Hiperparametre Optimizasyonu Destekli LSTM ve RNN Tabanlı Sezgisel Algoritmalar ile DeepFake Ses Tespiti

Tolun KELEŞ

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Ocak 2026

Deepfake Audio Detection Using Hyperparameter Optimization–Assisted LSTM- and RNN-Based Heuristic Algorithms

Tolun KELEŞ

MASTER OF SCIENCE THESIS

Department of Computer Engineering

Ocak 2026

Hiperparametre Optimizasyonu Destekli LSTM ve RNN Tabanlı Sezgisel Algoritmalar ile DeepFake Ses Tespiti

Tolun KELEŞ

Eskişehir Osmangazi Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Eğitim ve Öğretim Yönetmeliği Uyarınca
Bilgisayar Mühendisliği Anabilim Dalı
Yapay Zeka Bilim Dalında
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

Danışman: Prof. Dr. İdiris DAĞ

Ocak 2026

ONAY

Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans öğrencisi **Tolun KELEŞ**'in YÜKSEK LİSANS tezi olarak hazırladığı "Hiperparametre Optimizasyonu Destekli LSTM ve RNN Tabanlı Sezgisel Algoritmalar ile DeepFake Ses Tespiti" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğin ilgili maddeleri uyarınca değerlendirilerek oybirliği ile kabul edilmiştir.

Danışman : Prof. Dr. İdiris DAĞ

İkinci Danışman :-

Yüksek Lisans Tez Savunma Jürisi:

Üye: Prof. Dr. İdiris DAĞ

Üye: Dr. Öğr. Üyesi Zuhal CAN

Üye: Dr. Öğr. Üyesi Nagihan YAĞMUR

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ……..…….. tarih ve …..... sayılı kararıyla onaylanmıştır.

Prof. Dr. ....

Enstitü Müdürü

ETİK BEYAN

Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kılavuzuna göre, Prof. Dr. İdiris DAĞ danışmanlığında hazırlamış olduğum "Hiperparametre Optimizasyonu Destekli LSTM ve RNN Tabanlı Sezgisel Algoritmalar ile DeepFake Ses Tespiti" başlıklı Yüksek Lisans tezimin özgün bir çalışma olduğunu; tez çalışmamın tüm aşamalarında bilimsel etik ilke ve kurallarına uygun davrandığımı; tezimde verdiğim bilgileri, verileri akademik ve bilimsel ilke ve kurallara uygun olarak elde ettiğimi; tez çalışmamda yararlandığım eserlerin tümüne atıf yaptığımı ve kaynak gösterdiğimi ve bilgi, belge ve sonuçları bilimsel etik ilke ve kurallara göre sunduğumu beyan ederim. 22/01/2026

Tolun KELEŞ

[redacted]

İmza

T.C.
ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ÜRETKEN YAPAY ZEKÂ KULLANIMI BEYAN FORMU

Bu tezi/projeyi hazