; Doi vd., 2019; Wang vd., 2019; Loni vd., 2020; Mostafa vd., 2020; Rajagopal vd., 2020; Elghazi vd., 2024; Vitale vd., 2020). Bu çalışmaların çoğunda, amaçlar arası dengeye yeterince odaklanılmadığı ve amaçlar arasındaki potansiyel çatışmanın nadiren ele alındığı görülmektedir. Çok amaçlı optimizasyon süreçlerinde ideal olan, amaçlar arasındaki doğal çatışmaları dengeleyerek birden fazla amaca eş zamanlı ulaşılmaya çalışılır. Çoğu çalışmada, çatışmayı minimize etmek amaçlanırken (örneğin modelin karmaşıklığı ve hesaplama maliyeti gibi) diğer faktörler göz ardı edilmektedir. Bu durum belirli amaçların aşırı derecede tercih edildiği yani bazı kritik performans ölçütlerinin ihmal edildiği anlamına gelmektedir.

Bu tez çalışmasında, birden fazla amacı eş zamanlı optimize ederken model doğruluğu ve yanıt süresi arasında denge kurulmaya çalışılmıştır. Bu amaçla, çok amaçlı optimizasyon teknikleri ile hiper-parametrelerin optimizasyonu sağlanarak ESA model tasarımının gerçekleştirilmesi hedeflenmiştir. Önerilen çok amaçlı optimizasyona dayalı evrişimli sinir ağı model tasarımı (ÇESA) yönteminin hedeflediği özgün katkıları şöyle özetlenebilir:

1. ESA'nın daha hızlı cevap vermesini sağlamak: Çoğu ESA model tasarımında doğruluk ve genelleme yeteneğine odaklanılır. Bu durumda, blok, katman sayıları, filtre boyutları ve tekrarlılıkları gibi parametrelerden oluşan uzayın büyüklüğü göz ardı edilebilmektedir. Sonuçta tasarlanan ESA modelinde, doğruluğa az katkılı ya da katkısız parametreler de çalışma anında işlem görür. Bu problemi aşmak için budama teknikleri kullanılabilmektedir. Budama teknikleri daha genel bir kırpma yaparken

başarıdan ödün verilebilir. Başarıdan ödün vermeden budama yapmak için daha yerel, yani birim olarak kırpma yapılması gerekir. Böylece ESA modellerinin doğruluk performansından taviz verilmeden daha hızlı yanıt süresine sahip tasarımı yapılabilir.

2. Amaçlar arası denge: Model tasarımları yapılırken cevap süresine önem vermeden genellikle yüksek doğruluk oranı hedeflenmektedir. Bu modellerde, hızlı yanıt için parametreler ayarlanmaya çalışıldığında doğruluk oranı orantısız düşmektedir. Önerilen ÇESA yöntemi ile, cevap süresi ve doğruluk oranı arasında denge kurulmaya çalışılır. Böylece birbiriyle çatışan yüksek doğruluk ve düşük cevap süresi hedefleri arasında denge kurulabilir. Bu yaklaşım, doğruluktan taviz verildiği oranda daha hızlı yanıt süresi veren sistemlerin veya daha uzun yanıt süreli doğruluk oranı daha yüksek sistemlerin, aynı sistemle otomatik tasarlanabilmesine olanak verir.
3. Hiper-parametrelerin otomatik ayarlanması: DÖ modelleri tasarımında hiper-parametreler genellikle deneyimli kişilerce el ile ayarlanmaktadır. Bu ayarlamalar bazen seçim bazen kalibrasyon şeklinde olup tasarım sürecini zorlaştırmaktadır. Önerilen ÇESA yönteminde, hiper-parametre seçim ve kalibrasyonları otomatik yapılmaktadır. Bu, daha hızlı model tasarımı yapılmasını sağlarken optimum parametrelerle çalışma avantajlarını da getirmektedir.

Derin öğrenme modelleri tasarlandıktan sonra iki temel aşama daha vardır: modelin eğitimi ve test edilmesi. Modelin eğitimi ve değerlendirilmesi için veri; eğitim, doğrulama ve test veri grupları olmak üzere üçe ayrılır. Eğitim verisi, modelin öğrenmesini sağlamak için kullanılırken doğrulama verisi, modelin eğitim sırasında performansını izlemek için kullanılır. Test verisi ise eğitim ve doğrulama sırasında hiç kullanılmaz ve modelin genel performansını değerlendirmek için ayrılır. Tasarlanan model eğitim aşamasına alınır. Eğitim sürecinde, giriş verisi modele verilir ve modelin çıktısı hesaplanır. Çıktı ile gerçek değer arasındaki hata, bir kayıp fonksiyonu yardımıyla ölçülür. Bu hata bilgisi, modelin parametrelerini güncellemek için geriye doğru yayılır. Çevrim sayısı tamamlanana kadar modelin parametreleri güncellenmeye devam edilir. Çevrim tamamlanınca DÖ tasarımı gerçekleştirilmiş olur. Eğitim tamamlandıktan sonra model, test aşamasına geçirilir. Bu aşamada, model daha önce hiç görmediği test verisi üzerinde değerlendirilir. Performans,

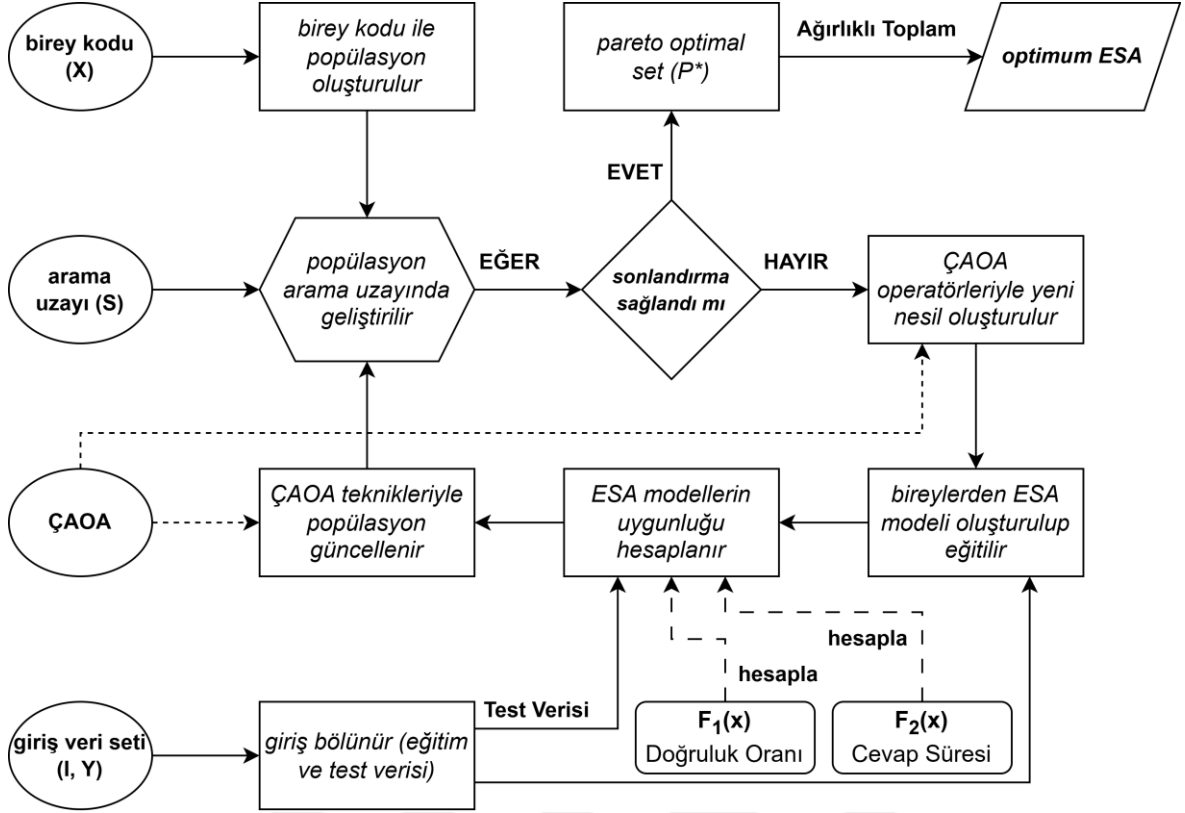
doğruluk oranı, tepki süresi, model karmaşıklığı gibi metrikler kullanılarak ölçülür. Önerilen ÇESA yöntemi birinci aşamada çalışmaktadır.

Bu çalışmada, dengeli ve etkili ESA model tasarımı için çok amaçlı yöntem önerilmektedir. Önerilen yöntem çeşitli veri setleri (MNIST, CIFAR-10 ve CIFAR-100) ile eğitilerek test edilmiştir. Çalışmada, genel bilgiler bölümünde verilen ESA modellerinden DenseNet121, DenseNet169, DenseNet201, EfficientNetB0, EfficientNetB2, EfficientNetB3, EfficientNetB5, ResNet50, ResNet101, ResNet152, ResNet101V2, ResNet152V2, NASNetMobile, NASNetLarge, MobileNet, MobileNetV2 modelleri ile ÇESA ile tasarlanan 4 model çalışılmıştır.

2.1. Önerilen ÇESA Yöntemi

Otomatik ESA modeli üreten, çok amaçlı optimizasyon tabanlı ESA (ÇESA) yöntemi temel olarak iki aşamadan oluşmaktadır. İlk aşamada uzmana gerek kalmadan ESA ağ modeli oluşturulmaktadır. İkinci aşamada, oluşturulan ağ modelinin hiper-parametreleri optimize edilmektedir. Önerilen yaklaşımda, başlangıçta bir parametre havuzundan seçim yapılarak derin öğrenme modeli tasarlanır. İstenirse varolan bir model referans alınabilir. Yani model bağımsız olup sınırsız sayıda ağ üretimine uygundur. Daha sonra üretilen ağ üzerinde hiper-parametreler otomatik olarak optimize edilir. Şekil 12’de, genel yapısı verilen yapıda olup doğruluk oranı ($F_1(x)$) ve cevap süresi ($F_2(x)$) amaç fonksiyonları dengede tutularak, çok amaçlı optimizasyon yapıldığı temsil edilmiştir. Burada, amaç fonksiyonu sayısı da ikiden fazla olabilir.

Önerilen yöntemde, giriş veri seti (I, Y), çok amaçlı optimizasyon algoritması (ÇAOA), birey kodu (X) ve arama uzayı (S) parametre olarak verilir. Giriş veri setinde I girdiyi, Y etiketini ifade eder. Giriş veri seti ÇESA yönteminde eğitim ve test verisi olarak iki gruba ayrılır. Eğitim verisi model eğitim sürecinde (ileri besleme – geri yayılım ile ağırlıkların güncellenmesinde), test verisi ise model uygunluk değerlendirilmesinde (sadece ÇAO algoritmasında amaç fonksiyonlarının uygunluk değeri hesaplamasında) kullanılır. ÇAOA olarak çok amaçlı optimizasyon teknikleri kullanan her algoritma kullanılabilir. Dinamik yapısıyla ÇESA yönteminin tüm ÇAO algoritmalarına entegre olabilme kabiliyeti vardır. Birey kodu (X), ESA oluşturma ürünütüsünü tanımlar. Arama uzayı (S), ayrık değerli daraltılmış hiper-parametre uzayını gösterir.



Şekil 12. ÇESA yöntemiyle otomatik DÖ model tasarımı blok diyagramı

Şekil 12’de gösterilen ÇESA yöntemi blok şemasının sözde kodu Algoritma 2’de verilmektedir. Önerilen yaklaşım, farklı performans metriklerini (doğruluk oranı – F_1 ve yanıt süresi – F_2) aynı anda optimize etmeye çalışarak, ESA’nın genel performansını artırmayı ve cevap süresini azaltmayı hedeflemektedir. Arama sürecinde giriş veri seti, eğitim ($I_{\text{eğitim}}, Y_{\text{eğitim}}$) ve test verisi ($I_{\text{test}}, Y_{\text{test}}$) olarak iki gruba ayrılır. Eğitim verisi modelin öğrenmesi için kullanılır. Test verisi modelin uygululuk değerini hesaplamak için ayrılan veri setidir.

Birey kodu (X) ile popülasyon büyüklüğü (p) kadar bireyden oluşan popülasyon (P), arama uzayı (S) sınırları içerisinde oluşturulur. Her birey potansiyel bir ESA modelini temsil eder. Her bir birey model olarak eğitilir ve uygunluk değeri hesaplanır. ÇAOA operatörleri (örneğin çaprazlama, mutasyon vb.) kullanılarak mevcut popülasyondan (P_g) bir sonraki nesil ($P_{(g+1)}$) oluşturulur. Bireyin tasarım parametreleriyle bir ESA modeli oluşturulur. Bu model, eğitim seti ($I_{\text{eğitim}}, Y_{\text{eğitim}}$) ile eğitilir. Test seti ($I_{\text{test}}, Y_{\text{test}}$) ile modelin uygunluğu değerlendirilir. Popülasyon üzerinde ÇAOA operatörleri kullanılarak popülasyonun en iyilenmesi hedeflenir. Sonlandırma kriteri (maksimum nesil sayısının (G) tamamlanması) sağlanmadığı sürece popülasyonun iteratif olarak güncellenmesine devam edilir. Her bir

iterasyonda (g), bir yandan yeni nesil $P_{(g+1)}$ oluşturulurken bir yandan ÇAOA teknikleriyle mevcut nesil P_g güncellenir. Sonlandırma kriterleri sağlandığında optimum çözüm adaylarını barındıran pareto optimal set (P^*) elde edilmiş olur. Set içerisindeki aranan çözüm, ağırlıklı toplam yöntemiyle belirlenir. Belirlenen çözüm optimum ESA modelini temsil etmektedir.

Algoritma 2. ÇESA yöntemiyle otomatik DÖ model tasarımı sözde kodu

Girdi: *giriş veri seti (I, Y), ÇAOA, birey kodu (X), arama uzayı (S), maksimum iterasyon sayısı (G)*

Çıktı: *optimum ESA modeli*

→ *giriş veri seti (I, Y) eğitim ve test kümelerine ayrılır ($I_{\text{eğitim}}, Y_{\text{eğitim}}, I_{\text{test}}, Y_{\text{test}}$)*

→ *p büyüklüğünde birey koduna (X) göre arama uzayında (S) başlangıç popülasyonu (P_0) oluşturulur (Bknz. Algoritma 3)*

→ *nesildeki tüm bireylerin (P_0) uygunluk değeri hesaplanır*

for (her nesil g için 0'dan G 'ye kadar)

P_g popülasyonundan ÇAOA operatörleriyle yeni nesil $P_{(g+1)}$ oluşturulur

for (P_{g+1} popülasyonundaki her birey i için)

$\text{model} \leftarrow P_{g+1}(i)$ ile yeni ESA modeli oluşturulur

$\text{model}_{\text{eğitilmiş}} \leftarrow \text{model} (I_{\text{eğitim}}, Y_{\text{eğitim}})$ seti ile eğitilir

 (*doğruluk oranı, cevap süresi*) $\leftarrow \text{model}_{\text{eğitilmiş}} (I_{\text{test}}, Y_{\text{test}})$ ile test edilir

$F_1(i) \leftarrow \text{doğruluk oranı}, F_2(i) \leftarrow \text{cevap süresi}$

end for

$P_{(g+1)}$ popülasyonu ÇAOA teknikleriyle güncellenir

$P_{(g+1)}$ popülasyonundan P^* pareto optimal seti belirlenir

end for

optimum ESA modeli ağırlıklı toplam yöntemi ile P^* setinden belirlenir

return optimum ESA modeli

2.2. Hiper-Parametrelerin ve Arama Uzayının Belirlenmesi

Hiper-parametrelerin optimizasyon yöntemleriyle belirlenmesi, derin öğrenme modellerinin performansını önemli ölçüde etkileyen bir faktördür. Ancak arama uzayının

büyük olması, özellikle karmaşık modellerde, arama sürecini zorlaştırır ve daha fazla hesaplama maliyeti gerektirir. Bu nedenle, arama uzayının daraltılması ve hiper-parametre değer aralıklarının sınırlandırılması, optimizasyon sürecini hem daha verimli hem de hızlı hale getirir.

Güncel ve güçlü ESA modellerinde, çeşitli hiper-parametrelerin belirli aralıklarda tutulması, deneysel olarak elde edilen sonuçlarla desteklenmiştir (Gülcü ve Kuş, 2020; Tian vd., 2021; Springenberg vd., 2014; Simonyan ve Zisserman, 2015; Hinton, 2012; Wu ve Gu, 2015a; Wu ve Gu, 2015b; Park ve Kwak, 2017). Bu modellerin yapısal özellikleri ve başarıları göz önünde bulundurularak, hiper-parametrelerin değer aralıkları daraltılır ve daha hedeflenmiş bir arama yapılması sağlanır. Böylece modelin optimize edilmesi sürecinde daha az sayıda ama daha doğru hiper-parametre kombinasyonları denenmiş olur. Bu tez çalışmasında belirlenen arama uzayı kısıtları şu şekildedir:

- Özellik haritası boyutunun erkenden küçültülmesini önlemek için adım hiper-parametresi, evrişim katmanında “1” ve havuzlama katmanında “2” olmalıdır (Gülcü ve Kuş, 2020).
- Aynı evrişim bloğu içerisindeki evrişim katmanları, aynı hiper-parametre değerlerini kullanmalıdır (Tian vd., 2021).
- Dolgu hiper-parametresi, tüm evrişim ve havuzlama katmanları için “aynı” olarak seçilmelidir (Springenberg vd., 2014).
- Bir evrişim katmanı için seçilen filtre boyutu, bir önceki evrişim bloğundaki filtre boyutundan küçük veya ona eşit olmalıdır (Simonyan ve Zisserman, 2015).
- Bir havuzlama katmanı için seçilen budama oranı, önceki evrişim bloğunda bulunan havuzlama katmanındaki budama oranına eşit veya daha büyük olmalıdır (Hinton, 2012). Budama oranı en fazla 0.5 olmalıdır. Birinci evrişim katmanı için budama oranının 0.2 olarak seçilmesi başarıyı artırmaktadır (Wu ve Gu, 2015b).
- Tam bağlı katmandaki nöron sayısı, önceki tam bağlı katmandaki nöron sayısına eşit veya iki katına ayarlanmalıdır (Park ve Kwak, 2017).
- Tam bağlı katman için seçilen budama oranı, önceki tam bağlı katmanda seçilen budama değerinden büyük veya ona eşit olmalıdır. İlk tam bağlı katman için budama oranını 0.2 olarak kullanmak başarılı sonuçlar vermektedir (Wu ve Gu, 2015a).

Tablo 2’de verilen hiper-parametreler ve değer aralıkları ($S = \llbracket S_{min}, S_{max} \rrbracket$), derin öğrenme modelinin farklı katmanlarında hangi parametrelerin ayarlanabileceğini ve bu

parametrelerin hangi aralıklarda değişkenlik gösterebileceğini göstermektedir. Bu parametre değerlerine herhangi bir kısıtlama getirilmediğinde modelin mimarisi büyük ve karmaşık hale gelmektedir. Modelin çok sayıda parametreye sahip olması eğitim süresinin gereksiz yere uzamasına yol açmaktadır. Özellikle büyük arama uzayları, modelin optimizasyon sürecini yavaşlatırken, daha fazla hesaplama kaynağı gerektirmektedir. Bu sebeple hiper-parametre arama uzayı önceki çalışmalar dikkate alınarak ayırık değerli olarak sınırlandırılmıştır.

Tablo 2. Hiper-parametreler ve önerilen arama uzayı

Katman	Hiper-Parametre	Değer Aralıkları
Evrişim	<i>blok sayısı</i>	1 - 5
	<i>katman sayısı</i>	1 - 7
	<i>filtre boyutu</i>	2, 3, 5, 7, 9
	<i>özellik haritası sayısı</i>	16, 32, 64, 128, 160, 192, 224, 256, 512
	<i>adım büyüklüğü</i>	1, 2
	<i>dolgu</i>	SAME, VALID
Havuzlama	<i>filtre boyutu</i>	2, 3, 4, 5
	<i>havuzlama yöntemi</i>	MAX, AVG
	<i>adım büyüklüğü</i>	1, 2
	<i>dolgu</i>	SAME, VALID
Tam Bağlı	<i>katman sayısı</i>	1 - 10
	<i>nöron sayısı</i>	64, 128, 256, 512, 1024
Eğitim Süreci	<i>öğrenme oranı</i>	0.001, 0.01, 0.1
	<i>aktivasyon fonksiyonu</i>	ReLU, Leaky-ReLU, Elu
	<i>yığın boyutu</i>	16, 32, 64, 128
	<i>yığın normalizasyonu</i>	True, False
	<i>optimizasyon yöntemi</i>	SGD, Adam, RMSprop
Düzenleştirme	<i>budama oranı</i>	0.0, 0.2, 0.3, 0.4, 0.5
	<i>düzenleştirme yöntemi</i>	L_1 , L_2
	<i>L_1-L_2 katsayıları</i>	0.0, 0.01, 0.001, 0.0001
Ağırlıklar	<i>ağırlık başlatma</i>	HeNormal, GlorotUniform, RandomNormal

Hiper-parametrelerin rastgele seçilmesi, genellikle modelin performansını olumsuz yönde etkileyen faktörlerden biridir. Örneğin, evrişim katmanlarında filtre boyutunun

başlangıçta çok büyük seçilmesi, giriş boyutunun hızlı bir şekilde küçülmesine ve bu durumun sonucunda veri kaybına neden olmaktadır. Bu durumda, model önemli özellikleri kaybedebilir ve genel performansı düşebilir. Benzer şekilde, adım parametresinin büyük seçilmesi, modelin ayrıntılı bilgi yakalamasını engelleyebilir ve bu da sınıflandırma ya da nesne algılama gibi görevlerde hatalı sonuçlar verebilir. Filtre boyutu küçük seçilirse, model daha ayrıntılı ve hassas özellikleri yakalayabilir, ancak bu durum aynı zamanda daha fazla katman gerektirdiğinden işlem maliyetini artırabilir ve geniş bağlam bilgisinin eksik kalmasına neden olabilir. Adım boyu küçük seçilirse, model daha yüksek çözünürlüklü ve detaylı bir özellik haritası üretebilir, ancak bu da işlem süresini uzatabilir ve gereksiz bilgi fazlalığına yol açabilir. Bu yüzden filtre boyutu ve adım boyunun dengeli bir şekilde seçilmesi, model performansını optimize etmek için gereklidir.

Güncel ESA mimarileri tasarlanırken başlangıçta özellik harita sayıları küçük ve filtre boyutları büyük seçilir. Daha sonra özellik harita sayıları artırılarak ve filtre boyutları azaltılarak kademeli olarak başarılı ESA modelleri aranır. Bu yaklaşım, derin ağların daha etkili bir şekilde öğrenmesini ve bilginin yoğun olduğu bölgelerde daha hassas olmasını sağlamaktadır. Özellikle derinlik artarken özellik haritası sayısının artması, modelin daha fazla özellik çıkarabilmesine olanak tanımaktadır. Filtre boyutunun küçülmesi ise detayların daha ince bir şekilde yakalanmasına yardımcı olmaktadır. Ayrıca, her ikisinin aynı anda artması veya azalması, arama sürecinin aşamalarında çok farklı maliyetlere sebep olabilir. Özellik haritası sayısının artırılması modelin kapasitesini artırırken işlem maliyetini yükseltir. Filtre boyutunun küçülmesi ise daha fazla işlem ve veri yoğunluğu gerektirir. Bu değişikliklerin dengeli bir şekilde yapılması, gereksiz bilgi işlenmesini önler ve bu sayede işlem maliyeti de azalmış olur.

Bu tez çalışmasında önceki çalışmalarda keşfedilen ve literatürde yer alan genel kurallar dikkate alınmakta, ancak aynı zamanda daha verimli bir arama süreci hedeflenmektedir. Arama uzayını keşfederken daha az parametre ile çalışan ve performansı büyük modellerle rekabet edebilecek seviyede olan modeller geliştirmek amaçlanmaktadır. Bu hedef doğrultusunda, daha hızlı cevap sürelerine sahip, optimize edilmiş modeller elde edilmeye çalışılmaktadır. Örneğin, filtre boyutlarının başlangıçta küçük tutulması ve özellik haritası sayısının aşamalı olarak artırılması gibi stratejiler, modelin daha hızlı ve daha verimli bir şekilde eğitilmesine olanak sağlamaktadır. Böylelikle, büyük ve karmaşık modellerin sunduğu performans seviyelerine yaklaşan, ancak daha az hesaplama kaynağı

gerektiren modeller üretilebilir. Bu modeller özellikle gerçek zamanlı uygulamalarda ve sınırlı kaynaklara sahip sistemlerde önemli avantajlar sunabilmektedir.

2.3. Birey Kodlama Stratejisi

Birey, ÇAOA'nın çözüm adayı, yani ESA mimarisinin kendisidir. Çalışmada, ESA mimarisine uygun bir birey kodlama stratejisi sunulmuştur. Önerilen birey kodlama stratejisinde dinamik olarak genişleyebilen veya küçülebilen bir mimari benimsenmiştir. Bu mimari sayesinde modelin karmaşıklığı eğitim sürecinin ihtiyacına göre uyarlanabilmektedir. Bu tür mimariler farklı veri kümeleri veya hesaplama kaynaklarına uygun şekilde esneklik sağlayarak, ESA modellerinin otomatik tasarlanmasını sağlamaktadır.

Birey kodlama stratejisi blok yapısına dayanır. Blok yapısı, belirli işlemleri bir araya getirerek, modelin daha modüler bir şekilde tasarlanmasını sağlamaktadır. Bu sayede her bir bloğun içindeki işlemler birlikte optimize edilebilir ve eğitim sırasında esneklik sağlanabilir.

$$\begin{aligned}
 X_{birey} &= [X_{\ddot{O}OB}, X_{SB}] \\
 , X_{\ddot{O}OB} &= [X_{EB}, HK, B] \cdot m \\
 , X_{EB} &= [EK, AF, YN] \cdot n \\
 , X_{SB} &= [TK, AF, YN, B] \cdot k \\
 , n_{min} &\leq n \leq n_{max}, m_{min} \leq m \leq m_{max}, k_{min} \leq k \leq k_{max}
 \end{aligned} \tag{21}$$

Denklem 21'de birey (X_{birey}) m adet özellik öğrenme bloğu ($X_{\ddot{O}OB}$) ve k adet sınıflandırma bloğunun (X_{SB}) dinamik bir şekilde birleştirilmesiyle oluşur. Buradaki n , m , k blok veya katmanların tekrar sayısıdır. Bu kısıtlar Tablo 2'de de gösterildiği gibi n , evrişim katman sayısı ($n_{min} = 1$, $n_{max} = 7$), m , evrişim blok sayısı ($m_{min} = 1$, $m_{max} = 5$) ve k , tam bağlı katman sayısı ($k_{min} = 1$, $k_{max} = 10$) olarak sınırları önceden belirlenmiştir.

Özellik öğrenme bloğu ($X_{\ddot{O}OB}$), evrişim katmanı (EK), aktivasyon fonksiyonu (AF), yığın normalizasyonu (YN), havuzlama katmanı (HK) ve budama (B) katmanlarından oluşmaktadır. Bu blok, modelin giriş verilerinden özellik çıkarmak ve bu özellikleri daha yüksek seviyelere taşımak için kullanılır. Her bir özellik öğrenme bloğu içerisinde, n adet evrişim bloğu (X_{EB}), bir adet havuzlama katmanı (HK) ve bir adet budama (B) bulunacak şekilde düzenlenmiştir. Özellik öğrenme bloğunun bir alt bileşeni olan evrişim bloğu (X_{EB})

üç ana bileşenden oluşur. Evrişim katmanı (*EK*), giriş verisinden yerel özellikleri çıkarır. Aktivasyon fonksiyonu (*AF*), özelliklerin doğrusal olmayan ilişkilerini öğrenmek için kullanılır. Çoğunlukla ReLU, veya türevleri kullanılır. Yığın normalizasyonu (*YN*), modelin her bir yığında hesaplanan istatistiklere göre verileri normalleştirerek eğitim sürecini hızlandırır ve stabil hale getirir.

Modelin sınıflandırma kısmı ise tam bağlı katman (*TK*), aktivasyon fonksiyonu (*AF*), yığın normalizasyonu (*YN*) ve budama (*B*) yapılarının bir araya getirilmesiyle oluşturulmuş olan sınıflandırma bloğundan (X_{SB}) oluşur. Bu blok, özellik öğrenme bloğu tarafından çıkarılan özelliklerin bir sınıfa atanması için kullanılır. Özellikle tam bağlı katmanlar, girişten gelen tüm bilgileri işleyerek sınıflandırma işlemini gerçekleştirir. Yığın normalizasyonu bu katmanda da eğitim sürecini hızlandırırken, budama işlemi ise fazla bağlantıların eğitim sırasında ortadan kaldırılmasına ve modelin daha hafif hale getirilmesine yardımcı olur.

2.4. Başlangıç Popülasyonun Oluşturulması

ÇESA yönteminde başlangıç popülasyonunun oluşturulma süreci, bireylerin önceden belirlenen hiper-parametre aralıkları ve sınırları doğrultusunda rastgele oluşturulmasıyla başlar. Bu süreç algoritmanın başlangıç aşamasında çözüm adaylarının çeşitli ve etkili bir şekilde dağıtılmasını sağlar. Her bir birey aslında farklı ESA modelini temsil eder ve arama uzayının geniş bir bölümünü keşfetmeye yönelik olarak yapılandırılır. Bireyler oluşturulurken vektörler dinamik olarak birbirine eklenir. Dinamik bir şekilde oluşan bireyler birleşerek başlangıç popülasyonunu oluşturur.

Algoritma 3’de önerilen ÇESA yönteminin p büyüklüğünde başlangıç popülasyonu (P_0) oluşturma süreci gösterilmektedir. Her birey m adet özellik öğrenme bloğu ($X_{ÖB}$) ve k adet sınıflandırma bloğu (X_{SB}) olmak üzere iki ana bileşenden oluşur. Bireylerin yapısını oluşturan bu blokların sayısı, arama uzayında (n_{min} , n_{max} , m_{min} , m_{max} , k_{min} , k_{max}) belirlenen minimum ve maksimum değerler arasında rastgele seçilen tam sayılarla belirlenir. Bu şekilde oluşturulan özellik öğrenme ve sınıflandırma blokları, her bireyin farklı bir ESA mimarisine sahip olmasını sağlar. Bu blok mimarisinde ESA modelleri (n , m , k parametreleri sayesinde) birbirinden bağımsız olarak dinamik bir şekilde genişleyip küçülebilir. Böylece aynı popülasyon içinde farklı derinliklerdeki modellerin olması sağlanır. Bu durum

çeşitliliğin artırılmasını ve optimizasyon sürecinin daha geniş bir arama uzayında yapılmasını sağlar.

Algoritma 3. ÇESA yönteminde başlangıç popülasyonun oluşturulması

Girdi: *popülasyon büyüklüğü (p), arama uzayı ($S = \llbracket S_{min}, S_{max} \rrbracket$), n_{min} , n_{max} , m_{min} , m_{max} , k_{min} , k_{max}*

Çıktı: *başlangıç popülasyonu (P_0)*

do:

birey $\leftarrow$ boş bir dinamik vektör oluşturulur

m $\leftarrow \llbracket m_{min}, m_{max} \rrbracket$ aralığında rastgele bir tam sayı belirlenir

k $\leftarrow \llbracket k_{min}, k_{max} \rrbracket$ aralığında rastgele bir tam sayı belirlenir

X_{ÖÖB} $\leftarrow$ *m* adet özellik öğrenme bloğuna sahip bir vektör oluşturulur

X_{SB} $\leftarrow$ *k* adet sınıflandırma bloğuna sahip bir vektör oluşturulur

for (*X_{ÖÖB}* vektöründeki her *ÖÖB* için)

n $\leftarrow \llbracket n_{min}, n_{max} \rrbracket$ aralığında rastgele bir tam sayı belirlenir

X_{EB} $\leftarrow$ *n* adet evrişim bloğuna sahip bir vektör oluşturulur

for (*X_{EB}* vektöründeki her *EB* için)

EB $\leftarrow$ *S* arama uzayında (*EK*,*AF*,*YN*) hiper-parametreleri rastgele belirlenir

end for

X_{HB} $\leftarrow$ *S* arama uzayında (*HK*,*B*) hiper-parametreleri rastgele belirlenir

ÖÖB $\leftarrow [X_{EB}, X_{HB}]$

end for

for (*X_{SB}* vektöründeki her *SB* için)

SB $\leftarrow$ *S* arama uzayında (*TK*,*AF*,*YN*,*B*) hiper-parametreleri rastgele belirlenir

end for

birey $\leftarrow [X_{ÖÖB}, X_{SB}]$

P₀ $\leftarrow \begin{bmatrix} P_0 \\ birey \end{bmatrix}$

while ($|P_0| < p$)

ÖÖB'ler ESA'nın öğrenme yeteneğini artıran ve evrişim işlemlerini barındıran yapılardır. ÖÖB'ler rastgele tam sayı olarak belirlenen n adet evrişim bloğu (X_{EB}) içerir. Her evrişim bloğu, evrişim katmanı (EK), aktivasyon fonksiyonu (AF) ve yığın normalizasyonu (YN) birleşiminden oluşur. Bu blokta bulunan hiper-parametreler arama uzayında ($S = \llbracket S_{min}, S_{max} \rrbracket$) belirlenen kısıtlar doğrultusunda rastgele olarak belirlenir. ÖÖB'ler evrişim bloğunun devamında, havuzlama katmanı (HK) ve budama (B) işleminin birleşmesiyle oluşturulur.

SB'ler ise sınıflandırma işlemini gerçekleştiren katmanlardan oluşur. SB, tam bağlı katman (TK), aktivasyon fonksiyonu (AF), yığın normalizasyonu (YN) ve budama (B) yapı taşlarını içerir. Bu blokta bulunan hiper-parametreler arama uzayında ($S = \llbracket S_{min}, S_{max} \rrbracket$) belirlenen kısıtlar doğrultusunda rastgele olarak belirlenir. Bu blok ESA'nın en son kısmına yerleşir ve modelin nihai sınıflandırma performansını belirler.

Sonuç olarak her bir birey, rastgele sayıda EB içeren rastgele sayıda ÖÖB ile rastgele sayıda SB bloğundan oluşur. Bireyler bu dinamik yapı sayesinde farklı derinliklerde ve yapılarda ESA modellerini temsil eder. Popülasyon, belirlenen popülasyon büyüklüğü (p) kadar bireyin bir araya gelmesiyle oluşur. Bu popülasyon, çeşitliliği yüksek bireylerden oluştuğu için arama uzayında farklı çözümleri keşfetme potansiyeli fazladır. Böylece ÇESA algoritması, başlangıç popülasyonunda farklı çözümlerle çalışarak, daha etkili ve hızlı bir şekilde optimum çözüme ulaşma şansını artırmaktadır.

2.5. Uygunluk Değerlendirme

Uygunluk değerlendirme fonksiyonları, popülasyondaki bireylerin kalitesini yani çözüm uygunluğunu belirlemek için kullanılır. Bu fonksiyonlar, her bireyin (yani potansiyel çözümün) ne kadar iyi performans gösterdiğini nicel olarak ifade eder. Böylece optimizasyon algoritması daha iyi performans gösteren bireyleri seçerek sonraki nesillere aktarır. Algoritmanın yapısına göre, daha az uygun olanları eler veya değiştirir. Bu süreç genel olarak algoritmanın daha iyi çözümlere doğru evrilmesini sağlar. Uygunluk değerlendirme fonksiyonlarının doğru ve etkili bir şekilde tanımlanması, optimizasyon sürecinin başarısı için kritik öneme sahiptir.

$$F_1(x) = \text{doğruluk oranı} = \frac{\text{doğru tahmin sayısı}}{\text{bütün tahmin sayısı}} \quad (22)$$

$$F_2(x) = \text{cevap süresi} = \text{modelin bir girdiyi sınıflandırma süresi}$$

Denklem 22’de ESA modellerinin, doğruluk ve yanıt süresi performanslarını ölçmek ve optimize etmek amacıyla iki temel uygunluk değerlendirme fonksiyonu belirlenmiştir.

- **Doğruluk Oranı:** Bu metrik, modelin sınıflandırma görevindeki başarısını nicel olarak ölçer. Doğruluk oranı, modelin test veri setindeki örneklerin ne kadarını doğru bir şekilde sınıflandırdığını gösterir.
- **Cevap Süresi:** Bu metrik, modelin bir girdi verisini işleyip tahmin yapması için geçen süreyi ölçer. Özellikle gerçek zamanlı uygulamalarda, modelin hızlı bir şekilde yanıt verebilmesi büyük önem taşır. Uzun cevap süreleri, uygulamanın kullanılabilirliğini ve etkinliğini azaltabilir. Bu nedenle cevap süresi düşük olan modeller tercih edilmektedir.

Denklem 22’de ifade edilen bu iki uygunluk değerlendirme fonksiyonu, modelin performansını hem etkinlik (doğruluk oranı) hem de verimlilik (cevap süresi) açısından değerlendirmeyi amaçlar. Yüksek doğruluk oranına sahip bir model, verileri doğru bir şekilde sınıflandırabilse de, eğer cevap süresi uzunsa özellikle gerçek zamanlı uygulamalarda kullanımı pratik olmayabilir. Benzer şekilde, çok hızlı yanıt süresine sahip bir model düşük doğruluk oranına sahipse, elde edilen sonuçların güvenilirliği sorgulanabilir. Optimizasyon sürecinde her iki metriği de dikkate alarak hem yüksek doğruluk oranına sahip hem de hızlı yanıt verebilen modellerin geliştirilmesi hedeflenmektedir.

Bu çalışmada, model uygunluklarının değerlendirilmesi için 2 amaç fonksiyonu aynı anda optimize edilmiştir. Önerilen ÇESA yönteminin dinamik yapısı sayesinde, duruma göre daha fazla amaç fonksiyonunun dengelenmesine çalışılabilir.

2.6. ÇAOA: Çok Amaçlı Optimizasyon Algoritması

Çok amaçlı optimizasyon algoritması (ÇAOA), birden fazla amaç fonksiyonunu aynı anda optimize etmek için kullanılır. Bu algoritmanın amacı yalnızca tek bir hedefi optimize etmek yerine çeşitli hedefler arasında denge kurarak çözüm kümesindeki her bireyin farklı açılardan optimum olmasını sağlamaktır. Dinamik yapısı sayesinde farklı çok amaçlı optimizasyon algoritmaları çalışmada önerilen ÇESA ile entegre edilebilir. Bu esneklik farklı optimizasyon stratejilerinin kullanılmasına imkan tanıyarak, ÇESA’nın çeşitli problem yapılarında ve farklı ÇAOA ile etkin bir şekilde çalışabilmesini sağlamaktadır.

Algoritma 4. NSGA-II algoritmasının oluşturulan sözde kodu (Deb vd., 2002)

Girdi: *popülasyon büyüklüğü (p), maksimum iterasyon sayısı (G)*

Çıktı: *pareto optimal set (P^*)*

→ *p büyüklüğünde P_0 popülasyonu oluşturulur*

→ *nesildeki tüm bireylerin (P_g) uygunluk değeri hesaplanır*

for (her nesil g için 1 'den G 'ye kadar)

P_g nesili baskınlıklarına göre sıralanır

P_g sıralanmış popülasyona kalabalık mesafesi uygulanır

$M \leftarrow$ boş bir eşleşme havuz vektörü oluşturulur

do:

ikili turnuva seçimi kullanılarak ebeveynler seçilir

çaprazlama ve mutasyon operatörleriyle yeni bireyler oluşturulur

bütün yeni bireyler eşleşme havuzuna (M) eklenir

while ($|M| < p$)

eşleşme havuzundaki tüm bireylerin (M) uygunluk değeri hesaplanır

$Q \leftarrow P_g \cup M$

$FS \leftarrow Q$ popülasyonu *baskınlıklarına göre sıralanır*

$FS \leftarrow$ *kalabalık mesafesi uygulanarak güncellenir*

$P^* \leftarrow FS_1$

yeni nesil $P_{(g+1)}$ boş bir popülasyon olarak başlatılır

$i \leftarrow 1$

do:

$P_{(g+1)} \leftarrow P_{(g+1)} \cup FS_i$

$i \leftarrow i + 1$

while ($|P_{(g+1)}| + |FS_i| < p$)

end for

return P^*

Bu çalışmada, optimizasyon sürecini yönetmek amacıyla popüler ve etkili bir çok amaçlı optimizasyon algoritması olan Baskın Olmayan Sıralı Genetik Algoritma-II (Non-dominated Sorting Genetic Algorithm-II, NSGA-II) tercih edilmiştir. NSGA-II pareto

optimal çözümleri hızlı ve etkili bir şekilde bulmayı hedefler. Hesaplama verimliliği ile öne çıkan bu algoritma, literatürde yaygın olarak kullanılması ve teorik sağlamlığı sayesinde tez çalışmasında tercih edilmiştir. Farklı performans metriklerini optimize etme gereksinimi olan ESA'nın çok amaçlı optimizasyon probleminde, NSGA-II'nin pareto optimal çözümler oluşturma ve çözüm uzayını etkin bir şekilde keşfetme yeteneği bu seçimde etkili olmuştur.

Algoritma 4'de ilk olarak rastgele bir başlangıç popülasyonu (P_0) oluşturulur. Popülasyon (P_g) pareto dominansına göre sıralanır. Her çözümün rank değeri belirlenir ve çözümler öncelik seviyelerine (FS) ayrılır. Kalabalık mesafesi metriği hesaplanarak çözümler arasındaki çeşitlilik korunur. Turnuva seçimi kullanılarak ebeveynler seçilir. Turnuva seçimi daha iyi çözümlerin daha yüksek bir seçilme olasılığına sahip olmasını sağlar. Çaprazlama ve mutasyon operatörleri kullanılarak yeni bireyler (M) oluşturulur. Yeni bireylerin uygunlukları hesaplanır ve mevcut popülasyonla birleştirilir. Bu birleşik popülasyon (Q), pareto dominansına göre yeniden sıralanır ve en iyi çözümler seçilir. Elitizm prensibi ile, en iyi çözümler bir sonraki nesle ($P_{(g+1)}$) doğrudan aktarılır. Bu, çözümlerin kalitesinin zaman içinde azalmasını önler. Bu adımlar (g), önceden belirlenen nesil sayısına (G) ulaşılan kadar tekrarlanır. İterasyon tamamlandığında algoritma bulduğu en iyi çözüm kümesini (P^*) sonuç olarak döndürür.

Bu çalışmada, NSGA-II algoritmasının UÇD yöntemi ile geliştirilerek yeni bir versiyonu da algoritma olarak önerilmiştir. Geliştirilen FDB-NSGA-II algoritması, birey seçim sürecinde klasik turnuva seçim yöntemi yerine UÇD yöntemi kullanır. Bu yöntemle geliştirilen algoritma, uygunluk değeri ve uygunluk değer uzayında bireyler arasındaki mesafeyi dikkate alarak daha dengeli ve çeşitli çözümler elde etmeyi amaçlar.

2.6.1. Baskın Olmayan Sıralama

Pareto tabanlı ÇAO algoritmalarında, popülasyonu sıralamak için baskın olmayan sıralama yöntemi kullanılır. Bu yöntem çözümleri birden fazla öncelik seviyesine ayırarak, her seviye için baskın olmayan çözümler kümesini belirler. Pareto dominansı, bir çözümün başka bir çözümden daha iyi performans gösterip göstermediğini değerlendiren bir ölçüttür. Bir çözüm, başka bir çözümü, tüm amaç fonksiyonlarında eşit ya da daha iyi sonuçlar verdiğinde ve en az bir amaç fonksiyonunda daha iyi performans gösterdiğinde baskılar (domine eder). Bu baskınlık ilişkilerine göre çözümler sıralanır ve baskın olan çözümler

pareto optimal olarak kabul edilir. Bu süreç, tüm çözümler baskınlıklarına göre sıralanana kadar devam eder.

Algoritma 5'te baskın olmayan sıralama yöntemi sözde kodu verilmektedir. Mevcut popülasyon (P) üzerinde doğrudan işlem yapmamak için, popülasyonu başka bir matrise (Q) aktararak başlanır. Bu işlem orijinal popülasyonun korunmasını sağlar ve sıralama işlemi Q popülasyonu üzerinde gerçekleştirilir. Q popülasyonu işlenmek üzere bekleyen bireylerin bulunduğu geçici bir matristir ve sıralama işlemi ilerledikçe bireyler bu matristen çıkarılır.

Algoritma 5. Baskın olmayan sıralama yöntemi sözde kodu

Girdi: *popülasyon (P)*
Çıktı: *pareto cephesi FS , $FS = \{FS_1, FS_2 \dots\}$*
 $Q \leftarrow P$
do:
 $i \leftarrow 1$
 for (her birey p için tüm bireyler (Q))
 $adet \leftarrow 0$
 for (her birey q için tüm bireyler (Q))
 if q, p 'yi baskılar **then:**
 $adet \leftarrow adet + 1$
 end if
 end for
 if $adet > 0$ **then:**
 $FS_i \leftarrow FS_i \cup \{p\}$
 $Q \leftarrow Q \setminus \{p\}$
 end if
 end for
 $i \leftarrow i + 1$
while ($|Q| < 1$)
return FS

Her birey p için popülasyonun diğer tüm bireyleri q ile baskınlık kontrolü yapılır. Bu kontrol bireyin pareto dominansına göre sıralanabilmesi için gerekli olan en önemli adımdır. Eğer bir birey (q), diğer birey (p) üzerinde baskınsa, yani tüm amaç fonksiyonlarında daha iyi performans gösteriyorsa, baskınlık sayacı ($adet$) artırılır. Ancak birey (p) baskınlık

açısından diğer bireylerden daha iyi ya da onlara eşitse, baskın olmayan bir çözüm olarak değerlendirilir. Eğer bir birey (p) üzerinde baskınlık kuran başka bir birey yoksa, bu birey ilk pareto seviyesine (rank 1) ait olarak kabul edilir ve FS_1 kümesine dahil edilir. FS_1 en baskın çözümlerden oluşan pareto optimal setidir ve sıralamada öncelik seviyesine sahiptir. Birey bu setin bir parçası olduktan sonra Q popülasyonundan çıkarılır.

İlk seviyedeki baskılanamayan bireyler (rank 1) belirlendikten sonra algoritma ikinci seviyeye (rank 2) dahil olacak bireyleri belirler. İkinci seviyede rank 1'deki bireyler hariç diğer bireyler arasında yeniden baskınlık kontrolü yapılır ve bu bireyler rank 2 setine yerleştirilir. Bu süreç tüm bireyler sıralanana kadar tekrarlanır ve her yeni pareto seti oluşturuldukça, bir önceki seviyede çıkarılan bireyler dikkate alınmaz. Böylece popülasyon katman katman baskılanamayan bireylere göre sıralanır. Algoritma, Q popülasyonunda sıralanacak birey kalmadığında sona erer. Tüm bireyler pareto dominansına göre sıralandıktan sonra elde edilen setler ($FS = \{FS_1, FS_2, \dots\}$) çıkış olarak döndürülür. FS_1 en baskın çözümlerden oluşan pareto optimal seti temsil ederken, FS_2 ve diğer setler sırasıyla daha düşük rütbeli çözümleri içermektedir.

2.6.2. Kalabalık Mesafesi

Kalabalık mesafesi (KM), ÇAO için çözümler arasında çeşitliliği korumak ve pareto cephesinde daha dengeli bir dağılım sağlamak için kullanılan uzaklık ölçütü metriğidir. Bu yöntem çözümler arasındaki mesafeyi hesaplayarak, çözümlerin birbirine ne kadar yakın olduğunu belirler. Böylece birbirine çok yakın çözümler arasında seçim yapılırken, daha geniş bir dağılıma sahip olan çözümler tercih edilir.

Algoritma 6'da kalabalık mesafesi yöntemi sözde kodu verilmektedir. KM değerinin hesaplanacağı popülasyon (P), baskın olmayan sıralama yöntemiyle sıralanmış bireylerden oluşur. Her bireyin baskın olmayan çözümler arasında hangi seviyede yer aldığı önceden belirlenmiştir. Her birey için kalabalık mesafesi metriği başlangıç değeri sıfır olarak atanır ($KM_p \leftarrow 0$). Bu, bireylerin mesafelerinin henüz hesaplanmadığını ve sıfırdan başlayarak güncelleneceğini ifade eder. Her bir amaç fonksiyonu (f) için popülasyondaki bireyler, her bir amaç fonksiyonu açısından uygunluklarına göre artan sırada sıralanır. Böylece bireylerin performansları karşılaştırılabilir. Her amaç fonksiyonu için sıralanan popülasyonun uç noktalarında yer alan çözümler (KM_{ilk} ve KM_{son}), belli bir amaç fonksiyonu açısından en iyi ve en kötü performansı gösteren çözümler olduğu için, bu çözümler sonsuz değerine sahip

olarak atanır. Bu çözümler popülasyonun en uç noktalarını temsil eder ve diğer çözümlerle kıyaslanamayacak kadar uzakta olduğu kabul edilir. Bu çözümlerin korunması pareto cephesinde uç noktalarda çeşitliliğinin korunmasını sağlar.

Algoritma 6. Kalabalık mesafesi yöntemi sözde kodu

Girdi: *popülasyon (P), amaç fonksiyonları (F)*
Çıktı: *kalabalık mesafesi (KM)*
for (her birey p için tüm bireyler (P))
 $KM_p \leftarrow 0$
end for
for (her amaç fonksiyonu f için tüm amaç fonksiyonları (F))
 P popülasyonu f amaç fonksiyonuna göre artan sırayla sıralanır
 $KM_{ilk} \leftarrow \text{sonsuz}$
 $KM_{son} \leftarrow \text{sonsuz}$
 for (her birey p için (ilk ve son birey hariç) tüm bireyler (P))
 $KM_p \leftarrow KM_p + (f_{p+1} - f_{p-1}) / (f_{max} - f_{min})$
 end for
end for
return KM

Algoritmada uç noktaların dışındaki çözümler için KM , komşu çözümler ($p+1$ ve $p-1$) arasındaki mesafe hesaplanarak belirlenir. Bu işlemde her bireyin yakınındaki komşu çözümlere göre ne kadar yakın olduğu hesaplanır. Bir bireyin KM değeri, kendisinden önceki (f_{p+1}) ve sonraki (f_{p-1}) bireylerin performans değerleri arasındaki farkın normalize edilmesiyle bulunur. Hesaplanan bu fark, amaç fonksiyonunun maksimum değerleri arasındaki fark ile normalize edilerek KM değeri güncellenir. Bu süreç her birey için her amaç fonksiyonu açısından gerçekleştirilir. Her birey için hesaplanan KM değerleri, her amaç fonksiyonu için elde edilen mesafelerin toplamı şeklinde birikir. Böylece her birey için toplam bir KM değeri elde edilir.

KM hesaplaması sırasında amaç fonksiyonlarının normalize edilmesi, farklı ölçeklerdeki fonksiyonlar arasında adil bir karşılaştırma yapılmasını sağlar. Popülasyondaki çözümler yalnızca performans açısından değil, aynı zamanda uzayda ne kadar yayıldıkları açısından da değerlendirilir. Sonuç olarak KM değeri yüksek olan bireyler, daha fazla

çeşitlilik sunarak algoritmanın daha geniş bir çözüm uzayını keşfetmesine olanak tanır ve böylece nihai pareto ön cephesinin daha dengeli bir şekilde oluşmasını sağlar.

2.7. Önerilen ÇESA Modelleri

Çalışmada ÇESA yöntemi ile 4 farklı model oluşturulmuştur. Model oluşturulurken kullanılan veri setleri CIFAR-10, CIFAR-100, MNIST iken, çok amaçlı optimizasyon algoritması olarak NSGA-II ile çalışmada geliştirilen FDB-NSGA-II kullanılmıştır. Bu veri seti ve algoritma kombinasyonları kullanılarak ÇESA ile tasarlanmış DÖ modelleri şunlardır:

- **ÇESA-1:** CIFAR-10 + NSGA-II
- **ÇESA-2:** CIFAR-100 + NSGA-II
- **ÇESA-3:** MNIST + NSGA-II
- **ÇESA-4:** CIFAR-10 + FDB-NSGA-II

Önerilen ÇESA modellerinde ÇESA-1 ve ÇESA-4 17 katmandan ÇESA-2 10 ve ÇESA-3 8 katmandan oluşmaktadır. Tablo 3'te kullanılan katman sayıları ve toplam parametre sayıları verilmiştir.

Tablo 3. ÇESA modellerindeki katman sayıları

Katman Adı	ÇESA-1	ÇESA-2	ÇESA-3	ÇESA-4
Conv2D	4	3	2	3
BatchNormalization	4	0	0	4
Dropout	3	1	1	4
MaxPooling2D	3	3	2	3
Flatten	1	1	1	1
Dense	2	2	2	2
Toplam Katman Sayıları	17	10	8	17
Toplam Parametre Sayıları	187402	878404	225610	1740234

DÖ modellerindeki katmanlardan *Conv2D*, giriş verisinden önemli özellikleri (örneğin, bir görüntüdeki kenarlar veya dokular gibi) çıkarmak için kullanılır. *BatchNormalization* katmanı, modelin daha hızlı ve kararlı öğrenmesi için giriş verilerini

normalleştirir. *Dropout*, aşırı öğrenmeyi önlemek için bazı nöronları rastgele devre dışı bırakır. Bu katmanlar modelin genelleme kabiliyetini artırmak için kritik öneme sahiptir ve modelin eğitim verisine aşırı uyum sağlamasını engeller. Havuzlama katmanları, özellik haritalarının boyutunu azaltarak hesaplama yükünü hafifletir ve modele uzamsal değişikliklere karşı dayanıklılık kazandırır. *MaxPooling2D* katmanı her özellik haritasından maksimum değeri seçerek bilgi kaybını en aza indirir. *Flatten*, çok boyutlu veriyi tek boyutlu vektöre dönüştürür. *Dense* tam bağlı katman olup, giriş verilerini işleyerek çıkış bilgisini oluşturur.

Katmanlardaki parametre sayısı, katmanlarda gerçekleştirilen grup işlem sayısıdır. Tablo 4-7’de gösterilen katman boyutu temsiline göre parametre sayısı şu şekilde hesaplanır:

- *Conv2D* katmanı parametre sayısı: filtre boyutları, giriş özellik haritası sayısı ve filtre sayısının çarpımına çıkış özellik haritası sayısının eklenmesiyle hesaplanır.
- *BatchNormalization* katmanı parametre sayısı: Çıkışına bağlı olduğu *Conv2D* katmanının, çıktı sayısının 4 (*Beta*, *Gamma*, *Mean*, *Variance*) ile çarpılmasıyla hesaplanır.
- *Dense* katmanı parametre sayısı: girdi sayısı ile nöron sayısı çarpımına çıktı sayısının eklenmesiyle hesaplanır.
- *MaxPooling2D* katmanında, çalışmada işlem yapılmadan maksimum değer seçildiği için parametre sayısı sıfır çıkmaktadır.
- *Dropout* katmanında eleme yapıldığı için parametre sayısı sıfır olmaktadır.
- *Flatten* katmanında, özellik haritaları tek bir vektöre çevrildiği için parametre sayısı sıfırdır.

2.7.1. ÇESA-1 Modeli

ÇESA-1 modeli ÇESA yaklaşımı ile NSGA-II algoritması kullanılarak CIFAR-10 veri setinde üretilmiştir. Modelde yer alan evrişim, havuzlama, yığın normalizasyonu, budama, düzleştirme ve tam bağlı katmanlar, görüntü sınıflandırma görevlerinde yüksek doğruluk ve hızlı cevap süresi sağlamak için çok amaçlı optimize edilmiştir. Üretilen ÇESA-1 modelinin her bir katmanı ve bu katmanların işlevleri Tablo 4’te gösterilmektedir. 17 katmandan oluşan bu model 187,402 parametre içermektedir.

Birinci *Conv2D* katmanında (3, 3, 3, 32) şeklinde bir filtre yapısı görülmektedir. Bu, 3x3 boyutunda ve 3 kanallı (RGB) bir çekirdeğin, 32 farklı filtre ile çalıştığını gösterir.

Sonraki *Conv2D* katmanları giderek daha derin özellikler çıkaracak şekilde yapılandırılmıştır ve filtre sayısı artırılarak modelin daha karmaşık özellikleri öğrenmesi sağlanır. *BatchNormalization*, girişe uygulanan normalizasyon parametrelerinin boyutu belirtilmiştir. Bu katman, ağırlıkların düzenli bir şekilde güncellenmesine ve modelin aşırı öğrenmeye karşı daha dirençli olmasına yardımcı olur. Modelde *Dense* katmanı (64, 256) boyutunda, yani 64 birimden gelen girdi 256 nörona bağlanır. İkinci *Dense* katmanı ise (256, 10) boyutunda olup, son çıktı sınıflarını temsil eder (örneğin, CIFAR-10 veri seti için 10 sınıf). Tablo 4'te, önerilen ÇESA-1 modelindeki katmanlar, boyutları ve parametre sayıları olarak verilmiştir.

Tablo 4. ÇESA-1 modeli sistem mimarisi

Sıra	Katman Adı	Katman Türü	Katman Boyutu	Parametre Sayısı
1	conv2d	Conv2D	[(3, 3, 3, 32), (32)]	896
2	batch_normalization	BatchNormalization	[(32), (32), (32), (32)]	128
3	dropout	Dropout	- - -	0
4	conv2d_1	Conv2D	[(3, 3, 32, 64), (64)]	18496
5	batch_normalization_1	BatchNormalization	[(64), (64), (64), (64)]	256
6	max_pooling2d	MaxPooling2D	- - -	0
7	conv2d_2	Conv2D	[(3, 3, 64, 128), (128)]	73856
8	batch_normalization_2	BatchNormalization	[(128), (128), (128), (128)]	512
9	max_pooling2d_1	MaxPooling2D	- - -	0
10	conv2d_3	Conv2D	[(3, 3, 128, 64), (64)]	73792
11	batch_normalization_3	BatchNormalization	[(64), (64), (64), (64)]	256
12	max_pooling2d_2	MaxPooling2D	- - -	0
13	dropout_1	Dropout	- - -	0
14	flatten	Flatten	- - -	0
15	dense	Dense	[(64, 256), (256)]	16640
16	dropout_2	Dropout	- - -	0
17	dense_1	Dense	[(256, 10), (10)]	2570

2.7.2. ÇESA-2 Modeli

ÇESA-2 modeli ÇESA yaklaşımı ile NSGA-II algoritması kullanılarak CIFAR-100 veri setinde üretilmiştir. ÇESA-1 modeline benzer şekilde, evrişim, havuzlama, budama, düzleştirme ve tam bağlı katmanlar içermektedir. ÇESA-2 modelinin her bir katmanı ve bu katmanların işlevleri Tablo 5'te gösterilmektedir. 10 katmandan oluşan bu model 878,400 parametre içermektedir. ÇESA-1'e göre daha az katmanı olup daha fazla parametre içermesi bazı katmanlardaki (özellikle 8. satırdaki birinci *Dense* katmanı) yüksek veri miktarı ile fazla hesaplama yapmasından kaynaklanır.

Tablo 5. ÇESA-2 modeli sistem mimarisi

Sıra	Katman Adı	Katman Türü	Katman Boyutu	Parametre Sayısı
1	conv2d_4	Conv2D	[(3, 3, 3, 32), (32)]	896
2	max_pooling2d_3	MaxPooling2D	---	0
3	conv2d_5	Conv2D	[(3, 3, 32, 64), (64)]	18496
4	max_pooling2d_4	MaxPooling2D	---	0
5	conv2d_6	Conv2D	[(3, 3, 64, 128), (128)]	73856
6	max_pooling2d_5	MaxPooling2D	---	0
7	flatten_1	Flatten	---	0
8	dense_2	Dense	[(3200, 256), (256)]	819456
9	dropout_3	Dropout	---	0
10	dense_3	Dense	[(256, 100), (100)]	25700

Birinci *Conv2D* katmanı (3, 3, 3, 32) şeklinde bir filtre yapısına sahip olup, 3x3 boyutundaki çekirdeği 3 kanallı (RGB) giriş üzerinde çalıştırarak 32 özellik haritası çıkarır. Sonraki *Conv2D* katmanları giderek artan filtre sayısı ile daha karmaşık özellikleri öğrenir. Her *Conv2D* katmanının ardından bir *MaxPooling2D* katmanı gelmektedir. *MaxPooling2D* katmanları, özellik haritalarının boyutunu azaltarak hesaplama yükünü hafifletir ve modelin performansını artırır. *Flatten* katmanı iki boyutlu özellik haritalarını tek bir vektör haline getirir, böylece veriler tam bağlı katmanlara aktarılabilir. Birinci *Dense* katmanı (3200, 256) boyutunda olup, 3200 boyutlu girdiyi 256 nörona bağlar. İkinci *Dense* katmanı ise 100 çıkış sınıfına sahip olup, sınıflandırma işlemi için son çıktı olarak kullanılır (örneğin, CIFAR-100 veri seti için 100 sınıf).

2.7.3. ÇESA-3 Modeli

ÇESA-3 modeli ÇESA yaklaşımı ile NSGA-II algoritması kullanılarak MNIST veri setinde üretilmiştir. ÇESA-2 modeline benzer şekilde, evrişim, havuzlama, budama, düzleştirme ve tam bağlı katmanlar içermektedir. ÇESA-3 modelinin her bir katmanı ve bu katmanların işlevleri Tablo 6’da gösterilmektedir. Bu model 8 katmandan oluşup 225,610 parametre içermektedir.

Tablo 6. ÇESA-3 modeli sistem mimarisi

Sıra	Katman Adı	Katman Türü	Katman Boyutu	Parametre Sayısı
1	conv2d_7	Conv2D	[(3, 3, 3, 32), (32)]	896
2	max_pooling2d_6	MaxPooling2D	---	0
3	conv2d_8	Conv2D	[(3, 3, 32, 64), (64)]	18496
4	max_pooling2d_7	MaxPooling2D	---	0
5	flatten_2	Flatten	---	0
6	dense_4	Dense	[(1600, 128), (128)]	204928
7	dropout_4	Dropout	---	0
8	dense_5	Dense	[(128, 10), (10)]	1290

ÇESA-3 modeli iki adet *Conv2D* katmanı içerir. Birinci *Conv2D* katmanı (3, 3, 3, 32) şeklinde bir filtre yapısına sahiptir. Bu yapı 3x3 boyutundaki çekirdeklerin 3 kanallı (RGB) girişe uygulanarak 32 özellik haritası çıkarmasını sağlar. İkinci *Conv2D* katmanı 64 filtre içerir ve daha derin özellikleri öğrenir. Her *Conv2D* katmanının ardından gelen *MaxPooling2D* katmanları, özellik haritalarının boyutunu azaltarak hesaplama yükünü düşürür. *Flatten* katmanı iki boyutlu özellik haritalarını tek boyutlu bir vektöre dönüştürerek veriyi tam bağlı katmanlara giriş olarak hazır hale getirir. İlk *Dense* katmanı 1600 boyutlu giriş verisini 128 nörona bağlayarak modelin daha karmaşık özellikleri öğrenmesine olanak tanır. İkinci *Dense* katmanı modelin çıkışında yer alır ve 10 sınıf içeren bir çıktı üretir (örneğin, MNIST veri seti gibi 10 sınıflı bir veri seti için uygundur). Aşırı öğrenmeyi önlemek ve modelin genelleme kabiliyetini artırmak için bir *Dropout* katmanı eklenmiştir.

2.7.4. ÇESA-4 Modeli

ÇESA-4 modeli ÇESA yaklaşımı ile FDB-NSGA-II algoritması kullanılarak CIFAR-10 veri setinde üretilmiştir. Farklı olarak bu model üretilirken ÇAOA olarak geliştirilen FDB-NSGA-II kullanılmıştır. ÇESA-1 modeline benzer şekilde, evrişim, havuzlama, budama, yığın normalizasyonu, düzleştirme ve tam bağlı katmanlar içermektedir. ÇESA-4 modelinin her bir katmanı ve bu katmanların işlevleri Tablo 7’de gösterilmektedir. ÇESA-1 gibi 17 katmandan oluşan bu model 1,740,234 parametre ile ÇESA modelleri arasında en yüksek parametre sayısına sahiptir. Özellikle 14. katmandaki 3200 boyutlu verinin 512 adet nörona girdi olarak verilmesi parametre sayısını artırmıştır.

Tablo 7. ÇESA-4 modeli sistem mimarisi

Sıra	Katman Adı	Katman Türü	Katman Boyutu	Parametre Sayısı
1	conv2d_9	Conv2D	[(3, 3, 3, 32), (32)]	896
2	batch_normalization_4	BatchNormalization	[(32), (32), (32,), (32)]	128
3	max_pooling2d_8	MaxPooling2D	---	0
4	dropout_5	Dropout	---	0
5	conv2d_10	Conv2D	[(3, 3, 32, 64), (64)]	18496
6	batch_normalization_5	BatchNormalization	[(64), (64), (64), (64)]	256
7	max_pooling2d_9	MaxPooling2D	---	0
8	dropout_6	Dropout	---	0
9	conv2d_11	Conv2D	[(3, 3, 64, 128), (128)]	73856
10	batch_normalization_6	BatchNormalization	[(128), (128), (128), (128)]	512
11	max_pooling2d_10	MaxPooling2D	---	0
12	dropout_7	Dropout	---	0
13	flatten_3	Flatten	---	0
14	dense_6	Dense	[(3200, 512), (512)]	1638912
15	batch_normalization_7	BatchNormalization	[(512), (512), (512), (512)]	2048
16	dropout_8	Dropout	---	0
17	dense_7	Dense	[(512, 10), (10)]	5130

ÇESA-4 modelinde 3 adet *Conv2D* katmanı bulunmaktadır. Birinci *Conv2D* katmanı (3, 3, 3, 32) şeklinde bir filtre yapısına sahip olup, 3x3 boyutundaki çekirdeği 3 kanallı

(RGB) bir görüntüye uygular ve 32 özellik haritası üretir. İkinci *Conv2D* katmanı 64 filtre içerirken, üçüncü *Conv2D* katmanı ise 128 filtreye sahiptir, bu da modelin giderek daha derin özellikleri öğrenmesini sağlar. *BatchNormalization* katmanları her evrişim katmanının çıkışını normalleştirerek öğrenmeyi hızlandırır. *MaxPooling2D* katmanları özellik haritalarının boyutunu azaltarak modelin daha büyük giriş verileriyle başa çıkmasını sağlar. Bu modelde her evrişim bloğundan sonra *Dropout* katmanı eklenmiştir. *Flatten* katmanı iki boyutlu özellik haritalarını tek boyutlu bir vektöre çevirerek tam bağlı katmanlara giriş olarak hazır hale getirir. İlk *Dense* katmanı, 3200 boyutlu giriş verisini 512 nörona bağlayarak daha karmaşık özellikleri öğrenmeye devam eder. Bu katman oldukça yüksek sayıda parametreye sahiptir. İkinci *Dense* katmanı ise çıkış olarak 10 nöron içerir ve sınıflandırma görevi için kullanılır (örneğin, CIFAR-10 veri seti 10 sınıfa sahip bir veri seti için uygundur). Son *Dense* katmandan önce bir *BatchNormalization* ve *Dropout* katmanı daha eklenmiştir. Bu katmanlar, modelin daha dengeli öğrenmesini ve aşırı öğrenme riskinin azaltılmasını sağlar.

ÇESA modellerine genel baktığımızda katmanlar ÇAOA tarafından oluşturulmuştur. Parametre sayısı, katman ve hiper-parametrelerini göre şekillenir. Parametre sayıları, katman sayısından çok, hangi katmanların kaç kere kullanıldığına, veriye, filtre boyut ve sayılarına vb. bağlıdır. Bu durum, modellerdeki *Dense* katmanı parametre sayılarında belirgin şekilde görülmektedir.

2.8. Kullanılan Veri Setleri

2.8.1. CIFAR-10

CIFAR-10, makine öğrenimi ve bilgisayarla görme alanlarında yaygın olarak kullanılan bir görüntü sınıflandırma veri kümesidir. Kanada İleri Araştırmalar Enstitüsü (Canadian Institute for Advanced Research, CIFAR) veri tabanı, Alex Krizhevsky ve Geoffrey Hinton 2009 yılında geliştirilmiştir (Krizhevsky ve Hinton, 2009). Veri seti 60,000 adet 32x32 piksel boyutundaki renkli görüntüden oluşur. Bu veri kümesinin temel amacı, farklı görüntü sınıflandırma algoritmalarının performansını değerlendirmek için standart bir benchmark sunmaktır.

CIFAR-10 veri kümesi 60,000 görüntüden oluşur ve bu görüntülerin 50,000'i eğitim için, 10,000'i ise test için ayrılmıştır. Veri kümesi 10 sınıfa ayrılmıştır ve her sınıf 6,000

görüntü içermektedir. Bu set, uçak, otomobil, kuş, kedi, geyik, köpek, kurbağa, at, gemi ve kamyon sınıflarından oluşmaktadır. Her görüntü 32x32 piksel boyutunda olup, RGB renk modelinde üç kanallıdır.

CIFAR-10 veri kümesi derin öğrenme modelleri özellikle de ESA üzerinde yapılan araştırmalarda yaygın olarak kullanılmaktadır. Veri kümesinin görece küçük boyutu ve iyi tanımlanmış sınıfları, çeşitli makine öğrenimi algoritmalarının performansını test etmek ve karşılaştırmak için ideal bir ortam sunmaktadır. Her sınıfta eşit sayıda görüntü bulunması, dengeli sınıflandırma sonuçları elde etmeyi kolaylaştırır. Görüntü sınıfları günlük nesneleri kapsadığından, geniş bir uygulama alanına sahiptir.

2.8.2. CIFAR-100

CIFAR-100, makine öğrenimi ve görüntü işleme araştırmalarında sıkça kullanılan bir diğer popüler veri kümesidir. CIFAR-100, Alex Krizhevsky ve Geoffrey Hinton tarafından oluşturulan CIFAR-10'un bir genişletmesi olup, daha fazla sınıf ve daha karmaşık bir yapı sunmaktadır. CIFAR tarafından yayımlanan bu veri kümesi, 100 farklı sınıfa ayrılmış 60,000 adet 32x32 boyutlarında renkli görüntü içermektedir (Krizhevsky ve Hinton, 2009).

CIFAR-100 veri kümesi içerdiği 60,000 görüntünün 50,000'i eğitim için, 10,000'i test için ayrılmıştır. Her biri 600 görüntüye sahip 100 sınıf bulunmaktadır. Bu sınıflar 20 üst sınıfa gruplandırılmıştır. Her üst sınıf 5 alt sınıf içerir. Örneğin, "araçlar" üst sınıfı altında bisiklet, otobüs, kamyon, otomobil ve motosiklet alt sınıfları bulunmaktadır. Her görüntü 32x32 piksel boyutunda olup 3 kanallı RGB renk modelindedir.

CIFAR-100, CIFAR-10'a kıyasla daha fazla sınıf içerir dolayısıyla sınıflandırma görevi daha karmaşıktır. 100 sınıfın bulunması sınıflandırma görevini zorlaştırır ve daha sofistike modellerin geliştirilmesini teşvik eder. Üst sınıflar ve alt sınıflar arasında yapılan analizler, sınıflandırma modellerinin hem genel hem de ayrıntılı düzeyde performansını değerlendirmeye yardımcı olur. Derin öğrenme algoritmalarının karmaşık sınıflandırma problemlerindeki performanslarını kıyaslamak için önemli bir benchmark veri kümesidir.

2.8.3. MNIST

Düzenlenmiş Ulusal Standartlar ve Teknoloji Enstitüsü (Modified National Institute of Standards and Technology, MNIST) veri kümesi, el yazısı rakamların sınıflandırılması için geliştirilmiş, makine öğrenimi ve derin öğrenme algoritmalarını test etmek amacıyla

yaygın olarak kullanılan bir veri setidir. MNIST 1998 yılında Yann LeCun, Corinna Cortes ve Christopher Burges tarafından geliştirilmiştir. Günümüzde görüntü işleme ve sınıflandırma problemlerinde temel benchmarklardan biri olarak kabul edilmektedir.

MNIST veri kümesi toplamda 70,000 el yazısı rakam görüntüsünden oluşur. Bu görüntüler 28x28 piksel boyutunda, gri tonlamalı ve 0 ile 9 arasındaki rakamları temsil eden etiketlerle birlikte gelir. 60,000 adet görüntü eğitim için ayrılmıştır. 10,000 adet görüntü test seti olarak kullanılır. Görüntülerin küçük boyutlu ve gri tonlu olması, hızlı eğitim ve test süreçlerine imkan tanır. Böylece daha büyük ve karmaşık veri kümelerine geçiş öncesi algoritmaların prototiplenmesi ve test edilmesi kolaylaşır. Veri kümesi her sınıf için dengeli sayıda görüntü içerir, bu da sınıflandırma algoritmalarının performansını sağlıklı bir şekilde değerlendirmeye olanak tanır. Görüntü sınıflandırma görevlerinden daha ileri derin öğrenme uygulamalarına kadar geniş bir kullanım alanına sahiptir. Temel seviyede bir veri kümesi olduğundan, modern derin öğrenme algoritmalarını tam anlamıyla zorlamayabilir. Bu nedenle daha karmaşık veri kümeleri üzerinde daha ileri deneyler yapmadan önce bir başlangıç noktası olarak kullanılmaktadır.

2.8.4. FRUITS-360

FRUITS-360 veri kümesi, farklı meyve ve sebze türlerinin sınıflandırılması amacıyla kullanılan geniş bir görüntü veri setidir. Bu veri seti özellikle bilgisayarla görme ve derin öğrenme tabanlı sınıflandırma algoritmalarını test etmek için Kaggle platformunda sunulmaktadır. FRUITS-360, meyve ve sebze görüntülerini içeren iyi yapılandırılmış ve etiketlenmiş bir veri kümesi olup, sınıflandırma algoritmaları üzerinde deneyler yapmak için ideal bir kaynak sağlamaktadır (Oltean, 2024).

FRUITS-360 veri kümesi 131 farklı meyve ve sebze türünden oluşan yaklaşık 90,000 görüntü içerir. Görüntüler sabit bir arka plana ve 100x100 piksel boyutuna sahiptir. Eğitim için 67,692, test seti için ise 22,847 görüntü ayrılmıştır. Her bir görüntü, farklı meyve ve sebze türlerini temsil eden etiketlerle ilişkilendirilmiştir. Veri setinde yaygın meyve ve sebzelerden egzotik türlere kadar geniş bir yelpazede sınıflar bulunmaktadır (örneğin, elma, muz, portakal, papaya, avokado, vb.).

Veri seti 131 farklı sınıfı içerdiğinden, sınıflandırma algoritmalarının geniş bir yelpazedeki meyve ve sebze türlerini tanımasını sağlar. Bu da modellerin genelleme yeteneğini test etmek için faydalıdır. Her sınıfta dengeli sayıda görüntü bulunması,

sınıflandırma modellerinin eğitiminde performansın tutarlı olmasını sağlar. Veri kümesinin gerçek dünya nesnelerini (meyve ve sebzeler) içermesi, uygulamalı bilgisayarla görme projelerinde kullanılabilirliği artırır. Özellikle, marketlerde veya gıda işleme endüstrisinde görüntü tanıma uygulamaları için kullanılabilir. Bazı meyve ve sebzeler görsel olarak birbirine çok benzer olabilir. Bu durum sınıflandırma modellerinin bu ince farkları öğrenmesini gerektirir ve veri setinin zorluk derecesini artırır. Tüm görüntüler sabit bir arka plan ve ışık koşulları altında çekildiğinden, gerçek dünya senaryolarında daha farklı koşullarda performansı test etmek gerekebilir. Veri kümesinin büyük boyutu ve geniş sınıf çeşitliliği, modellerin farklı sınıflarda nasıl performans gösterdiğini gözlemlemek açısından zengin bir ortam sunmaktadır.



3. BULGULAR VE TARTIŞMA

Deneylerde, literatürdeki çeşitli ESA modellerinin eğitim ve test performansları tasarlanan ÇESA modelleri ile kıyaslanmıştır. Performans değerlendirmesinde, modellerin doğruluk oranları ve cevap süreleri temel ölçütler olarak kullanılmıştır. Bunun yanı sıra, modellerin karmaşıklık düzeylerini anlamak amacıyla parametre ve katman sayıları da detaylı bir şekilde analiz edilmiştir.

Deneyisel çalışmalarda, DenseNet121, DenseNet169, DenseNet201, EfficientNetB0, EfficientNetB2, EfficientNetB3, EfficientNetB5, ResNet50, ResNet101, ResNet152, ResNet101V2, ResNet152V2, NASNetMobile, NASNetLarge, MobileNet, MobileNetV2 modelleri kullanılmıştır. Bu modeller derin öğrenme alanında en yaygın kullanılan güncel ve güçlü ESA mimarileri olup, görüntü sınıflandırma problemlerinde yüksek performans sergilemektedirler.

3.1. Deneyisel Ayarlar

Deneylerde ÇESA model üretiminde de kullanılan CIFAR-10, CIFAR-100 ve MNIST veri setlerine ek olarak FRUITS-360 veri seti de kullanılarak istatistiksel analiz ve yakınsama analizi deneyleri gerçekleştirilmiştir. Her veri seti önce kendine özgü belirlenen oranlarında eğitim ve test olmak üzere ikiye gruba ayrılmıştır. Eğitim seti ESA ağlarını eğitmek için kullanılırken doğrulama seti de hiper-parametreleri ayarlamak ve aşırı uyumdan kaçınmak için kullanılmıştır. Doğrulama veri seti miktarı eğitim verisinin 20%'si olarak belirlenmiştir. Test seti ise modellerin nihai performansını değerlendirmek için ayrılmıştır. Deneyler, Intel i9-13900K işlemcili, NVIDIA GeForce RTX 4060 grafik kartlı, 32GB RAM bellek ve Windows 11 işletim sistemine sahip yüksek performanslı bilgisayarda gerçekleştirilmiştir.

Modeller TensorFlow 2.0, PyTorch ve Keras kütüphaneleri kullanılarak uygulanmış, veri işleme ve görselleştirme için ek olarak NumPy, Pandas ve Matplotlib gibi Python kütüphaneleri kullanılmıştır. Eğitim ve değerlendirme süreçlerinde tekrarlanabilirlik ve verimlilik sağlamak amacıyla özel Python betikleri geliştirilmiştir. Eğitim sürecinde çevrim sayısı 50, yığın boyutu 32 ve doğrulama oranı 0.2 olarak deneyler gerçekleştirilmiştir.

Önerilen ÇESA yöntemi, CIFAR-10, CIFAR-100 ve MNIST veri setleri üzerinde NSGA-II algoritmasıyla modeller oluşturmıştır. Ek olarak CIFAR-10 veri seti üzerinde, geliştirilen FDB-NSGA-II algoritması ile de ayrıca bir model oluşturulmuştur. ÇESA yöntemi hiç bir uzman bilgisi gerektirmeden modelin oluşturulması ve optimizasyon süreçlerini tamamlamıştır.

Deneyler modelin doğruluk oranı ve cevap süresi bakımından kararlı sonuçlar verdiğini değerlendirmek amacıyla yalnızca test veri seti üzerinde 21 tekrarlı olarak gerçekleştirilmiştir. Tekrarlı testler, modelin farklı koşullarda tutarlı performans sergileyip sergilemediğini gözlemlemek için önemlidir. 21 tekrarlı test süreci, özellikle doğruluk oranı ve cevap süresi gibi kritik performans metriklerinin rastlantısal faktörlerden etkilenmesini en aza indirir ve bu metriklerde istatistiksel güvenilirlik sağlar. Bu şekilde modelin test verisi üzerindeki kararlılığı ve genelleme kapasitesi daha sağlam bir temelde değerlendirilmiştir. Hesaplanan doğruluk oranları ve cevap süreleri 21 tekrarlı deneyin ortalaması şeklinde verilmiştir. Model spesifikasyonu ise eğitim süreciyle tamamlanmış test aşamasında değişmemiştir.

3.2. MNIST Veri Setiyle Yapılan Deneyler

MNIST veri seti küçük boyutlu olduğundan orjinal boyutlarıyla yaygın kullanılan DÖ algoritmalarında çalışmamaktadır. Önerilen ÇESA modelleri MNIST verisetinde başarılı sonuçlar üretebilmiştir. Bu amaçla, ÇESA modellerinin çevrim sayısı (epok) bu veriseti üzerinden belirlenmiştir. Önerilen ÇESA modelleri, 50 ve 100 çevrim sayısında denenmiş ve 50 çevrim sayısının oldukça yeterli olduğu görülmüştür. Diğer verisetlerinde de, literatürdeki ve önerilen modellerde 50 çevrim sayısının yeterli olduğu yakınsama grafiklerinde görülmektedir.

3.2.1. MNIST İstatistiksel Analiz

Tablo 8 MNIST veri seti üzerinde 50 çevrim sayısı çalıştırılan ÇESA modellerinin doğruluk oranlarını, model spesifikasyonlarını ve cevap sürelerini karşılaştırmaktadır. Çalışmada dört farklı ÇESA modeli kullanılmış ve her modelin eğitim, doğrulama, test doğruluk oranları ile parametre sayısı, katman sayısı, model boyutu ve cevap süreleri analiz edilmiştir.

Tablo 8. MNIST veri setinde 50 çevrim sayısı ile yapılan istatistiksel analiz

Model	Doğruluk Oranı			Model Spesifikasyonu			Cevap Süresi (ms)	
	Eğitim	Doğrulama	Test	Parametre Sayısı	Katman Sayısı	Boyut (MB)	Eğitim	Test
ÇESA-1	0,9984	0,9940	0,9939	1,87E+05	17	0,7127	2,67E+05	613,4484
ÇESA-2	0,9988	0,9910	0,9921	3,90E+05	10	1,4893	1,71E+05	493,7785
ÇESA-3	0,9983	0,9928	0,9932	2,25E+05	8	0,8584	1,30E+05	438,7431
ÇESA-4	0,9961	0,9943	0,9940	6,91E+05	17	2,6363	2,53E+05	758,7855

Modellerin eğitim, doğrulama ve test setleri üzerindeki doğruluk oranları oldukça yüksektir. ÇESA-4 test doğruluk oranında %99.40 ile en yüksek performansı göstermiştir. ÇESA-1 eğitimde %99.84, doğrulamada %99.40, testte %99.39 ile oldukça iyi bir performans sergilemiştir. ÇESA-4 eğitimde %99.61, doğrulamada %99.43 ve testte %99.40 ile en düşük eğitim doğruluğuna sahip model olup test performansında en iyi sonucu vermektedir. Tüm modellerin doğruluk oranları %99'un üzerinde olup MNIST veri seti için oldukça başarılı sonuçlar elde edildiğini göstermektedir.

ÇESA-4, 6.91E+05 parametre sayısı ve 2.6363 MB model boyutuna sahip olup, en büyük modeldir ve 17 katman içermektedir. ÇESA-2, 3.90E+05 parametreye ve 1.4893 MB boyuta sahiptir. Bu model 10 katman ile en düşük katman sayısına sahiptir. ÇESA-3, 2.25E+05 parametre sayısı ile orta düzeyde bir model olup 0.8584 MB boyutunda ve 8 katmanlıdır. ÇESA-1, 1.87E+05 parametre sayısı ve 0.7127 MB boyutu ile 17 katman içerir. Diğer modellerden daha küçük bir boyuta sahiptir. ÇESA-3 az katman ve parametreye sahip olmasına rağmen, diğer modellere kıyasla oldukça rekabetçi sonuçlar elde etmiştir.

ÇESA-3 hem eğitim süresinde 1.30E+05 ms ile hemde test süresinde 438.7431 ms ile en hızlı sonuç veren modeldir. ÇESA-4 en büyük model olduğu için eğitimde 2.53E+05 ms ve test süresinde 758.7855 ms ile en uzun süreleri almaktadır. ÇESA-1 modeli eğitimde 2.67E+05 ms sürede eğitimi tamamlamış, testte ise 613.4484 ms sürede yanıt vermiştir. ÇESA-2 eğitim süresinde 1.71E+05 ms ile diğer modellere göre hızlı, test süresinde ise 493.7785 ms ile dengeli bir sonuç vermektedir.

Doğruluk oranları tüm modellerde oldukça yüksek olduğundan, modellerin hepsi MNIST veri seti üzerinde etkili sonuçlar vermektedir. Daha az parametre ve katman ile iyi sonuçlar veren modeller özellikle hızlı çözümler gerektiren uygulamalarda tercih edilebilir.

Daha küçük modellerin eğitim ve test aşamalarında daha hızlı olduğu görülmektedir. Özellikle ÇESA-3 daha hızlı yanıt süresi gerektiren sistemlerde kullanılabilir. ÇESA-3 az katman ve parametre sayısına rağmen oldukça iyi performans sunmuş ve daha hızlı yanıt sürelerine sahip olmuştur.

3.2.2. Çevrim Sayısının Performans Üzerine Etkisi

Tablo 9 MNIST veri seti üzerinde 100 çevrim sayısı çalıştırılan ÇESA modellerinin doğruluk oranları, model spesifikasyonları ve cevap sürelerini göstermektedir. Çalışmada dört farklı ÇESA modeli kullanılmış ve her modelin eğitim, doğrulama, test doğruluk oranları ile parametre sayısı, katman sayısı, model boyutu ve cevap süreleri analiz edilmiştir.

Tablo 9. MNIST veri setinde 100 çevrim sayısı ile yapılan istatistiksel analiz

Model	Doğruluk Oranı			Model Spesifikasyonu			Cevap Süresi (ms)	
	Eğitim	Doğrulama	Test	Parametre Sayısı	Katman Sayısı	Boyut (MB)	Eğitim	Test
ÇESA-1	0,9991	0,9935	0,9944	1,87E+05	17	0,7127	5,31E+05	597,1210
ÇESA-2	0,9990	0,9924	0,9923	3,90E+05	10	1,4893	3,05E+05	516,4685
ÇESA-3	0,9986	0,9922	0,9934	2,25E+05	8	0,8584	2,60E+05	433,2864
ÇESA-4	0,9980	0,9948	0,9951	6,91E+05	17	2,6363	5,14E+05	777,0326

Tüm ÇESA modelleri %99'un üzerinde doğruluk oranlarına ulaşmıştır. Modeller MNIST veri seti için oldukça yüksek bir performans göstermektedir. ÇESA-4 %99.51 test doğruluğu ile en yüksek performansı göstermiştir. ÇESA-1 %99.91 eğitim doğruluğu ve %99.44 test doğruluğu ile diğer modellerden biraz daha düşük performans sergilemiştir. 50 çevrim sayısı sonuçları ile karşılaştırıldığında, doğruluk oranlarında belirgin bir fark olmadığı gözlenmektedir. Yani 50 ve 100 çevrim sayısı çalıştırma arasında doğruluk oranı açısından anlamlı bir fark olmamıştır.

ÇESA-3 eğitimde $2.60E+05$ ms ve testte 433.2864 ms ile en hızlı yanıt süresine sahip modeldir. ÇESA-1 ve ÇESA-4 daha büyük parametre ve model boyutlarına sahip oldukları için eğitim ve test süreçlerinde daha uzun süreler almıştır. Test süresi açısından ÇESA-4 en büyük model olmasından dolayı en uzun süreyi gerektirmiştir (777.0326 ms).

50 ve 100 çevrim sayısı arasındaki deney sonuçları incelendiğinde doğruluk oranlarında anlamlı bir fark olmadığı gözlenmiştir. Eğitim ve test süreleri doğal olarak 100 çevrim sayısında biraz daha uzamış olsa da, doğruluk oranı bakımından, 50 çevrim sayısında elde edilen sonuçlar ile benzerdir. Bu durum daha hızlı sonuç almak için 50 çevrim sayısının yeterli olabileceğini, 100 çevrim sayısının ise doğruluğu çok az iyileştirdiğini göstermektedir.

3.3. CIFAR-10 Veri Setiyle Yapılan Deneyler

Eğitim doğruluğu, modelin eğitildiği veri setindeki başarı oranını gösterir. Genel olarak, yüksek eğitim doğruluğu modelin veri setine iyi uyum sağladığını gösterse de, çok yüksek değerler aşırı öğrenmeye işaret edebilir. Modellerin parametre sayıları, boyutları ile paralellik göstermektedir. Ancak katman sayısı doğrudan parametre sayısı ve boyut hakkında bir fikir vermez. Kullanılan katmanlara ve yapılarına göre parametre sayısı ile boyut değişkenlik gösterir.

3.3.1. CIFAR-10 İstatistiksel Analiz

Tablo 10'da CIFAR-10 veri seti üzerinde yapılan deneysel çalışmadan elde edilen modellerin eğitim, doğrulama ve test performanslarını gösterilmektedir. Elde edilen doğruluk oranları, model spesifikasyonları ve cevap süresi detaylı bir şekilde analiz edilmektedir.

DenseNet121 ve DenseNet169 modelleri yaklaşık 0.96 eğitim doğruluğu ile en yüksek değere sahiptir. ÇESA-1 (0.78), ÇESA-2 (0.82), ÇESA-3 (0.69) ve ÇESA-4 (0.80) doğruluğa sahip olup, eğitim doğruluk oranında daha düşük seviyelerde kalmaktadır. Ancak modellerin nihai amacı test veriseti üzerinde yüksek başarı elde etmektir. Eğitim setindeki yüksek doğruluk oranı test verisetinde düşük kalıyorsa model eğitim setini ezberliyor demektir. Doğrulama doğruluğu, modelin yeni verilere nasıl genellenebileceğini gösteren önemli bir ölçümdür. Aşırı öğrenme olmadan yüksek doğrulama doğruluğu, modelin genelleme yeteneğinin iyi olduğunu gösterir.

Tablo 10. CIFAR-10 veri seti test sonuçları

Model	Doğruluk Oranı			Model Spesifikasyonu			Cevap Süresi (ms)	
	Eğitim	Doğrulama	Test	Parametre Sayısı	Katman Sayısı	Boyut (MB)	Eğitim	Test
ÇESA-1	0,7804	0,7404	0,7651	1,87E+05	17	0,7149	5,61E+05	998,6804
ÇESA-2	0,8214	0,7624	0,7863	9,15E+05	10	3,4915	5,28E+05	932,6549
ÇESA-3	0,6940	0,7028	0,7185	5,45E+05	8	2,0794	5,19E+05	949,0840
ÇESA-4	0,8025	0,8050	0,8145	1,74E+06	17	6,6385	5,06E+05	1060,2422
DenseNet121	0,9659	0,7736	0,8069	7,05E+06	429	26,8850	2,14E+06	4091,8598
DenseNet169	0,9615	0,7883	0,8064	1,27E+07	597	48,2923	2,94E+06	5151,6500
DenseNet201	0,9597	0,7847	0,7952	1,83E+07	709	69,9661	3,57E+06	6147,6851
EfficientNetB0	0,8709	0,7879	0,7895	4,06E+06	240	15,4968	1,36E+06	2190,2933
EfficientNetB2	0,8729	0,7941	0,7940	7,78E+06	342	29,6885	1,90E+06	2808,7497
EfficientNetB3	0,8223	0,7997	0,8026	1,08E+07	387	41,1946	2,13E+06	3442,8854
EfficientNetB5	0,8318	0,7431	0,7245	2,85E+07	579	108,8486	3,33E+06	5265,0771
ResNet50	0,9520	0,7944	0,7970	2,36E+07	177	90,0581	1,64E+06	3124,5930
ResNet101	0,9100	0,6746	0,6958	4,27E+07	347	162,8062	2,87E+06	4885,3741
ResNet152	0,7387	0,5608	0,5822	5,84E+07	517	222,7456	4,17E+06	6505,7518
ResNet101V2	0,9548	0,7933	0,8066	4,26E+07	379	162,6856	2,88E+06	5327,6455
ResNet152V2	0,9518	0,7611	0,7797	5,84E+07	566	222,5957	4,22E+06	7990,2363
NASNetMobile	0,8971	0,7389	0,7477	4,28E+06	771	16,3280	3,36E+06	6036,9632
NASNetLarge	0,9324	0,7297	0,7497	8,50E+07	1041	324,0858	5,74E+06	7618,3507
MobileNet	0,9034	0,7940	0,8035	3,24E+06	91	12,3562	6,88E+05	1353,0164
MobileNetV2	0,8884	0,7625	0,7826	2,27E+06	156	8,6624	9,00E+05	1883,0471

DenseNet169 (0.788) ve EfficientNetB2 (0.794) yüksek doğrulama doğruluğuna sahiptir. ÇESA-4 (0.805) doğrulama doğruluğuna ulaşarak, büyük modellerle benzer doğruluk sergilerken daha kompakt bir yapıya sahip olduğunu göstermektedir. Diğer ÇESA modelleri daha düşük doğrulama doğruluğuna sahiptir, ancak bu durum, daha hafif olmalarına karşı elde edilen bir denge olarak kabul edilebilir. Test doğruluğu, modelin tamamen yeni bir veri kümesi üzerindeki performansını gösterir. Yüksek test doğruluğu, modelin genelleme kapasitesinin bir göstergesidir. EfficientNetB0 (0.789) ve ResNet101V2 (0.806) gibi modeller yüksek test doğruluğu sağlar. ÇESA-4 (0.814) test doğruluğuna sahip olup, bütün modellerden daha iyi olduğunu göstermektedir. Bu sonuç ÇESA-4 modelinin hem verimli hem de etkili olduğunu ispatıdır. ÇESA-4 modelinde, diğer ÇESA modellerinden farklı olarak UÇD seçimi kullanılmasının etkili olduğu söylenebilir.

Eğitim doğruluğu, modelin eğitildiği veri setindeki başarı oranını gösterir. Genel olarak, yüksek eğitim doğruluğu modelin veri setine iyi uyum sağladığını gösterse de, çok yüksek değerler aşırı öğrenmeye işaret edebilir. DenseNet121 ve DenseNet169 modelleri yaklaşık 0.96 eğitim doğruluğu ile en yüksek değere sahiptir. ÇESA-1 (0.78), ÇESA-2 (0.82), ÇESA-3 (0.69) ve ÇESA-4 (0.80) eğitim doğruluğuna sahip olup, daha düşük seviyelerde kalmaktadır. Ancak modellerin nihai amacı test veriseti üzerinde yüksek başarı elde etmektir. Eğitim setindeki yüksek doğruluk oranı test verisetinde düşük kalıyorsa model eğitim setini ezberliyor demektir. Doğrulama doğruluğu, modelin yeni verilere nasıl genellenebileceğini gösteren önemli bir ölçümdür. Aşırı öğrenme olmadan yüksek doğrulama doğruluğu, modelin genelleme yeteneğinin iyi olduğunu gösterir. DenseNet169 (0.788) ve EfficientNetB2 (0.794) yüksek doğrulama doğruluğuna sahiptir. ÇESA-4 (0.805) doğrulama doğruluğuna ulaşarak, büyük modellerle benzer doğruluk sergilerken daha kompakt bir yapıya sahip olduğunu göstermektedir. Diğer ÇESA modelleri daha düşük doğrulama doğruluğuna sahiptir, ancak bu durum, daha hafif olmalarına karşı elde edilen bir denge olarak kabul edilebilir. Test doğruluğu, modelin tamamen yeni bir veri kümesi üzerindeki performansını gösterir. Yüksek test doğruluğu, modelin genelleme kapasitesinin bir göstergesidir. EfficientNetB0 (0.789) ve ResNet101V2 (0.806) gibi modeller yüksek test doğruluğu sağlar. ÇESA-4 (0.814) test doğruluğuna sahip olup, bütün modellerden daha iyi olduğunu göstermektedir. Bu sonuç ÇESA-4 modelinin hem verimli hem de etkili olduğunu ispatıdır. ÇESA-4 modelinde, diğer ÇESA modellerinden farklı olarak UÇD seçimi kullanılmasının etkili olduğu söylenebilir.

Parametre sayısı modelin karmaşıklığını ve hesaplama gereksinimlerini belirler. Daha az parametre, hafif ve hızlı modeller için avantaj sağlar. MobileNetV2 (2,270,794 parametre) ve MobileNet (3,239,114 parametre) gibi modeller daha az parametreye sahiptir. ÇESA-1 (187,402) ve ÇESA-2 (915,274) parametre sayısında oldukça verimlidir. Özellikle ÇESA-1, çok düşük parametre sayısı ile dikkat çekmektedir. Katman sayısı modelin derinliğini ve öğrenme kapasitesini gösterir. Daha fazla katman, daha karmaşık desenlerin öğrenilmesine olanak tanır ancak aynı zamanda daha fazla hesaplama gerektirir. NASNetLarge (1041 katman) ve DenseNet201 (709 katman) en derin modellerdir. ÇESA-3 (8 katman) ve ÇESA-2 (10 katman) oldukça düşük katman sayısına sahiptir ve bu da hızlı ve hafif bir yapı sunduğunu gösterir. Model boyutu modelin hafızada kapladığı yeri gösterir. Hafif ve düşük bellek gereksinimine sahip modeller daha geniş bir uygulama alanına sahiptir. MobileNetV2 (8.66 MB) ve MobileNet (12.35 MB) en küçük modellere örnektir. ÇESA-1 (0.71 MB) ve ÇESA-2 (3.49 MB) en küçük boyutlu modeller arasında yer alır. Bu durum, ÇESA modellerinin düşük bellek kullanımı gerektiren ortamlarda avantaj sağladığını göstermektedir.

Eğitim süresi, modelin eğitilmesi için gereken süreyi ifade eder. Daha kısa eğitim süresi, modelin daha hızlı eğitilmesini sağlar. ÇESA-4 (505,872 ms) ve ÇESA-3 (519,390 ms) eğitim süresi açısından en hızlı olan modellerdir. ÇESA modelleri, diğer büyük modellere göre çok daha kısa sürede eğitilmektedir. Özellikle ÇESA-4 ve ÇESA-3, eğitim süresi açısından çok daha verimlidir ve hızlı eğitim gerektiren durumlarda avantaj sağlar. Test süresi, modelin bir tahmin yapma süresini ifade eder. Düşük test süresi gerçek zamanlı uygulamalarda avantaj sağlar. ÇESA-1 (998 ms) ve ÇESA-2 (932 ms) test süresi açısından en hızlı modellerdir. Tüm ÇESA modelleri, düşük test süreleri ile öne çıkmaktadır. Bu durum, ÇESA modellerinin gerçek zamanlı uygulamalar için ideal olduğunu göstermektedir.

ÇESA modelleri özellikle düşük parametre sayısı, düşük model boyutu, hızlı eğitim süresi ve hızlı test süresi ile öne çıkmaktadır. ÇESA-4 modeli, yüksek doğrulama ve test doğruluğu ile diğer büyük modellerle rekabet ederken, boyut ve hız açısından büyük bir avantaj sağlar. Diğer ÇESA modelleri ise daha hafif, hızlı ve kompakt bir yapı sunarak düşük kapasiteli cihazlar için ideal seçeneklerdir. ÇESA-4, yüksek doğrulama ve test doğruluğu ile büyük modellerden daha iyi performans sergilemiştir. ÇESA-1 ve ÇESA-2 gibi modeller, düşük parametre ve katman sayısı ile hafif ve hızlı uygulamalarda mükemmel bir denge sağlayacaklardır.

3.3.2. CIFAR-10 Yakınsama Analizi

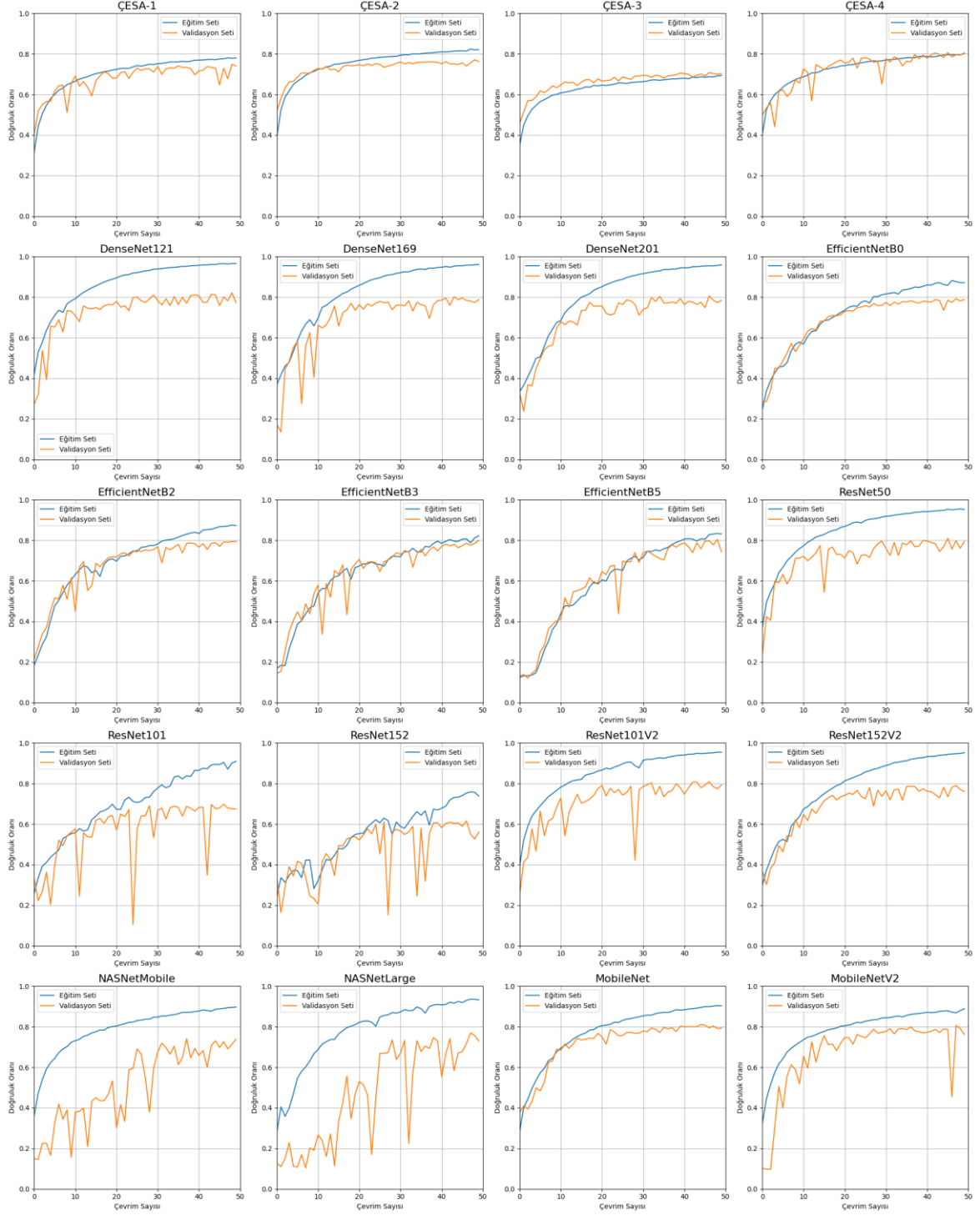
Şekil 13'te her bir model için eğitim ve doğrulama setlerine ait doğruluk oranlarının yakınsama eğrileri gösterilmektedir. Grafikler farklı modellerin eğitim süreçlerinde nasıl performans sergilediklerini ve doğrulama setindeki başarımlarını görselleştirmektedir. Bu tür doğruluk eğrileri genellikle modelin ne kadar iyi öğrendiğinin ve genelleme yapabildiğinin değerlendirilmesini sağlamaktadır.

DenseNet modelleri, eğitim doğruluğunda yüksek performans sergilemelerine rağmen doğrulama doğruluğundaki düşüş ve dalgalanmalar nedeniyle aşırı uyum belirtileri göstermektedir. DenseNet121 modelinde eğitim doğruluğu oldukça yüksekken doğrulama doğruluğundaki belirgin fark, modelin eğitim setine fazla uyum sağladığını işaret etmektedir. DenseNet169 modelinde de benzer şekilde yüksek eğitim doğruluğuna karşı doğrulama doğruluğu düşük ve dalgalıdır. Bu durum aşırı uyum sorununu düşündürmektedir. DenseNet201 modeli, eğitim doğruluğu yüksek olmasına rağmen doğrulama doğruluğunda belirgin bir düşüş ve dalgalanma yaşamakta, doğrulama setinde stabil bir performans sergilememektedir.

EfficientNet modelleri, genel olarak eğitim doğruluğunda yüksek bir performans sunmaktadır. Ancak doğrulama doğruluğunda dengesizlikler gözlenmektedir. EfficientNetB0 ve EfficientNetB3 modelleri, doğrulama doğruluğunda dalgalanma eğilimindeyken EfficientNetB2 modeli daha sınırlı bir stabilite sağlamaktadır. EfficientNetB5 modeli, eğitim doğruluğu çok yüksek olmasına rağmen doğrulama doğruluğunda dalgalanmalarla karşılaşmaktadır. Bu durum, EfficientNet ailesinde doğrulama performansını artırmak için daha fazla optimizasyon gerektiğini ortaya koymaktadır.

ResNet modellerinde eğitim doğruluğu genelde yüksek seviyedeysen doğrulama doğruluğundaki düşüş ve dalgalanmalar dikkat çekmektedir. ResNet50 modeli, doğrulama doğruluğunun belirgin şekilde düşük olduğu bir performans sergilerken, ResNet101 ve ResNet101V2 modellerinde doğrulama doğruluğundaki dalgalanma öne çıkmaktadır. ResNet152 ve ResNet152V2 modellerinde ise eğitim doğruluğu çok yüksek seviyelere ulaşmasına rağmen doğrulama doğruluğu düşük kalmaktadır. Bu, ResNet modellerinde eğitim ve doğrulama performansı arasındaki tutarlılığın sağlanmasının önemli bir zorluk olduğunu göstermektedir.

NASNet modelleri doğrulama doğruluğu açısından dalgalanmalar yaşamaktadır. NASNetMobile modeli yüksek eğitim doğruluğu sunarken doğrulama doğruluğunda dalgalanmalar görülmektedir. NASNetLarge modelinde doğrulama doğruluğu düşük ve oldukça dalgalıdır.



Şekil 13. CIFAR-10 veri seti üzerinde yapılan yakınsama analizi

MobileNet mimarilerine bakıldığında, MobileNet modeli eğitim ve doğrulama doğrulukları arasında dengeli bir performans sergilerken, MobileNetV2 modeli doğrulama doğruluğunda düşük ve dalgalı bir performans göstermektedir. Bu durum, NASNet ve MobileNet modellerinde doğrulama setine yönelik optimizasyon ihtiyacını vurgulamaktadır.

ÇESA modelleri, eğitim ve doğrulama doğruluğu arasındaki uyum açısından diğer modellere kıyasla daha dengeli bir performans sergilemektedir. ÇESA-1 modelinde eğitim doğruluğu yüksek olmasına rağmen doğrulama doğruluğunun nispeten düşük kalması, modelin aşırı uyuma yatkınlığını göstermektedir. ÇESA-2 modeli, eğitim ve doğrulama doğrulukları arasındaki uyumla iyi bir genelleme kapasitesine işaret etmektedir. ÇESA-3 modeli dengeli bir performans sunarken, ÇESA-4 modeli eğitim ve doğrulama doğruluğu arasındaki farkı minimize ederek en stabil performansı göstermektedir.

Grafiklerden görüldüğü üzere, ÇESA modelleri özellikle doğrulama seti üzerinde daha dengeli bir performans sergilemektedir. Özellikle ÇESA-4, eğitim ve doğrulama doğruluğu arasındaki farkı minimize ederek en stabil performansı göstermektedir. Diğer taraftan, DenseNet, EfficientNet ve ResNet gibi daha büyük modellerde aşırı uyum belirtileri gözlemlenmektedir. Bu modellerde eğitim doğruluğu çok yüksek seviyelere ulaşırken doğrulama doğruluğu dalgalanmakta veya daha düşük seviyelerde kalmaktadır.

3.4. CIFAR-100 Veri Setiyle Yapılan Deneyler

3.4.1. CIFAR-100 İstatistiksel Analiz

Tablo 11’de CIFAR-100 veri seti üzerinde yapılan deneysel çalışmadan elde edilen modellerin eğitim, doğrulama ve test performanslarını gösterilmektedir. Elde edilen doğruluk oranları, model spesifikasyonları ve cevap süresi detaylı bir şekilde analiz edilmektedir.

Test doğruluğu, modelin yeni veriler üzerinde nasıl performans gösterdiğini ifade eder. Yüksek test doğruluğu, modelin gerçek dünya senaryolarında başarılı olacağına işaret eder. DenseNet121 (0.4987) ve ÇESA-4 (0.5466) modelleri, en yüksek test doğruluğuna sahip olan modellerdir. ÇESA-4 en yüksek test doğruluğuna ulaşarak, DenseNet121 gibi daha büyük modellerle rekabet etmektedir. ÇESA-4 modelinin hafif yapısına rağmen yüksek test doğruluğu sağlaması, modelin genelleme kapasitesinin güçlü olduğunu gösterir.

Tablo 11. CIFAR-100 veri seti test sonuçları

Model	Doğruluk Oranı			Model Spesifikasyonu			Cevap Süresi (ms)	
	Eğitim	Doğrulama	Test	Parametre Sayısı	Katman Sayısı	Boyut (MB)	Eğitim	Test
ÇESA-1	0,4468	0,4177	0,4539	2,11E+05	17	0,8031	5,12E+05	1091,3274
ÇESA-2	0,4265	0,3950	0,4200	9,38E+05	10	3,5797	4,98E+05	852,7615
ÇESA-3	0,2797	0,3197	0,3537	5,57E+05	8	2,1237	4,96E+05	1034,0948
ÇESA-4	0,5458	0,4935	0,5466	1,79E+06	17	6,8146	5,01E+05	962,8179
DenseNet121	0,9202	0,4664	0,4987	7,14E+06	429	27,2370	2,17E+06	3882,3948
DenseNet169	0,9065	0,4535	0,4699	1,28E+07	597	48,8639	2,91E+06	4901,3271
DenseNet201	0,9154	0,4381	0,4511	1,85E+07	709	70,6256	3,38E+06	5767,0991
EfficientNetB0	0,6296	0,4185	0,4368	4,18E+06	240	15,9366	1,31E+06	2007,0488
EfficientNetB2	0,5165	0,3921	0,4414	7,91E+06	342	30,1722	1,85E+06	2871,2475
EfficientNetB3	0,4433	0,3901	0,4145	1,09E+07	387	41,7222	2,10E+06	3404,9375
EfficientNetB5	0,3959	0,3773	0,3930	2,87E+07	579	109,5521	3,44E+06	4587,8673
ResNet50	0,9385	0,4193	0,4531	2,38E+07	177	90,7616	1,68E+06	2877,8951
ResNet101	0,8813	0,3616	0,3763	4,29E+07	347	163,5097	2,85E+06	4761,1787
ResNet152	0,6480	0,2597	0,2880	5,86E+07	517	223,4491	4,10E+06	6142,7236
ResNet101V2	0,9395	0,4201	0,4386	4,28E+07	379	163,3891	2,60E+06	4460,6495
ResNet152V2	0,9437	0,4403	0,4588	5,85E+07	566	223,2992	3,79E+06	6196,2404
NASNetMobile	0,6437	0,3939	0,3939	4,38E+06	771	16,6909	3,19E+06	4979,1009
NASNetLarge	0,5409	0,2504	0,2946	8,53E+07	1041	325,4704	5,51E+06	7430,1062
MobileNet	0,5443	0,3981	0,3956	3,33E+06	91	12,7081	6,64E+05	1391,9659
MobileNetV2	0,4639	0,4158	0,4173	2,39E+06	156	9,1022	8,97E+05	1853,6744

Eğitim doğruluğu, modelin eğitildiği veri setine ne kadar uyum sağladığını gösterir. Çok yüksek eğitim doğruluğu ile düşük test doğruluğu arasındaki fark, aşırı uyum belirtisi olabilir. DenseNet121 modelinin eğitim doğruluğu (%92.0) ile test doğruluğu (%49.8) arasında büyük bir fark var, bu da modelin eğitildiği veri setine fazla uyum sağladığını ve yeni verilerde benzer başarıyı sağlayamayabileceğini gösterir. ÇESA modelleri arasında ÇESA-4, eğitim doğruluğu (%54.6) ile test doğruluğu (%54.7) arasındaki küçük fark sayesinde aşırı uyum yapmadığını gösterir. Bu da, ÇESA-4 modelinin dengeli bir performans sunduğunu gösterir.

Parametre sayısı, modelin hesaplama gereksinimlerini belirler. Daha düşük parametre sayısı daha hafif modeller için avantaj sağlar. ÇESA-1 (210,532 parametre) ve ÇESA-2 (938,404 parametre) parametre sayısında oldukça verimlidir. Özellikle ÇESA-1 en düşük parametre sayısına sahip model olarak dikkat çeker. DenseNet121 7,140,004 parametre ile çok yüksek bir parametre sayısına sahip olup, bu durum daha fazla hesaplama gerektirir ve modelin hafif cihazlarda kullanımı zor olabilir. Katman sayısı, modelin derinliğini ve öğrenme kapasitesini ifade eder. Daha fazla katman, daha karmaşık desenlerin öğrenilmesine olanak tanır ancak hesaplama yükünü de artırır. DenseNet121, 429 katmanla en derin modeldir ve bu da yüksek bir hesaplama gereksinimi anlamına gelir. ÇESA modelleri, özellikle ÇESA-2 (10 katman) ve ÇESA-3 (8 katman) gibi düşük katman sayısına sahip modeller daha hafif bir yapı sunar. Model boyutu, modelin bellekte kapladığı yeri ifade eder. Daha küçük modeller hafif cihazlar için idealdir. ÇESA-1 (0.80 MB) ve ÇESA-2 (3.58 MB) model boyutlarında en verimli modellerdir. DenseNet121 27.23 MB boyutuyla çok daha büyük bir model olup, bu da düşük bellek kapasitesine sahip cihazlarda kullanımını zorlaştırır.

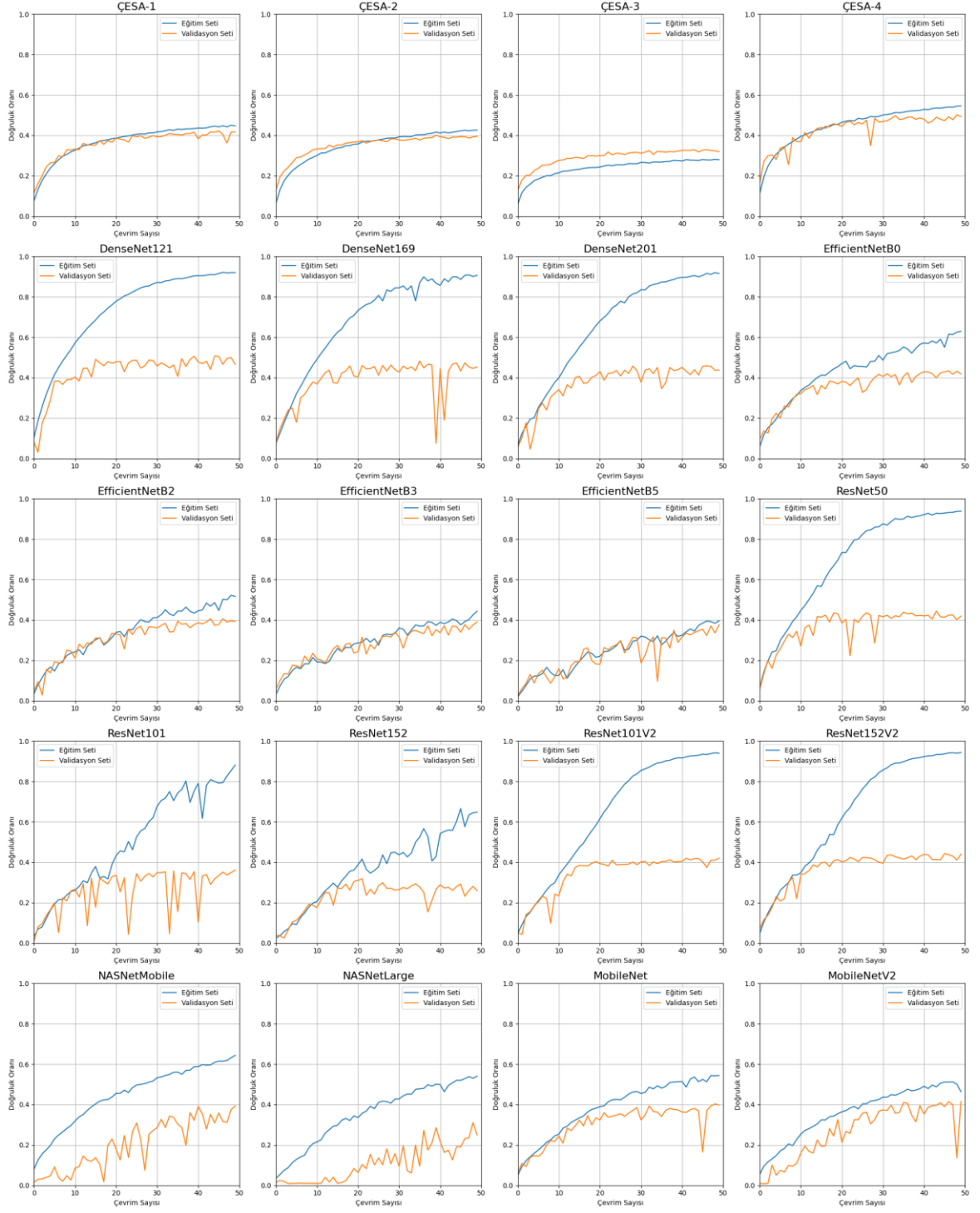
Eğitim süresi, modelin eğitilmesi için gereken süreyi ifade eder. Daha kısa eğitim süresi hızlı bir geliştirme süreci sağlar. ÇESA-2 (497,735 ms) ve ÇESA-3 (495,904 ms) modelleri en hızlı eğitim sürelerine sahiptir. DenseNet121 eğitim süresi 2,170,903 ms ile çok daha uzun sürmektedir, bu da eğitimde yüksek hesaplama gücü gerektirdiğini gösterir. Test süresi, modelin bir tahmin yapması için gereken süreyi ifade eder. Gerçek zamanlı uygulamalarda düşük test süresi avantaj sağlar. ÇESA-2 (852 ms) ve ÇESA-4 (962 ms) test süreleriyle gerçek zamanlı uygulamalara uygun yapıda hızlı modellerdir. DenseNet121 modelinin test süresi 3,882 ms ile daha yavaş bir modeldir, bu nedenle gerçek zamanlı uygulamalarda tercih edilmeyebilir.

ÇESA modelleri özellikle hafif yapısı ve hızlı test/eğitim süreleri ile öne çıkmaktadır. DenseNet121 gibi büyük modellerle karşılaştırıldığında, ÇESA modelleri düşük parametre ve katman sayısı, küçük model boyutu ve hızlı test süresi avantajları ile gerçek dünya uygulamaları için daha uygundur. ÇESA-4 yüksek test doğruluğu ile genelleme kapasitesini kanıtlar. Bu model, hafif yapısı ve hızlı çalışması ile yüksek doğruluk gerektiren uygulamalarda dahi iyi performans sunabilir. ÇESA-1 ve ÇESA-2, düşük parametre sayısı ve hızlı test süresi sayesinde verimli modeller olarak öne çıkar. Bu modeller, bellek kullanımı ve yanıt süresinin kritik olduğu durumlarda tercih edilebilir. ÇESA modelleri, hem hız hem de hafiflik odaklı tasarlanmış olmalarına rağmen, dengeli test doğruluğu ile aşırı uyumdan kaçınmakta ve yüksek genelleme kapasitesi sunmaktadır. Bu durum, ÇESA modellerinin çok amaçlı optimizasyon hedeflerine ulaşan başarılı bir model tasarımını temsil ettiğini gösterir.

3.4.2. CIFAR-100 Yakınsama Analizi

Şekil 14’de yer alan CIFAR-100 veri seti için yapılan model yakınsama analizine göre her bir modelin eğitim ve doğrulama doğrulukları grafiklerle gösterilmiştir. Bu grafikler analiz edildiğinde modellerin eğitim sürecindeki öğrenme kapasitesi ve doğrulama setindeki genelleme yetenekleri karşılaştırılmaktadır.

DenseNet modelleri, eğitim doğruluğunda hızlı ve yüksek başarı elde etmelerine rağmen doğrulama doğruluğundaki dengesizlikler ve dalgalanmalar nedeniyle genelleme performansı açısından sınırlamalar göstermektedir. DenseNet121, eğitim doğruluğunu hızla %100 seviyelerine çıkarırken doğrulama doğruluğu daha düşük kalmakta ve dalgalanmalar göstermektedir. Bu durum, modelin doğrulama setine yeterince uyum sağlayamadığını ve aşırı uyum belirtileri taşıdığını işaret etmektedir. DenseNet169 modelinde de eğitim doğruluğu oldukça yüksektir, ancak doğrulama doğruluğu ciddi dalgalanmalar sergileyerek modelin doğrulama verisinde istikrarsız bir performans gösterdiğini ortaya koymaktadır. DenseNet201, eğitim doğruluğunda yüksek değerlere ulaşırken doğrulama doğruluğundaki düşük performans ve dalgalanmalar, modelin genelleme kapasitesinin sınırlı olduğunu ve aşırı uyuma eğilim gösterdiğini açıkça göstermektedir. Özetle DenseNet modelleri hızlı yakınsama yaparak eğitim verisine aşırı uyum sağlamaktadır.



Şekil 14. CIFAR-100 veri seti üzerinde yapılan yakınsama analizi

EfficientNet modelleri genel olarak yüksek eğitim doğruluğu sağlamakla birlikte doğrulama doğruluğunda istikrar sorunları yaşamaktadır. EfficientNetB0 modelinde eğitim doğruluğu hızla yükselirken doğrulama doğruluğundaki ciddi dalgalanmalar, modelin doğrulama verisine uyum sağlamada zorlandığını ortaya koymaktadır. EfficientNetB2,

eğitim doğruluğu yüksek bir seviyede dengelerken doğrulama doğruluğu nispeten daha stabil görünmektedir; ancak dalgalanmalar tam anlamıyla giderilememiştir. EfficientNetB3, doğrulama doğruluğunda belirgin dalgalanmalar göstermekte ve modelin doğrulama setine tam uyum sağlama konusunda sınırlı bir performans sunduğunu işaret etmektedir. EfficientNetB5, eğitim doğruluğunu çok yüksek seviyelere taşıırken doğrulama doğruluğundaki dalgalanmalar artmakta ve aşırı uyuma eğilim göstermektedir.

ResNet modelleri eğitim doğruluğunda yüksek performans elde etmekle birlikte doğrulama doğruluğunda ciddi düşüşler ve dalgalanmalar yaşamaktadır. ResNet50, hızlı bir şekilde yüksek eğitim doğruluğuna ulaşmasına rağmen doğrulama doğruluğunun düşük seviyelerde kalması, modelin doğrulama verisinde başarısız olduğunu göstermektedir. ResNet101, eğitim doğruluğunda yüksek başarı sağlarken doğrulama doğruluğundaki dalgalanmalar, modelin doğrulama setine tutarlı bir şekilde uyum sağlayamadığını ortaya koymaktadır. ResNet152, eğitim doğruluğunda çok yüksek değerlere ulaşmasına rağmen doğrulama doğruluğundaki düşüşler, modelin aşırı uyum belirtileri taşıdığını güçlendirmektedir. Benzer şekilde, ResNet101V2 ve ResNet152V2 modellerinde de doğrulama doğruluğundaki düşüklük ve dalgalanmalar dikkat çekmektedir; bu durum doğrulama verisinde yeterli stabilitenin sağlanamadığını işaret etmektedir.

NASNet modellerinde doğrulama doğruluğu açısından tutarsızlıklar gözlemlenmektedir. NASNetMobile, eğitim doğruluğunda yüksek başarı sergilemesine rağmen doğrulama doğruluğunda dalgalanmalar yaşamakta ve genelleme kapasitesinde zayıflık göstermektedir. NASNetLarge, eğitim doğruluğu yüksek olsa da doğrulama doğruluğu düşük ve dalgalıdır, bu da modelin doğrulama verisinde stabil bir performans sergileyemediğini göstermektedir. MobileNet ailesinde ise daha dengeli bir performans sunulmaktadır. MobileNet, eğitim ve doğrulama doğrulukları arasında uyumlu bir ilişki sergilerken MobileNetV2, doğrulama doğruluğunda dalgalanmalarla birlikte düşüş göstermektedir. Bu durum, MobileNet modellerinin genelleme kapasitesinde daha başarılı olduğunu işaret etmektedir.

ÇESA modelleri, eğitim ve doğrulama doğrulukları arasında dengeli bir performans sergileyerek genelleme kapasitesini daha iyi korumaktadır. ÇESA-1, yüksek eğitim doğruluğu elde ederken doğrulama doğruluğunda nispeten düşük bir başarı göstermekte ve bazı dalgalanmalar sergileyerek aşırı uyum belirtileri taşımaktadır. ÇESA-2, eğitim ve doğrulama doğrulukları arasındaki uyumu koruyarak genelleme kabiliyetinin güçlü olduğunu göstermektedir. ÇESA-3, eğitim ve doğrulama doğrulukları açısından dengeli bir

performans sergileyerek modelin genelleme kapasitesinin yeterli olduğunu ortaya koymaktadır. ÇESA-4, eğitim ve doğrulama doğruluğu arasında en iyi dengeyi sağlayarak her iki veri setinde de güçlü bir performans sunmaktadır. Modelin doğrulama doğruluğundaki stabilite, ÇESA-4'ün genelleme kabiliyetinin yüksek olduğunu desteklemektedir.

Bu analizler, her modelin eğitim ve doğrulama setlerindeki performans eğilimlerini ayrıntılı bir şekilde ortaya koymaktadır. DenseNet, EfficientNet ve ResNet gibi daha büyük modeller aşırı uyum belirtileri ve doğrulama setinde dalgalanmalar göstermektedir. NASNet modelleri genelleme kapasitesi açısından zayıflıklar sergilerken, MobileNet modelleri dengeli bir performans sunmaktadır. ÇESA modelleri, özellikle ÇESA-4, doğrulama doğruluğundaki stabilitesi ve genelleme kapasitesindeki üstünlüğü ile öne çıkmaktadır.

3.5. FRUITS-360 Veri Setiyle Yapılan Deneyler

3.5.1. FRUITS-360 İstatistiksel Analiz

Tablo 12 FRUITS-360 veri seti üzerinde gerçekleştirilen deneysel sonuçları göstermektedir. FRUITS-360 geniş çeşitlilikte meyve ve sebze görüntüleri içeren bir veri seti olup, görüntü sınıflandırma problemlerinde derin öğrenme modellerinin performansını test etmek için yaygın olarak kullanılmaktadır. Tabloda çeşitli ÇESA ve ESA modellerinin eğitim, doğrulama ve test doğruluk oranları, model spesifikasyonları (parametre sayısı, katman sayısı ve boyut) ile eğitim ve test süreleri karşılaştırılmaktadır. ÇESA modelleri oluşturulma aşamasında bu veri seti kullanılmamıştır. Bu analiz ile ÇESA modellerinin dayanaklılık özelliği de test edilmiş olacaktır. Başka veri setlerinde model tasarımı ve hiper-parametre optimizasyonu gerçekleşen modeller farklı bir veri setinde analiz edilecektir.

Test doğruluğu, modelin yeni verilerde nasıl performans gösterdiğini gösteren en önemli metriklerden biridir. Yüksek test doğruluğu modelin genelleme kapasitesinin yüksek olduğunu işaret eder. MobileNet (0.9923) ve DenseNet169 (0.9903) modelleri en yüksek test doğruluğuna ulaşan modellerdir. ÇESA-2 (0.9888) ve ÇESA-4 (0.9877) yüksek test doğruluğuna sahip olup, diğer büyük modellerle rekabet edebilmektedir. Bu sonuçlar ÇESA modellerinin yüksek doğruluğa sahip uygulamalarda güvenilir bir seçenek olduğunu gösterir.

Tablo 12. FRUITS-360 veri seti test sonuçları

Model	Doğruluk Oranı			Model Spesifikasyonu			Cevap Süresi (ms)	
	Eğitim	Doğrulama	Test	Parametre Sayısı	Katman Sayısı	Boyut (MB)	Eğitim	Test
ÇESA-1	0,9953	0,9616	0,9766	6,12E+05	17	2,3335	3,85E+06	12744,2746
ÇESA-2	0,9848	0,9735	0,9888	4,85E+06	10	18,4851	3,37E+06	6831,5947
ÇESA-3	0,9638	0,9629	0,9774	4,37E+06	8	16,6702	3,33E+06	6460,5207
ÇESA-4	0,9961	0,9789	0,9877	9,60E+06	17	36,6253	3,33E+06	6812,7675
DenseNet121	0,9989	0,9785	0,9894	7,17E+06	429	27,3582	5,77E+06	13602,6495
DenseNet169	0,9995	0,9862	0,9903	1,29E+07	597	49,0608	7,15E+06	16638,3858
DenseNet201	0,9992	0,9748	0,9858	1,86E+07	709	70,8528	8,53E+06	19436,5702
EfficientNetB0	0,9983	0,9727	0,9731	4,22E+06	240	16,0880	4,17E+06	11086,7913
EfficientNetB2	0,9974	0,9781	0,9849	7,95E+06	342	30,3389	5,31E+06	12873,3697
EfficientNetB3	0,9972	0,9619	0,9742	1,10E+07	387	41,9040	6,68E+06	12953,5632
EfficientNetB5	0,9977	0,9770	0,9766	2,88E+07	579	109,7944	1,14E+07	21456,7208
ResNet50	0,9987	0,8472	0,9122	2,39E+07	177	91,0039	5,52E+06	12083,8714
ResNet101	0,9989	0,9702	0,9891	4,29E+07	347	163,7520	8,45E+06	20519,2590
ResNet152	0,9988	0,9711	0,9812	5,86E+07	517	223,6914	1,13E+07	27664,2177
ResNet101V2	0,9982	0,9851	0,9896	4,29E+07	379	163,6314	7,25E+06	17543,4306
ResNet152V2	0,9989	0,9850	0,9899	5,86E+07	566	223,5415	1,03E+07	25805,7511
NASNetMobile	0,9985	0,8733	0,8332	4,41E+06	771	16,8159	5,70E+06	15441,4127
NASNetLarge	0,9985	0,9554	0,9576	8,54E+07	1041	325,9473	1,71E+07	35583,8151
MobileNet	0,9977	0,9778	0,9923	3,36E+06	91	12,8294	4,92E+06	12943,4214
MobileNetV2	0,9975	0,8111	0,8228	2,43E+06	156	9,2537	4,71E+06	11655,0364

Eğitim doğruluğu, modelin eğitim verisine ne kadar uyum sağladığını ifade eder. Eğitim doğruluğu ile test doğruluğu arasındaki büyük bir fark aşırı uyum belirtisi olabilir. Eğitim doğruluğu (%99.95) ile test doğruluğu (%99.03) arasında küçük bir fark vardır, bu da modelin eğitim verisine çok iyi uyum sağladığını ancak yeni verilerde benzer başarıyı gösterdiğini işaret eder. ÇESA-1 (%99.53 eğitim doğruluğu) ile ÇESA-4 (%99.60) arasındaki eğitim doğruluğu yüksek olup, test doğruluğuyla uyum içindedir. ÇESA modelleri dengeli bir performans sunarak aşırı uyum yapmadıklarını gösterir.

Parametre sayısı, modelin hesaplama gereksinimlerini ve hafifliğini belirler. Daha düşük parametre sayısı özellikle mobil veya gömülü sistemler için avantaj sağlar. ÇESA-1 (611,715 parametre) en az parametreye sahip modeldir. Bu durum ÇESA-1 modelinin düşük bellek ve işlem kapasitesine sahip cihazlarda verimli bir şekilde çalışabileceğini gösterir. DenseNet201 ve ResNet152V2 modelleri parametre sayıları çok yüksektir (DenseNet201 için 18 milyon, ResNet152V2 için 58 milyonun üzerinde) ve bu nedenle daha fazla hesaplama gücü gerektirir. Katman sayısı, modelin derinliğini ve öğrenme kapasitesini gösterir. Daha fazla katman daha karmaşık desenlerin öğrenilmesine olanak tanır, ancak aynı zamanda daha yüksek hesaplama gerektirir. NASNetLarge (1,041 katman) ve DenseNet169 (597 katman) gibi modeller çok derin yapıdadır ve bu da yüksek hesaplama yükü anlamına gelir. ÇESA modelleri arasında en düşük katman sayısına sahip olan ÇESA-3 (8 katman) ve ÇESA-2 (10 katman) daha hafif ve hızlı bir yapı sunar. Model boyutu, modelin bellek gereksinimini belirler. Daha düşük boyutlu modeller hafif ve düşük bellek kapasiteli cihazlarda daha avantajlıdır. ÇESA-1 (2.33 MB) ve ÇESA-3 (16.67 MB) boyutları ile en küçük modellere örnektir. EfficientNetB5 ve NASNetLarge modelleri çok büyük boyutlara sahiptir (EfficientNetB5 için 109 MB, NASNetLarge için 326 MB), bu da düşük bellek kapasitesine sahip cihazlarda kullanımını zorlaştırır.

Eğitim süresi, modelin eğitilmesi için gereken süreyi ifade eder. Daha kısa eğitim süresi geliştirme sürecini hızlandırır. ÇESA-2 (3,370,318 ms) ve ÇESA-3 (3,331,465 ms) en hızlı eğitim sürelerine sahiptir, bu modeller daha hızlı eğitim yapmayı mümkün kılar. DenseNet201 modeli 8,529,834 ms eğitim süresi ile en uzun eğitim süresine sahip modellerden biridir. Test süresi, modelin bir tahmin yapması için geçen süreyi ifade eder. Gerçek zamanlı uygulamalarda düşük test süresi önemlidir. ÇESA-2 (6,831 ms) ve ÇESA-3 (6,460 ms) hızlı test sürelerine sahip modellerdir. DenseNet169 ve NASNetLarge modelleri çok uzun test sürelerine sahiptir (DenseNet169 için 16,638 ms ve NASNetLarge için 35,583 ms), bu da gerçek zamanlı uygulamalarda dezavantaj yaratır.

ÇESA modelleri yüksek test doğruluğu ile düşük parametre sayısı, küçük model boyutu ve hızlı test süresi özelliklerini dengeli bir şekilde sunmaktadır. Özellikle ÇESA-2 ve ÇESA-4, yüksek test doğruluğu ile büyük modellerle rekabet edebilir ve düşük kapasiteli cihazlarda daha hızlı çalışabilir. ÇESA-2 ve ÇESA-4, test doğruluğu, düşük parametre sayısı ve hızlı eğitim/test süreleri ile öne çıkarak verimli ve yüksek doğruluk gerektiren uygulamalar için uygun seçeneklerdir. ÇESA-1 ve ÇESA-3 hafif ve hızlı yapılarıyla gerçek zamanlı veya düşük kapasiteli cihazlar için idealdir. ÇESA modellerinin başarısı, hem yüksek doğruluk hem de düşük kaynak kullanımı gerektiren uygulamalar için optimize edilmiş çok amaçlı bir çözüm sunduğunu göstermektedir.

3.5.2. FRUITS-360 Yakınsama Analizi

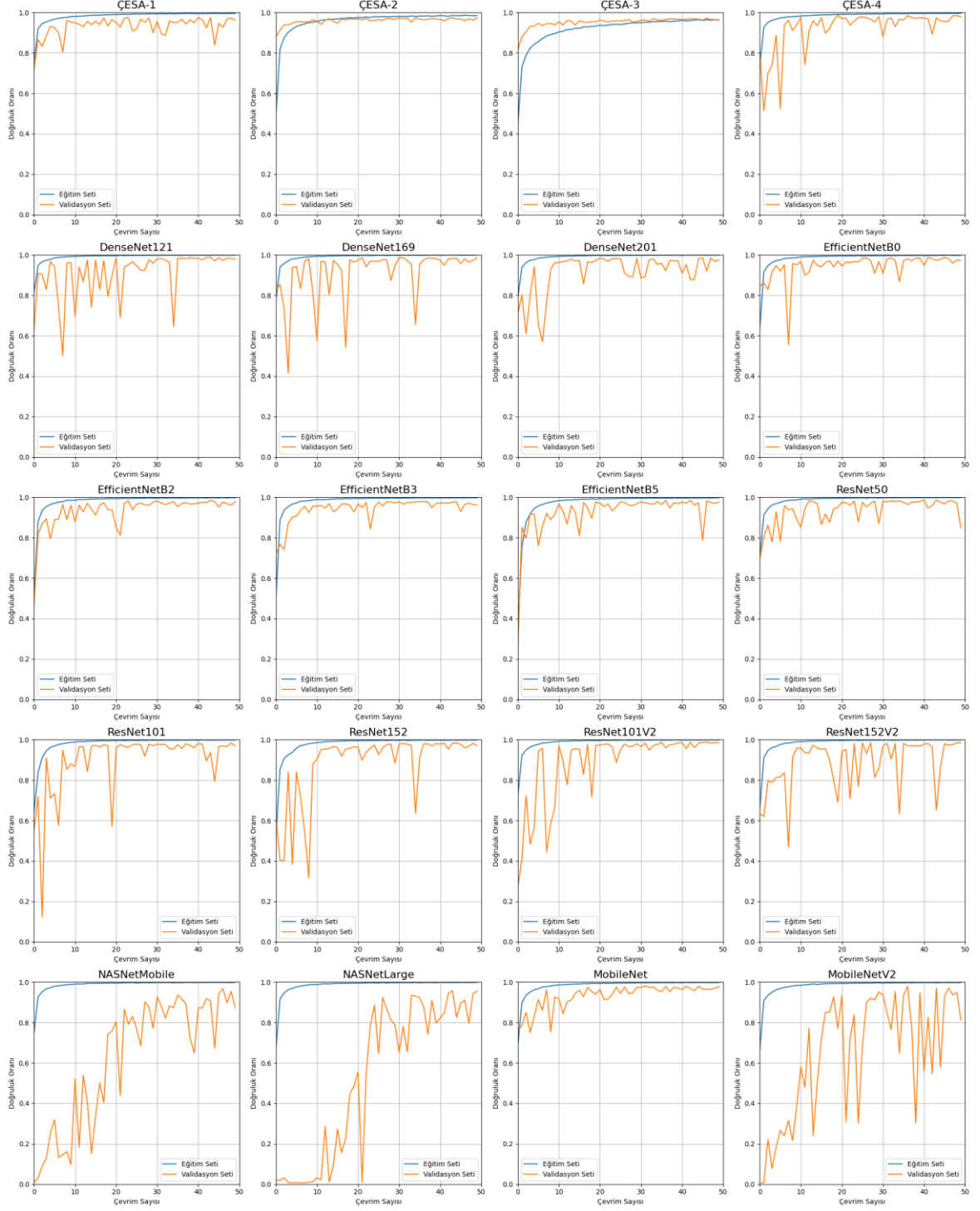
Şekil 15'te FRUITS-360 veri seti için yapılan yakınsama analizine bakıldığında, farklı modellerin eğitim ve doğrulama doğrulukları gösterilmektedir. Bu grafikler her modelin yakınsama davranışını ve genelleme yeteneklerini değerlendirmek için önemli ipuçları sunmaktadır.

ÇESA modelleri, eğitim doğruluğunda hızla yüksek seviyelere ulaşırken doğrulama doğruluğunda da genel olarak stabil ve dengeli bir performans sergilemektedirler. Özellikle ÇESA-2 ve ÇESA-4 modellerinde eğitim ve doğrulama doğrulukları arasında güçlü bir uyum bulunmaktadır, bu da bu modellerin aşırı uyumdan kaçınarak genelleme kapasitesinin iyi olduğunu göstermektedir. ÇESA-1 ve ÇESA-3 modelleri de dengeli bir doğrulama doğruluğu sunmaktadır. Genel olarak ÇESA modelleri genelleme kapasitesi açısından başarılı ve güvenilir bir performans sergiler.

DenseNet modelleri yüksek eğitim doğruluğuna çok hızlı bir şekilde ulaşmakta, ancak doğrulama doğruluğu genellikle düşük kalmakta ve dalgalı bir yapı sergilemektedir. DenseNet121, DenseNet169 ve DenseNet201 modellerinde eğitim doğruluğu yüksek olmasına rağmen doğrulama doğruluğu üzerinde istikrarsızlıklar gözlenmektedir. Bu durum DenseNet modellerinin eğitim verisine aşırı uyum sağlama eğiliminde olduğunu ve doğrulama verisinde genelleme performansının sınırlı kaldığını gösterir.

EfficientNet modelleri de eğitim doğruluğunda yüksek seviyelere ulaşmakla birlikte, doğrulama doğruluğunda bazı dalgalanmalar yaşamaktadır. EfficientNetB0 ve EfficientNetB3 gibi modeller doğrulama doğruluğunda daha dalgalı bir performans sergilerken, EfficientNetB2 ve EfficientNetB5 modellerinde nispeten daha stabil, ancak yine

de tam anlamıyla kararlı olmayan bir doğrulama doğruluğu görülmektedir. EfficientNet modelleri, eğitim doğruluğunu yükseltme konusunda başarılı olsa da doğrulama setinde kararlı bir genelleme kapasitesi sağlayamamaktadır.



Şekil 15. FRUITS-360 veri seti üzerinde yapılan yakınsama analizi

ResNet modelleri eğitim doğruluğunda yüksek değerlere ulaşırken doğrulama doğruluğu genellikle daha düşük seviyede kalmakta ve sık sık dalgalanmalar göstermektedir. ResNet50, ResNet101 ve ResNet152 gibi modeller eğitim verisine çok iyi uyum sağlasa da doğrulama setinde benzer bir başarı gösterememektedir. ResNet101V2 ve ResNet152V2 modellerinde de benzer dalgalanmalar görülmektedir, bu durum ResNet modellerinin aşırı uyum yapmaya eğilimli olduklarını ve doğrulama setinde istikrarlı bir performans gösteremediklerini işaret etmektedir.

NASNet modelleri, özellikle NASNetLarge modelinde, eğitim doğruluğunda yüksek seviyelere ulaşırken doğrulama doğruluğunda düşük ve dalgalı bir yapı sergilemektedir. NASNetMobile modelinde de eğitim doğruluğu yüksekken doğrulama doğruluğundaki istikrarsızlık gözlemlenmektedir. Bu durum, NASNet modellerinin doğrulama setinde stabil performans gösteremediğini ve aşırı uyuma eğilimli olduğunu göstermektedir.

MobileNet modelleri eğitim ve doğrulama doğrulukları arasında daha uyumlu ve dengeli bir performans sergileyen modeller olarak öne çıkmaktadır. MobileNet modelinde eğitim ve doğrulama doğrulukları arasındaki fark minimum düzeyde olup, modelin genelleme kapasitesinin yüksek olduğunu gösterir. MobileNetV2 modeli ise eğitim doğruluğunda yüksek seviyelere ulaşırken doğrulama doğruluğunda dalgalanmalar yaşamaktadır, bu da doğrulama verisinde stabilite sorununa işaret etmektedir.

3.6. Genel Değerlendirme

Tablo 13'te, 3 farklı veri setinde, önerilen ÇESA modeli sonuçları ortalamaları ile literatürde yaygın kullanılan, tez çalışmasında uygulanmış ESA modellerinin (diğer) sonuçlarının ortalamaları verilmiştir.

Tablo 13 incelendiğinde, ÇESA modeli ve diğer model arasındaki karşılaştırmalara bakıldığında, ÇESA'nın her üç veri setinde de düşük kaynak kullanımıyla öne çıktığı görülmektedir. CIFAR-10 veri setinde ÇESA, %77.11 doğruluk oranıyla diğer modelin %76.65 oranından daha iyi performans göstermiştir. Üstelik ÇESA, yalnızca 847 bin parametre ve 3.23 MB boyut kullanırken, diğer modelin 25.6 milyon parametresi ve 97.67 MB boyutu bulunmaktadır. Bunun yanı sıra, ÇESA'nın cevap süresi 985 ms iken, diğer modelin 4614 ms'lik süresi, ÇESA'nın düşük gecikmeli uygulamalar için çok daha uygun olduğunu göstermektedir.

Tablo 13. Yapılan deneylerin genel deęerlendirmesi

Veri Seti	Model	Test Doğruluk Oranı	Parametre Sayısı	Boyut (MB)	Test Cevap Süresi
CIFAR-10	ÇESA	0,7711	8,47E+05	3,2311	985,1654
	Dięer	0,7665	2,56E+07	97,6685	4613,9487
CIFAR-100	ÇESA	0,4436	8,74E+05	3,3303	985,2504
	Dięer	0,4139	2,58E+07	98,2806	4219,7161
FRUITS-360	ÇESA	0,9826	4,86E+06	18,5285	8212,2894
	Dięer	0,9589	2,58E+07	98,4915	17955,5166

CIFAR-100 veri setinde benzer bir durum gözlenmektedir. ÇESA, %44.36 doğruluk oranıyla dięer modelin %41.39 oranını geride bırakmıştır. Ayrıca, parametre sayısı ve model boyutu açısından ÇESA yine çok daha avantajlıdır. Cevap sürelerinde de ÇESA'nın 985 ms ile dięer modelin 4219 ms'lik süresine kıyasla çok daha hızlı olduęu görölmektedir. Bu, özellikle düşük donanımlı cihazlarda ÇESA'nın büyük bir fark yaratabileceğini ortaya koymaktadır.

FRUITS-360 veri setinde, her iki modelin doğruluk oranı yüksek olsa da ÇESA'nın %98.26'lık oranı, dięer modelin %95.89'luk oranından daha iyidir. ÇESA, bu veri setinde yaklaşık 4.86 milyon parametre ve 18.53 MB boyutla çalışırken, dięer modelin 25.8 milyon parametre ve 98.49 MB boyutu, ÇESA'nın ne kadar kompakt bir model olduęunu bir kez daha göstermektedir. Cevap sürelerinde ise ÇESA 8212 ms ile dięer modelin 17,956 ms'lik süresine kıyasla yaklaşık iki kat daha hızlıdır.

Genel olarak, ÇESA modeli, düşük parametre sayısı, küçük boyutu ve hızlı cevap süresiyle öne çıkmaktadır. Daha yüksek doğruluk oranlarıyla beraber, bu özellikleri ÇESA'yı özellikle mobil cihazlar, gömülü sistemler ve gerçek zamanlı uygulamalar için ideal bir seçenek haline getirmektedir. Dięer model ise güçlü donanımların bulunduęu daha büyük veri işleme projelerinde tercih edilebilir. Bu karşılaştırma, ÇESA'nın kaynak verimlilięi açısından optimize edilmiş bir model olduęunu açıkça ortaya koymaktadır.

CIFAR-10 veri seti üzerinde yapılan analizde, doğruluk açısından en iyi ÇESA modelinin ÇESA-4 olduęu ve %81.45'lik bir test doğruluęu sağladığı görölmektedir. Buna karşılık dięer modeller arasında en iyi performansı DenseNet121 %80.69 doğruluk ile sergilemiştir. ÇESA modeli doğruluk açısından %0.76'lık bir üstünlük sağladığını göstermektedir. Ortalama doğruluk deęerlendirmesinde, ÇESA modelleri %77.11'lik bir ortalama ile dięer modellerin %76.65 olan ortalamasını %0.46 oranında geçmiştir.

CIFAR-10 test süreleri açısından incelendiğinde, ÇESA modellerinden ÇESA-2'nin 932.65 ms ile en hızlı model olduğu belirlenmiştir. Diğer modeller arasında ise MobileNet 1353.02 ms ile en kısa süreye sahiptir. ÇESA modelinin test süresi açısından %31.07'lik bir avantaj sağladığını ortaya koymaktadır. ÇESA modellerinin ortalama test süresi (985.17 ms), diğer modellerin ortalamasına (4613.95 ms) kıyasla %78.65 daha hızlıdır.

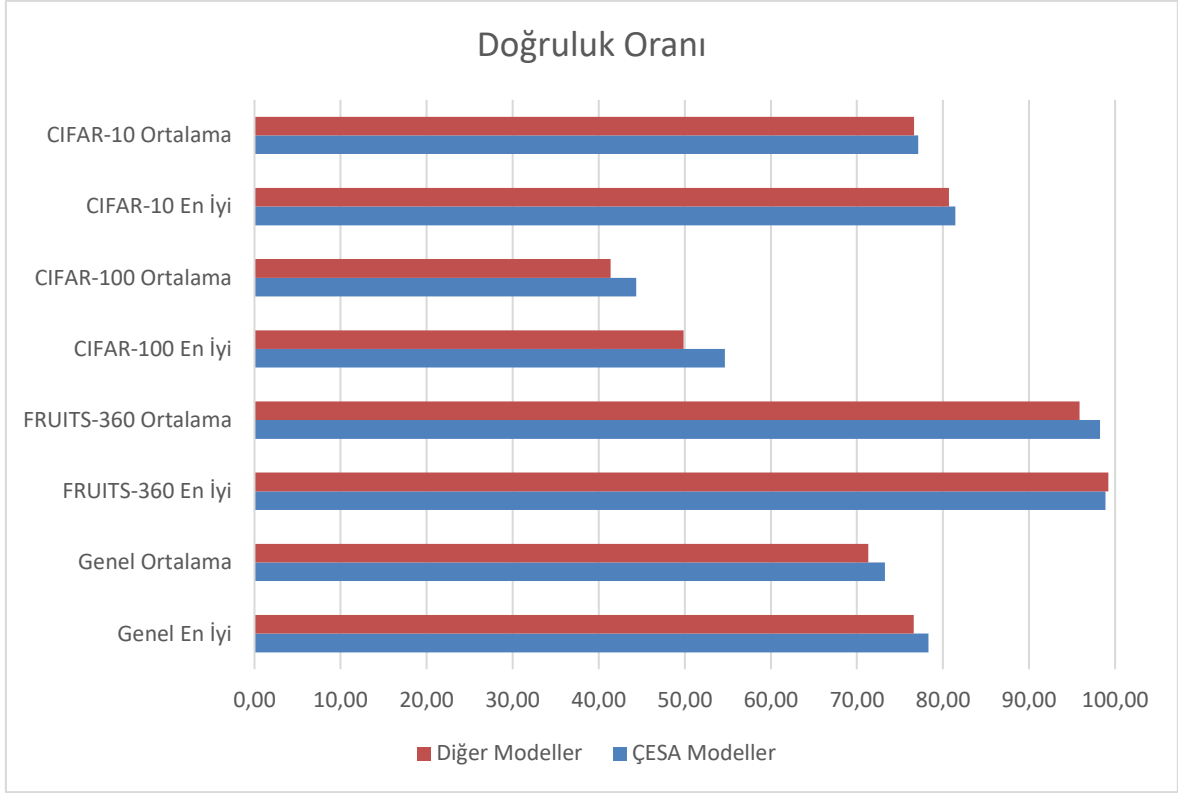
CIFAR-100 veri setinde doğruluk açısından en iyi ÇESA modelinin ÇESA-4 (%54.66), diğer modeller arasında ise DenseNet121'in (%49.87) en iyi performansı gösterdiği tespit edilmiştir. ÇESA modellerinin %4.79 gibi önemli bir doğruluk üstünlüğü sağladığını göstermektedir. Ortalama doğruluklarda ise ÇESA modelleri %44.36 ile diğer modellerin %41.39 olan ortalamasını %2.97 oranında geçmiştir.

CIFAR-100 test süreleri bakımından, en hızlı ÇESA modelinin ÇESA-2 olduğu ve 852.76 ms ile dikkat çektiği görülmektedir. Diğer modeller arasında MobileNet 1391.97 ms ile en hızlı test süresine sahiptir. ÇESA-2 modeli test süresi açısından %38.74'lük bir avantaj sağlamaktadır. Ortalama test süresi analizine göre, ÇESA modelleri (985.25 ms), diğer modellerin ortalamasına (4219.72 ms) kıyasla %76.65 daha hızlıdır.

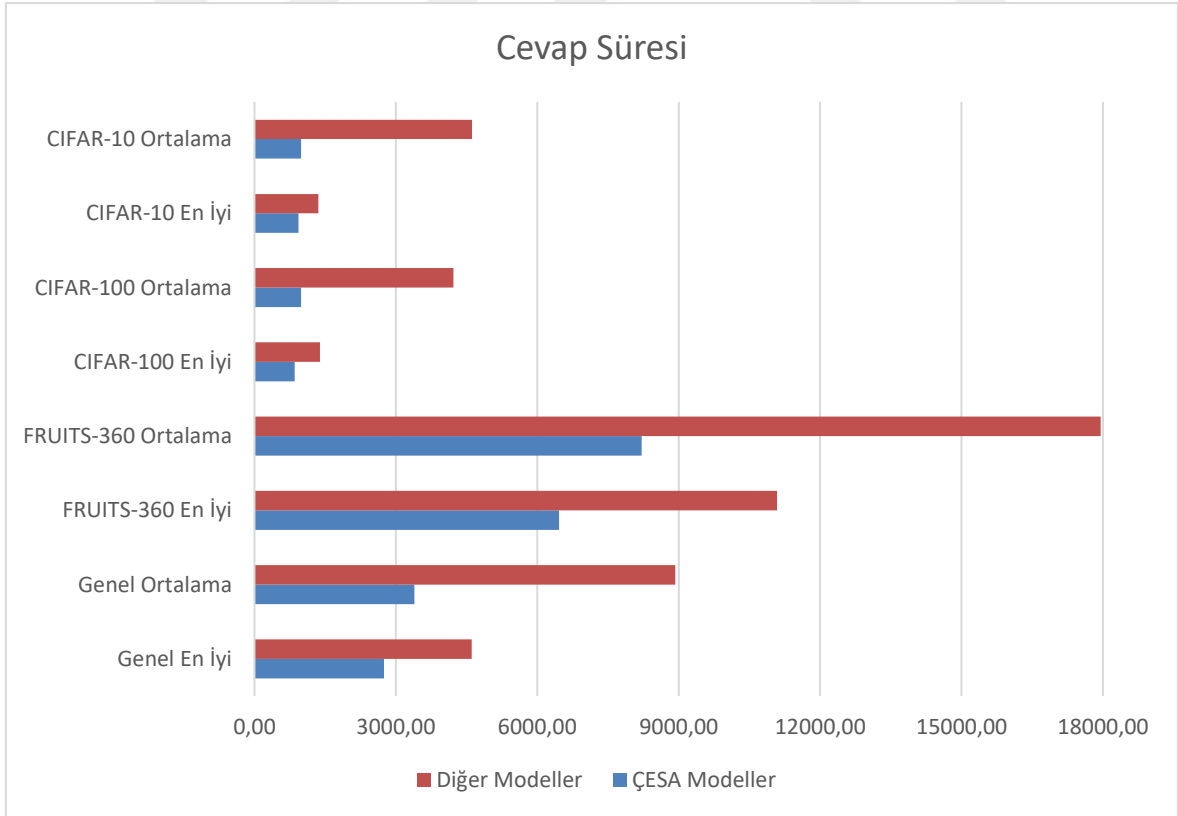
FRUITS-360 veri setinde doğruluk açısından en iyi ÇESA modelinin ÇESA-2 olduğu ve %98.88'lik doğruluk sağladığı görülmektedir. Diğer modeller arasında MobileNet %99.23 ile daha iyi performans göstermiştir. Bu da diğer modelin %0.35'lik bir üstünlüğü olduğunu ortaya koymaktadır. Ortalama doğruluk değerlendirmesinde, ÇESA modelleri %98.26 ile diğer modellerin %95.89 olan ortalamasını %2.37 oranında geçmiştir.

FRUITS-360 test süreleri açısından, en hızlı ÇESA modelinin ÇESA-3 olduğu ve 6460.52 ms ile dikkat çektiği görülmektedir. Diğer modeller arasında ise EfficientNetB0 11086.79 ms ile en hızlı modeldir. Bu durumda, ÇESA modeli test süreleri açısından %41.73'lük bir avantaj sağlamaktadır. Ayrıca, ÇESA modellerinin ortalama test süresi (8212.29 ms), diğer modellerin ortalamasına (17955.52 ms) kıyasla %54.26 daha hızlıdır.

Şekil 16'da test verileri üzerinde yapılan doğruluk analiz özeti, Şekil 17'de ise test verileri üzerinde yapılan cevap süresi analiz özeti verilmektedir. ÇESA modelleri doğruluk ve test süreleri açısından genel olarak üstün bir performans sergilemektedir. CIFAR-10 ve CIFAR-100 veri setlerinde hem doğruluk hem de test süreleri açısından belirgin avantajlar sağlarken, FRUITS-360 veri setinde doğrulukta diğer modellerle rekabetçi performans sergilenmiş, test sürelerinde ise açık bir üstünlük sağlanmıştır. Bu durum, ÇESA modellerinin hem doğruluk hem de hız optimizasyonunu bir arada sağlayabildiğini göstermektedir.



Şekil 16. Test verisi doğruluk oranı genel değerlendirme



Şekil 17. Test verisi cevap süresi genel değerlendirme

Yapılan analizlere göre, ÇESA modelleri hem doğruluk hem de test süreleri açısından dikkat çekici bir performans sunmaktadır. Ortalama doğruluklarda ÇESA modelleri, diğer modellere kıyasla %1.93'lük bir üstünlük sağlamıştır. Test cevap süreleri açısından ise ÇESA modelleri ortalamada %69.85 daha hızlıdır. Bu, ÇESA modellerinin sadece yüksek doğruluk sağlamakla kalmayıp aynı zamanda test süreçlerini önemli ölçüde hızlandırarak pratik uygulamalar için daha uygun olduğunu göstermektedir. Özellikle hız ve doğruluk arasındaki bu denge, ÇESA modellerinin farklı veri setlerinde geniş bir uygulama yelpazesi için tercih edilebilir bir seçenek olduğunu ortaya koymaktadır.



4. SONUÇLAR

Bu tezde “Çok Amaçlı Optimizasyon Teknikleriyle Dayanıklı Otomatik CNN Model Tasarımı” konusu ele alınarak, hem teorik hem de pratik katkılar sağlanmıştır. Çalışma kapsamında, ESA mimarilerinin çok amaçlı optimizasyon algoritmalarıyla otomatik olarak tasarlanması ve hiper-parametre optimizasyonunun etkin bir şekilde gerçekleştirilmesi üzerine yenilikçi bir yöntem geliştirilmiştir. Bu bağlamda, aşağıdaki önemli sonuçlara ulaşılmıştır:

- ✓ Model Performansı ve Verimlilik Dengesi: Önerilen ÇESA yöntemi, mevcut ESA mimarileriyle kıyaslandığında, daha az katman ve parametre ile hem doğruluk hem de hız açısından çok daha başarılı sonuçlar elde etmiştir. Yapılan deneylerde, önerilen modellerin test doğruluğunda %1.93 oranında artış sağlarken, test sürelerinin %69.85 oranında hızlandığı gözlemlenmiştir. Bu durum, önerilen yöntemlerin hesaplama kaynaklarını verimli kullanırken doğruluk oranından da ödün vermediğini kanıtlamaktadır.
- ✓ Çok Amaçlı Optimizasyon Algoritmalarına Uyum: Önerilen ÇESA yöntemi, farklı çok amaçlı optimizasyon algoritmaları (örneğin, NSGA-II, MOEA/D gibi) ile uyumlu çalışabilmektedir. Bu esneklik, yöntemin geniş bir problem yelpazesine uygulanabilirliğini artırarak, farklı optimizasyon senaryolarında etkin bir şekilde kullanılmasını sağlamaktadır. Bu durum, yöntemin çeşitli optimizasyon ihtiyaçlarına adapte edilebilecek esnek bir çerçeve sunduğunu göstermektedir.
- ✓ Uzman Bilgisi Gerektirmeden Otomatik Model Tasarımı: ÇESA yöntemi, uzman müdahalesine gerek kalmadan ESA mimarilerinin tasarımı ve hiper-parametre optimizasyonunu otomatik olarak gerçekleştirebilme yeteneğine sahiptir. Bu özellik, hem zaman hem de insan kaynağı tasarrufu sağlamakta, karmaşık derin öğrenme modellerinin geniş bir kullanıcı kitlesi tarafından kolayca uygulanabilmesini mümkün kılmaktadır.
- ✓ Çok Amaçlı Optimizasyonun Etkinliği: ÇESA, pareto optimal çözümler üreterek doğruluk ve hız arasındaki kritik dengeyi optimize etmeyi başarmıştır. Bu sayede, farklı uygulama ihtiyaçlarına göre özelleştirilebilen esnek ESA modelleri tasarlanmıştır. Optimizasyon süreçlerinde kullanılan baskın olmayan sıralama ve

kalabalık mesafesi yöntemleri, çözüm uzayının daha etkin bir şekilde taranmasını sağlamıştır.

- ✓ Hiper-Parametrelerin Otomatik Optimizasyonu: ESA mimarilerinin hiper-parametrelerinin otomatik olarak belirlenmesi, model tasarım sürecinde önemli bir zaman ve kaynak tasarrufu sağlamıştır. Hiper-parametrelerin çok amaçlı optimizasyon yöntemleriyle belirlenmesi, kaba kuvvetle el ile yapılan ayarlamalar sırasında karşılaşılan hata risklerini ve performans dalgalanmalarını minimize etmiştir.
- ✓ Deneysel Sonuçların Genelleştirilebilirliği: Çalışmada kullanılan dört farklı veri seti (MNIST, CIFAR-10, CIFAR-100 ve FRUITS-360) üzerinde gerçekleştirilen deneyler, önerilen yöntemin genelleştirilebilirliğini ve uygulanabilirliğini kanıtlamıştır. Farklı veri setleri üzerinde elde edilen yüksek doğruluk oranları, yöntemin geniş bir uygulama yelpazesi için uygun olduğunu göstermektedir.
- ✓ Hafif ve Dayanıklı Modellerin Geliştirilmesi: Önerilen ÇESA modelleri, özellikle sınırlı hesaplama gücüne sahip cihazlar için optimize edilmiştir. Modellerin düşük karmaşıklıkla yüksek performans sunması, mobil ve gömülü cihazlarda kullanılabilirlik açısından büyük bir avantaj sağlamaktadır.

Bu sonuçlar, ESA mimarilerinin tasarımı ve optimizasyonunda yenilikçi yaklaşımların nasıl etkili olabileceğini göstermektedir. Çalışmada geliştirilen yöntem ve elde edilen bulgular, gelecekteki araştırmalara yol gösterebilecek niteliktedir ve derin öğrenme modellerinin tasarımı konusunda önemli bir kaynak teşkil etmektedir. Tez kapsamında önerilen yöntemlerin, farklı problem türlerinde ve veri setlerinde test edilmesi, yöntemin daha geniş bir bağlamda geçerliliğini incelemek için önerilmektedir.

Bu çalışma, hem akademik hem de endüstriyel uygulamalarda kullanılabilecek etkili ve dayanıklı ESA modellerinin geliştirilmesine olanak sağlamaktadır.

5. ÖNERİLER

Bu tezde geliştirilen ÇESA yöntemi, derin öğrenme modellerinin tasarımı ve hiper-parametre optimizasyonu alanında yenilikçi bir çözüm sunmaktadır. Çalışmanın bulguları ışığında, ileride yapılabilecek araştırma ve uygulamalar şu şekilde olabilir:

1. Farklı Çok Amaçlı Optimizasyon Algoritmalarının Entegrasyonu: Çalışmada kullanılan NSGA-II ve FDB-NSGA-II gibi algoritmalar dışında, diğer çok amaçlı optimizasyon tekniklerinin (örneğin, MOEA/D, Genetik Algoritma varyantları) ÇESA yöntemiyle entegrasyonu incelenerek performans karşılaştırmaları yapılabilir. Böylece yöntemin farklı problem ve algoritma türlerinde performansı daha kapsamlı değerlendirilebilir.
2. Farklı Veri Setlerinde Genelleştirilebilirlik Testleri: MNIST, CIFAR-10, CIFAR-100 ve FRUITS-360 veri setleri üzerinde yapılan deneylerin ötesine geçilerek, medikal görüntüleme, doğal dil işleme veya otonom sürüş gibi özel alanlara yönelik veri setlerinde önerilen ÇESA modelleri test edilebilir.
3. Gerçek Zamanlı Sistemlerde Kullanım: ÇESA'nın hafif ve hızlı modeller üretme kapasitesi, gömülü sistemler, IoT cihazları ve mobil uygulamalarda daha geniş ölçekte test edilerek gerçek zamanlı sistemlerde uygulanabilirliği daha detaylı araştırılabilir.
4. Transfer Öğrenme ile Entegrasyon: ÇESA yöntemiyle üretilen modellerin, önceden eğitilmiş modellerle transfer öğrenme teknikleri kullanılarak daha büyük veri setlerinde veya farklı uygulama alanlarında performansının nasıl iyileştirilebileceği incelenebilir.
5. Adaptif Hiper-Parametre Optimizasyonu: Hiper-parametrelerin farklı veri setleri veya uygulama ihtiyaçlarına göre dinamik olarak uyarlanması için adaptif optimizasyon yaklaşımları geliştirilebilir. Örneğin, çevrim sayısı veya öğrenme oranı gibi parametreler, modelin eğitim sürecindeki performansa göre otomatik olarak optimize edilebilir.
6. Hata Toleranslı Model Geliştirme: ÇESA yöntemi, hatalara karşı daha dayanıklı modeller oluşturacak şekilde yeniden yapılandırılabilir. Bu tür bir geliştirme, özellikle kritik görevler (örneğin, tıbbi teşhis veya güvenlik uygulamaları) için önemlidir.

7. Parelel Hesaplama ve Dağıtık Sistemler Üzerinde Optimizasyon: ÇESA algoritması, çoklu GİB veya dağıtık sistemler üzerinde paralel olarak çalıştırılarak daha büyük ve karmaşık modellerin optimize edilmesi sağlanabilir. Bu, yöntemin daha büyük ölçekte uygulanabilirliğini artırabilir.
8. Enerji Verimliliği Odaklı Optimizasyon: ÇESA yöntemi, özellikle enerji tüketimi kısıtlı cihazlar için, modelin doğruluk ve hız hedeflerine ek olarak enerji verimliliğini optimize edecek şekilde genişletilebilir.

Bu öneriler, ÇESA yönteminin geniş bir uygulama yelpazesinde geliştirilmesini ve uygulanmasını sağlamaya yönelik rehber niteliği taşımaktadır. Gelecekteki çalışmalar, bu tezde sunulan yöntemleri daha da ileri taşıyabilir ve derin öğrenme modellerinin tasarımı konusunda yeni ufuklar açabilir.

6. KAYNAKLAR

- Abd Elaziz, M., Dahou, A., Abualigah, L., Yu, L., Alshinwan, M., Khasawneh, A. M., & Lu, S. (2021). Advanced metaheuristic optimization techniques in applications of deep neural networks: a review. *Neural Computing and Applications*, 1-21.
- Abdolrasol, M. G., Hussain, S. S., Ustun, T. S., Sarker, M. R., Hannan, M. A., Mohamed, R., ... & Milad, A. (2021). Artificial neural networks based optimization techniques: A review. *Electronics*, 10(21), 2689.
- Adadi, A. (2021). A survey on data-efficient algorithms in big data era. *Journal of Big Data*, 8(1), 24.
- Ahmed, S. F., Alam, M. S. B., Hassan, M., Rozbu, M. R., Ishtiaq, T., Rafa, N., ... & Gandomi, A. H. (2023). Deep learning modelling techniques: current progress, applications, advantages, and challenges. *Artificial Intelligence Review*, 56(11), 13521-13617.
- Akay, B., Karaboga, D., & Akay, R. (2022). A comprehensive survey on optimizing deep learning models by metaheuristics. *Artificial Intelligence Review*, 55(2), 829-894.
- Alam, M., Samad, M. D., Vidyaratne, L., Glandon, A., & Iftekharruddin, K. M. (2020). Survey on deep neural networks in speech and vision systems. *Neurocomputing*, 417, 302-321.
- Alnashwan, R. O., Chelloug, S. A., Almalki, N., Issaoui, I., Motwakel, A., & Sayed, A. (2023). Lighting Search Algorithm With Convolutional Neural Network-Based Image Captioning System for Natural Language Processing. *IEEE Access*.
- AlZubi, A. A., & Galyna, K. (2023). Artificial intelligence and internet of things for sustainable farming and smart agriculture. *IEEE Access*.
- Andonie, R., & Florea, A. C. (2020). Weighted random search for CNN hyperparameter optimization. *arXiv preprint arXiv:2003.13300*.
- Ankalaki, S., & Thippeswamy, M. N. (2024). Optimized Convolutional Neural Network Using Hierarchical Particle Swarm Optimization for Sensor Based Human Activity Recognition. *SN Computer Science*, 5(5), 447.
- Aras, S., Gedikli, E., & Kahraman, H. T. (2021). A novel stochastic fractal search algorithm with fitness-distance balance for global numerical optimization. *Swarm and Evolutionary Computation*, 61, 100821.
- Atteia, G., Abdel Samee, N., El-Kenawy, E. S. M., & Ibrahim, A. (2022). CNN-hyperparameter optimization for diabetic maculopathy diagnosis in optical coherence tomography and fundus retinography. *Mathematics*, 10(18), 3274.

- Aziz, M. A. E., Ewees, A. A., & Hassanien, A. E. (2018). Multi-objective whale optimization algorithm for content-based image retrieval. *Multimedia tools and applications*, 77, 26135-26172.
- Babatunde, S. O., Odejide, O. A., Edunjobi, T. E., & Ogundipe, D. O. (2024). The role of AI in marketing personalization: A theoretical exploration of consumer engagement strategies. *International Journal of Management & Entrepreneurship Research*, 6(3), 936-949.
- Bacanin, N., Bezdan, T., Tuba, E., Strumberger, I., & Tuba, M. (2020). Optimizing convolutional neural network hyperparameters by enhanced swarm intelligence metaheuristics. *Algorithms*, 13(3), 67.
- Bahmei, B., Birmingham, E., & Arzanpour, S. (2022). CNN-RNN and data augmentation using deep convolutional generative adversarial network for environmental sound classification. *IEEE Signal Processing Letters*, 29, 682-686.
- Baldominos, A., Saez, Y., & Isasi, P. (2020). On the automated, evolutionary design of neural networks: past, present, and future. *Neural computing and applications*, 32, 519-545.
- Basodi, S., Ji, C., Zhang, H., & Pan, Y. (2020). Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, 3(3), 196-207.
- Bathla, G., Bhadane, K., Singh, R. K., Kumar, R., Aluvalu, R., Krishnamurthi, R., ... & Basheer, S. (2022). Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities. *Mobile Information Systems*, 2022(1), 7632892.
- Bejani, M. M., & Ghatee, M. (2021). A systematic review on overfitting control in shallow and deep neural networks. *Artificial Intelligence Review*, 54(8), 6391-6438.
- Chen, F., Li, S., Han, J., Ren, F., & Yang, Z. (2024). Review of lightweight deep convolutional neural networks. *Archives of Computational Methods in Engineering*, 31(4), 1915-1937.
- Chen, J. S., Le, T. T. Y., & Florence, D. (2021). Usability and responsiveness of artificial intelligence chatbot on online customer experience in e-retailing. *International Journal of Retail & Distribution Management*, 49(11), 1512-1531.
- Choudhary, A., Mian, T., Fatima, S., & Panigrahi, B. K. (2024). Nature-inspired artificial bee colony-based hyperparameter optimization of CNN for anomaly detection in induction motor. *Expert Systems*, 41(7), e13407.
- Cinar, A. C. (2020). Training feed-forward multi-layer perceptron artificial neural networks with a tree-seed algorithm. *Arabian Journal for Science and Engineering*, 45(12), 10915-10938.
- Colaco, S. J., Kim, J. H., Poulouse, A., Neethirajan, S., & Han, D. S. (2023). DISubNet: depthwise separable inception subnetwork for pig treatment classification using thermal data. *Animals*, 13(7), 1184.

- Cong, S., & Zhou, Y. (2023). A review of convolutional neural network architectures and their optimizations. *Artificial Intelligence Review*, 56(3), 1905-1969.
- da Silva, C. A., Rosa, D. C., Miranda, P. B., Si, T., Cerri, R., & Basgalupp, M. P. (2024). Automated CNN optimization using multi-objective grammatical evolution. *Applied Soft Computing*, 151, 111124.
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2), 182-197.
- Doi, S., Sasaki, H., & Igarashi, H. (2019). Multi-objective topology optimization of rotating machines using deep learning. *IEEE transactions on magnetics*, 55(6), 1-5.
- Duman, S., Kahraman, H. T., Guvenc, U., & Aras, S. (2021). Development of a Lévy flight and FDB-based coyote optimization algorithm for global optimization and real-world ACOPF problems. *Soft Computing*, 25, 6577-6617.
- Duman, S., Kahraman, H. T., Sonmez, Y., Guvenc, U., Kati, M., & Aras, S. (2022). A powerful meta-heuristic search algorithm for solving global optimization and real-world solar photovoltaic parameter estimation problems. *Engineering Applications of Artificial Intelligence*, 111, 104763.
- Elfatimi, E., Eryigit, R., & Elfatimi, L. (2022). Beans leaf diseases classification using mobilenet models. *IEEE Access*, 10, 9471-9482.
- Elghazi, K., Ramchoun, H., & Masrour, T. (2024). Enhancing CNN structure and learning through NSGA-II-based multi-objective optimization. *Evolving Systems*, 1-17.
- Eremia, M., Liu, C. C., & Edris, A. A. (2016). Artificial Intelligence Techniques.
- Feng, S., Zhao, L., Shi, H., Wang, M., Shen, S., & Wang, W. (2023). One-dimensional VGGNet for high-dimensional data. *Applied Soft Computing*, 135, 110035.
- Florea, A. C., & Andonie, R. (2020). Weighted random search for hyperparameter optimization. *arXiv preprint arXiv:2004.01628*.
- Gadhvi, B., Savsani, V., & Patel, V. (2016). Multi-objective optimization of vehicle passive suspension system using NSGA-II, SPEA2 and PESA-II. *Procedia Technology*, 23, 361-368.
- Goay, C. H., Ahmad, N. S., & Goh, P. (2021). Transient simulations of high-speed channels using CNN-LSTM with an adaptive successive halving algorithm for automated hyperparameter optimizations. *IEEE Access*, 9, 127644-127663.
- Guo, Y., Li, J. Y., & Zhan, Z. H. (2020). Efficient hyperparameter optimization for convolution neural networks in deep learning: A distributed particle swarm optimization approach. *Cybernetics and Systems*, 52(1), 36-57.
- Gunantara, N. (2018). A review of multi-objective optimization: Methods and its applications. *Cogent Engineering*, 5(1), 1502242.

- Gurjar, K., Jangra, A., Baber, H., Islam, M., & Sheikh, S. A. (2024). An Analytical Review on the Impact of Artificial Intelligence on the Business Industry: Applications, Trends, and Challenges. *IEEE Engineering Management Review*.
- Guvenc, U., Duman, S., Kahraman, H. T., Aras, S., & Katı, M. (2021). Fitness–Distance Balance based adaptive guided differential evolution algorithm for security-constrained optimal power flow problem incorporating renewable energy sources. *Applied Soft Computing*, 108, 107421.
- Gülcü, A., & Kuş, Z. (2020). Hyper-parameter selection in convolutional neural networks using microcanonical optimization algorithm. *IEEE Access*, 8, 52528-52540.
- Gülmez, B. (2024). A new multi-objective hyperparameter optimization algorithm for COVID-19 detection from x-ray images. *Soft Computing*, 1-17.
- Gürgen, S., Kahraman, H. T., Aras, S., & Altın, İ. (2022). A comprehensive performance analysis of meta-heuristic optimization techniques for effective organic rankine cycle design. *Applied Thermal Engineering*, 213, 118687.
- Habib, G., & Qureshi, S. (2022). Optimization and acceleration of convolutional neural networks: A survey. *Journal of King Saud University-Computer and Information Sciences*, 34(7), 4244-4268.
- He, C., Ge, D., Yang, M., Yong, N., Wang, J., & Yu, J. (2021). A data-driven adaptive fault diagnosis methodology for nuclear power systems based on NSGAI-CNN. *Annals of Nuclear Energy*, 159, 108326.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14* (pp. 630-645). Springer International Publishing.
- Hinton, G. E. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.
- Hong, T. Y., & Chen, C. C. (2023). Hyperparameter optimization for convolutional neural network by opposite-based particle swarm optimization and an empirical study of photomask defect classification. *Applied Soft Computing*, 148, 110904.
- Hurbans, R. (2020). Grokking artificial intelligence algorithms. *Manning Publications*.
- Huixian, J. (2020). The analysis of plants image recognition based on deep learning and artificial neural network. *IEEE Access*, 8, 68828-68841.
- İnik, Ö. (2023). CNN hyper-parameter optimization for environmental sound classification. *Applied Acoustics*, 202, 109168.
- Jana, C., Banerjee, S., Maur, S., & Dalai, S. (2024). Power System Disturbance Classification using CWT Guided Customized AlexNet CNN. *IEEE Sensors Letters*.

- Jarrahi, M. H. (2018). Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making. *Business horizons*, 61(4), 577-586.
- Jia, W., Gao, J., Xia, W., Zhao, Y., Min, H., & Lu, J. T. (2021). A performance evaluation of classic convolutional neural networks for 2D and 3D palmprint and palm vein recognition. *International Journal of Automation and Computing*, 18(1), 18-44.
- Jiang, A., Du, L., & Du, Y. (2024). GroupQ: Group-Wise Quantization With Multi-Objective Optimization for CNN Accelerators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.
- Kaddani, S., Vanderpooten, D., Vanpeperstraete, J. M., & Aissi, H. (2017). Weighted sum model with partial preference information: Application to multi-objective optimization. *European Journal of Operational Research*, 260(2), 665-679.
- Kahraman, H. T., Aras, S., & Gedikli, E. (2020). Fitness-distance balance (FDB): a new selection method for meta-heuristic search algorithms. *Knowledge-Based Systems*, 190, 105169.
- Kahraman, H. T., Bakir, H., Duman, S., Katı, M., Aras, S., & Guvenc, U. (2022). Dynamic FDB selection method and its application: modeling and optimizing of directional overcurrent relays coordination. *Applied Intelligence*, 1-36.
- Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). A review of the use of convolutional neural networks in agriculture. *The Journal of Agricultural Science*, 156(3), 312-322.
- Kahraman, H. T., Katı, M., Aras, S., & Taşci, D. A. (2023). Development of the Natural Survivor Method (NSM) for designing an updating mechanism in metaheuristic search algorithms. *Engineering Applications of Artificial Intelligence*, 122, 106121.
- Khayyam, H., Javadi, B., Jalili, M., & Jazar, R. N. (2020). Artificial intelligence and internet of things for autonomous vehicles. Nonlinear approaches in engineering applications: *Automotive applications of engineering problems*, 39-68.
- Kilichev, D., & Kim, W. (2023). Hyperparameter optimization for 1D-CNN-based network intrusion detection using GA and PSO. *Mathematics*, 11(17), 3724.
- Koonce, B., & Koonce, B. (2021). EfficientNet. *Convolutional neural networks with swift for Tensorflow: image recognition and dataset categorization*, 109-123.
- Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

- Lewis, M., Bhosale, S., Dettmers, T., Goyal, N., & Zettlemoyer, L. (2021, July). Base layers: Simplifying training of large, sparse models. *In International Conference on Machine Learning* (pp. 6265-6274). PMLR.
- Li, W., Zhang, T., Wang, R., & Ishibuchi, H. (2021). Weighted indicator-based evolutionary algorithm for multimodal multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 25(6), 1064-1078.
- Li, Z., Jiang, X., Dong, Y., Meng, L., & Sun, T. (2022). An anti-steganalysis HEVC video steganography with high performance based on CNN and PU partition modes. *IEEE Transactions on Dependable and Secure Computing*, 20(1), 606-619.
- Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 33(12), 6999-7019.
- Liang, J., Yang, H., Gao, J., Yue, C., Ge, S., & Qu, B. (2019). MOPSO-based CNN for keyword selection on google ads. *IEEE Access*, 7, 125387-125400.
- Liu, J., Wu, Q., Sui, X., Chen, Q., Gu, G., Wang, L., & Li, S. (2021). Research progress in optical neural networks: theory, applications and developments. *Photonix*, 2, 1-39.
- Liu, S., Yang, J., Huang, C., & Yang, M. H. (2015). Multi-objective convolutional learning for face labeling. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3451-3459).
- Liu, X., Gong, X., Plaza, A., Cai, Z., Xiao, X., Jiang, X., & Liu, X. (2022). MO-CNN: Multiobjective Optimization of Convolutional Neural Networks for Hyperspectral Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 60, 1-14.
- Loni, M., Sinaei, S., Zoljodi, A., Daneshtalab, M., & Sjödin, M. (2020). DeepMaker: A multi-objective optimization framework for deep neural networks in embedded systems. *Microprocessors and Microsystems*, 73, 102989.
- Loussaief, S., & Abdelkrim, A. (2018). Convolutional neural network hyper-parameters optimization based on genetic algorithms. *International Journal of Advanced Computer Science and Applications*, 9(10).
- Lu, X., & Zhang, H. (2020). A Content-Aware POI Recommendation Method in Location-Based Social Networks Based on Deep CNN and Multi-Objective Immune Optimization. *Journal of Internet Technology*, 21(6), 1761-1772.
- Mahmud, M., Kaiser, M. S., Hussain, A., & Vassanelli, S. (2018). Applications of deep learning and reinforcement learning to biological data. *IEEE transactions on neural networks and learning systems*, 29(6), 2063-2079.
- Makarenko, A. V. (2020). Deep neural networks: origins development current status. *Control Sciences*, (2), 3-19.

- Mao, Q., Hu, F., & Hao, Q. (2018). Deep learning for intelligent wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 20(4), 2595-2621.
- Marler, R. T., & Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, 41, 853-862.
- Menghani, G. (2023). Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, 55(12), 1-37.
- Mohakud, R., & Dash, R. (2022). Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection. *Journal of King Saud University-Computer and Information Sciences*, 34(8), 6280-6291.
- Moradi, S., & Mokhtab Rafiei, F. (2019). A dynamic credit risk assessment model with data mining techniques: evidence from Iranian banks. *Financial Innovation*, 5(1), 1-27.
- Mostafa, S. S., Mendonca, F., Ravelo-Garcia, A. G., Juliá-Serdá, G. G., & Morgado-Dias, F. (2020). Multi-objective hyperparameter optimization of convolutional neural network for obstructive sleep apnea detection. *IEEE Access*, 8, 129586-129599.
- Murtaza, M., Ahmed, Y., Shamsi, J. A., Sherwani, F., & Usman, M. (2022). AI-based personalized e-learning systems: Issues, challenges, and solutions. *IEEE access*, 10, 81323-81342.
- Nadeem, M. I., Ahmed, K., Li, D., Zheng, Z., Naheed, H., Muaad, A. Y., ... & Abdel Hameed, H. (2022). SHO-CNN: A metaheuristic optimization of a convolutional neural network for multi-label news classification. *Electronics*, 12(1), 113.
- Naiemi, F., Ghods, V., & Khalesi, H. (2020). Scene text detection using enhanced extremal region and convolutional neural network. *Multimedia Tools and Applications*, 79(37), 27137-27159.
- Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of big data*, 2, 1-21.
- Naseri, H., & Mehrdad, V. (2023). Novel CNN with investigation on accuracy by modifying stride, padding, kernel size and filter numbers. *Multimedia Tools and Applications*, 82(15), 23673-23691.
- Naulia, P. S., Watada, J., & Aziz, I. A. (2024). A Mathematically Inspired Meta-Heuristic Approach to Parameter (Weight) Optimization of Deep Convolution Neural Network. *IEEE Access*.
- Nazar, M., Alam, M. M., Yafi, E., & Su'ud, M. M. (2021). A systematic review of human-computer interaction and explainable artificial intelligence in healthcare with artificial intelligence techniques. *IEEE Access*, 9, 153316-153348.

- Park, S., & Kwak, N. (2017). Analysis on the dropout effect in convolutional neural networks. In *Computer Vision—ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20–24, 2016, Revised Selected Papers, Part II 13* (pp. 189–204). Springer International Publishing.
- Oh, B. K., & Kim, J. (2021). Multi-objective optimization method to search for the optimal convolutional neural network architecture for long-term structural health monitoring. *IEEE Access*, 9, 44738–44750.
- Oltean M., (2024, Augustos 30). Fruits-360 dataset: *A dataset of images containing fruits, vegetables, and nuts.*: <https://www.kaggle.com/datasets/moltean/fruits>.
- Qian, L., Hu, L., Zhao, L., Wang, T., & Jiang, R. (2020). Sequence-dropout block for reducing overfitting problem in image classification. *IEEE Access*, 8, 62830–62840.
- Rahimi, I., Gandomi, A. H., Chen, F., & Mezura-Montes, E. (2023). A review on constraint handling techniques for population-based algorithms: from single-objective to multi-objective optimization. *Archives of Computational Methods in Engineering*, 30(3), 2181–2209.
- Raiaan, M. A. K., Sakib, S., Fahad, N. M., Al Mamun, A., Rahman, M. A., Shatabda, S., & Mukta, M. S. H. (2024). A systematic review of hyperparameter optimization techniques in Convolutional Neural Networks. *Decision Analytics Journal*, 100470.
- Rajagopal, A., Joshi, G. P., Ramachandran, A., Subhalakshmi, R. T., Khari, M., Jha, S., ... & You, J. (2020). A deep learning model based on multi-objective particle swarm optimization for scene classification in unmanned aerial vehicles. *IEEE Access*, 8, 135383–135393.
- Rajaraman, V. (2014). JohnMcCarthy—Father of artificial intelligence. *Resonance*, 19, 198–207.
- Raziani, S., & Azimbagirad, M. (2022). Deep CNN hyperparameter optimization algorithms for sensor-based human activity recognition. *Neuroscience Informatics*, 2(3), 100078.
- Saini, K., & Devi, R. (2023). A systematic scoping review of the analysis of COVID-19 disease using chest X-ray images with deep learning models. *J. Auton. Intell*, 7(10.32629).
- Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., & Müller, K. R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3), 247–278.
- Sarker, I. H. (2021). Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN computer science*, 2(6), 420.
- Shin, H. C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... & Summers, R. M. (2016). Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5), 1285–1298.

- Shinde, S., Kulkarni, U., Mane, D., & Sapkal, A. (2021). Deep learning-based medical image analysis using transfer learning. *Health Informatics: A Computational Perspective in Healthcare*, 19-42.
- Shlezinger, N., Eldar, Y. C., & Boyd, S. P. (2022). Model-based deep learning: On the intersection of deep learning and optimization. *IEEE Access*, 10, 115384-115398.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Song, S., Lam, J. C., Han, Y., & Li, V. O. (2020). ResNet-LSTM for Real-Time PM 2.5 and PM₁₀ Estimation Using Sequential Smartphone Images. *IEEE Access*, 8, 220069-220082.
- Sonmez, Y., Duman, S., Kahraman, H. T., Kati, M., Aras, S., & Guvenc, U. (2024). Fitness-distance balance based artificial ecosystem optimisation to solve transient stability constrained optimal power flow problem. *Journal of Experimental & Theoretical Artificial Intelligence*, 36(5), 745-784.
- Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. *In Proceedings of the AAAI conference on artificial intelligence* (Vol. 31, No. 1).
- Taha, K. (2020). Methods that optimize multi-objective problems: A survey and experimental evaluation. *IEEE Access*, 8, 80855-80878.
- Talaat, F. M., & Gamel, S. A. (2023). RL based hyper-parameters optimization algorithm (ROA) for convolutional neural network. *Journal of Ambient Intelligence and Humanized Computing*, 14(10), 13349-13359.
- Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. *In International conference on machine learning* (pp. 6105-6114). PMLR.
- Tang, T., Chen, T., Chen, X., & Gui, G. (2024). Integration of VGGNet-Driven Precipitation Similarity Analysis and Hydrological Modeling for Enhanced Streamflow Forecasting. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.
- Tedjopurnomo, D. A., Bao, Z., Zheng, B., Choudhury, F. M., & Qin, A. K. (2020). A survey on modern deep neural network for traffic prediction: Trends, methods and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 34(4), 1544-1561.
- Tian, Y., Liu, R., Zhang, X., Ma, H., Tan, K. C., & Jin, Y. (2020). A multipopulation evolutionary algorithm for solving large-scale multimodal multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 25(3), 405-418.

- Tian, Y., Si, L., Zhang, X., Cheng, R., He, C., Tan, K. C., & Jin, Y. (2021). Evolutionary large-scale multi-objective optimization: A survey. *ACM Computing Surveys (CSUR)*, 54(8), 1-34.
- Turing, A. M. (2009). Computing machinery and intelligence (pp. 23-65). *Springer Netherlands*.
- Ullah, A., Javaid, N., Asif, M., Javed, M. U., & Yahaya, A. S. (2022). Alexnet, adaboost and artificial bee colony based hybrid model for electricity theft detection in smart grids. *IEEE Access*, 10, 18681-18694.
- Varughese, D. A., & Sridevi, S. (2024). DFPT-CNN: A Dual feature extraction and pretrained CNN synergy for minimal computational overhead and enhanced accuracy in multi-class medical image classification. *IEEE Access*.
- Vetrivel, S. C., Mohanasundaram, T., Saravanan, T. P., & Maheswari, R. (2024). Impact of AI Adoption in Current Trends of the Financial Industry. *Artificial Intelligence for Risk Mitigation in the Financial Industry*, 103-131.
- Vitale, S., Ferraioli, G., & Pascazio, V. (2020). Multi-objective CNN-based algorithm for SAR despeckling. *IEEE Transactions on Geoscience and Remote Sensing*, 59(11), 9336-9349.
- Wang, B., Sun, Y., Xue, B., & Zhang, M. (2019, July). Evolving deep neural networks by multi-objective particle swarm optimization for image classification. *In Proceedings of the genetic and evolutionary computation conference* (pp. 490-498).
- Wang, S., & Summers, R. M. (2012). Machine learning and radiology. *Medical image analysis*, 16(5), 933-951.
- Wang, S., Zhang, W., Cao, G., & Huang, Y. (2024). Functional data analysis using deep neural networks. *Wiley Interdisciplinary Reviews: Computational Statistics*, 16(4), e70001.
- Wang, Y., Yan, J., Yang, Z., Zhao, Y., & Liu, T. (2021). Optimizing GIS partial discharge pattern recognition in the ubiquitous power internet of things context: A MixNet deep learning model. *International Journal of Electrical Power & Energy Systems*, 125, 106484.
- Wang, Y., Zhang, H., & Zhang, G. (2019). cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. *Swarm and Evolutionary Computation*, 49, 114-123.
- Wei, C., Zhang, X., Zhang, C., & Song, Z. (2024). Fault Diagnosis of Ultra-Supercritical Thermal Power Units Based on Improved ICEEMDAN and LeNet-5. *IEEE Transactions on Instrumentation and Measurement*.
- Wei, G., Li, G., Zhao, J., & He, A. (2019). Development of a LeNet-5 gas identification CNN structure for electronic noses. *Sensors*, 19(1), 217.

- Wei, H., Lee, F., Hu, C., & Chen, Q. (2022). MOO-DNAS: Efficient neural network design via differentiable architecture search based on multi-objective optimization. *IEEE Access*, 10, 14195-14207.
- Wierichs, D., Gogolin, C., & Kastoryano, M. (2020). Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer. *Physical Review Research*, 2(4), 043246.
- Won, C. S. (2020). Multi-scale CNN for fine-grained image recognition. *IEEE Access*, 8, 116663-116674.
- Wu, H., & Gu, X. (2015a). Max-pooling dropout for regularization of convolutional neural networks. In *Neural Information Processing: 22nd International Conference, ICONIP 2015, Istanbul, Turkey, November 9-12, 2015, Proceedings, Part I* 22 (pp. 46-54). Springer International Publishing.
- Wu, H., & Gu, X. (2015b). Towards dropout training for convolutional neural networks. *Neural Networks*, 71, 1-10.
- Wu, Z., Shen, C., & Van Den Hengel, A. (2019). Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern recognition*, 90, 119-133.
- Xin, R., Zhang, J., & Shao, Y. (2020). Complex network classification with convolutional neural network. *Tsinghua Science and technology*, 25(4), 447-457.
- Yang, Y., Liu, J., & Tan, S. (2020). A constrained multi-objective evolutionary algorithm based on decomposition and dynamic constraint-handling mechanism. *Applied Soft Computing*, 89, 106104.
- Yousefi, M., & Hansen, J. H. (2020). Block-based high performance CNN architectures for frame-level overlapping speech detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29, 28-40.
- Yu, K., Tang, G., Chen, W., Hu, S., Li, Y., & Gong, H. (2023). MobileNet-YOLO v5s: An improved lightweight method for real-time detection of sugarcane stem nodes in complex natural environments. *IEEE Access*.
- Yu, Z., Tang, J., & Wang, Z. (2021). GCPS: A CNN performance evaluation criterion for radar signal intrapulse modulation recognition. *IEEE Communications Letters*, 25(7), 2290-2294.
- Yuesheng, F., Jian, S., Fuxiang, X., Yang, B., Xiang, Z., Peng, G., ... & Shengqiao, X. (2021). Circular fruit and vegetable classification based on optimized GoogLeNet. *IEEE Access*, 9, 113599-113611.
- Zatarain Cabada, R., Rodriguez Rangel, H., Barron Estrada, M. L., & Cardenas Lopez, H. M. (2020). Hyperparameter optimization in CNN for learning-centered emotion recognition for intelligent tutoring systems. *Soft Computing*, 24(10), 7593-7602.
- Zeng, M., & Xiao, N. (2019). Effective combination of DenseNet and BiLSTM for keyword spotting. *IEEE Access*, 7, 10767-10775.

- Zhang, K., Guo, Y., Wang, X., Yuan, J., & Ding, Q. (2019). Multiple feature reweight densenet for image classification. *IEEE access*, 7, 9872-9880.
- Zhang, M., Li, H., Pan, S., Lyu, J., Ling, S., & Su, S. (2021). Convolutional neural networks-based lung nodule classification: A surrogate-assisted evolutionary algorithm for hyperparameter optimization. *IEEE Transactions on Evolutionary Computation*, 25(5), 869-882.
- Zhang, T., Li, F., Zhao, X., Qi, W., & Liu, T. (2022). A convolutional neural network-based surrogate model for multi-objective optimization evolutionary algorithm based on decomposition. *Swarm and Evolutionary Computation*, 72, 101081.
- Zhang, R., Yao, W., Shi, Z., Ai, X., Tang, Y., & Wen, J. (2023). Encoding time series as images: A robust and transferable framework for power system DIM identification combining rules and VGGNet. *IEEE Transactions on Power Systems*, 38(6), 5781-5793.
- Zhang, Y., Hu, H., Li, T., Chen, B., Tian, J., & Lei, S. (2024). Array Nulling Synthesis based on Hyper-Parameter Optimized Self-Paced Learning Convolutional Neural Networks. *IEEE Transactions on Antennas and Propagation*.
- Zhang, Z., Li, L., & Lu, J. (2024). Multiobjective visual evolutionary neural network and related convolutional neural network optimization. *Expert Systems with Applications*, 243, 122751.
- Zhang, Z., Wu, Z., Zhang, H., & Wang, J. (2022). Meta-learning-based deep reinforcement learning for multiobjective optimization problems. *IEEE Transactions on Neural Networks and Learning Systems*, 34(10), 7978-7991.
- Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. (2024). A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, 57(4), 99.
- Zhao, Y. P., Yang, T. L., & Tan, H. J. (2023). Identification of Supersonic Inlet Unstarted State and Optimal Sensor Configuration Based on EI-SPEA2. *Journal of Aerospace Engineering*, 36(2), 04022131.
- Zhou, J., Zou, J., Zheng, J., Yang, S., Gong, D., & Pei, T. (2021). An infeasible solutions diversity maintenance epsilon constraint handling method for evolutionary constrained multiobjective optimization. *Soft Computing*, 25(13), 8051-8062.

ÖZGEÇMİŞ

Sefa ARAS, İlköğretim eğitimi Araklı Cumhuriyet İlköğretim Okulu'nda, lise eğitimini ise Kanuni Anadolu Lisesi'nde tamamlamıştır. 2009 yılında Gazi Üniversitesi Bilgisayar Mühendisliği Bölümü'nde başladığı lisans eğitimini 2014 bahar döneminde başarıyla tamamlamış ve aynı yılın Kasım ayında askerlik görevine başlamıştır. 2015 yılının Nisan ayında askerlik hizmetini tamamladıktan sonra, aynı yılın güz döneminde Karadeniz Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü'nde yüksek lisans eğitimine ve Yazılım Mühendisliği Bölümü'nde araştırma görevlisi olarak görev yapmaya başlamıştır. 2018 yılında yüksek lisans eğitimini tamamlamış olup, aynı yıl Karadeniz Teknik Üniversitesi'nde doktora eğitimine başlamış ve halen bu eğitime devam etmektedir.

Kahraman, H. T., Aras, S., & Gedikli, E. (2020). Fitness-distance balance (FDB): a new selection method for meta-heuristic search algorithms. *Knowledge-Based Systems*, 190, 105169.

Aras, S., Gedikli, E., & Kahraman, H. T. (2021). A novel stochastic fractal search algorithm with fitness-distance balance for global numerical optimization. *Swarm and Evolutionary Computation*, 61, 100821.

Duman, S., Kahraman, H. T., Guvenc, U., & Aras, S. (2021). Development of a Lévy flight and FDB-based coyote optimization algorithm for global optimization and real-world ACOPF problems. *Soft Computing*, 25, 6577-6617.

Guvenc, U., Duman, S., Kahraman, H. T., Aras, S., & Katı, M. (2021). Fitness–Distance Balance based adaptive guided differential evolution algorithm for security-constrained optimal power flow problem incorporating renewable energy sources. *Applied Soft Computing*, 108, 107421.

Kahraman, H. T., Bakir, H., Duman, S., Katı, M., Aras, S., & Guvenc, U. (2022). Dynamic FDB selection method and its application: modeling and optimizing of directional overcurrent relays coordination. *Applied Intelligence*, 1-36.

Duman, S., Kahraman, H. T., Sonmez, Y., Guvenc, U., Kati, M., & Aras, S. (2022). A powerful meta-heuristic search algorithm for solving global optimization and real-world solar photovoltaic parameter estimation problems. *Engineering Applications of Artificial Intelligence*, 111, 104763.

Gürgeç, S., Kahraman, H. T., Aras, S., & Altın, İ. (2022). A comprehensive performance analysis of meta-heuristic optimization techniques for effective organic rankine cycle design. *Applied Thermal Engineering*, 213, 118687.

Kahraman, H. T., Kati, M., Aras, S., & Taşci, D. A. (2023). Development of the Natural Survivor Method (NSM) for designing an updating mechanism in metaheuristic search algorithms. *Engineering Applications of Artificial Intelligence*, 122, 106121.

Sonmez, Y., Duman, S., Kahraman, H. T., Kati, M., Aras, S., & Guvenc, U. (2024). Fitness-distance balance based artificial ecosystem optimisation to solve transient stability constrained optimal power flow problem. *Journal of Experimental & Theoretical Artificial Intelligence*, 36(5), 745-784.

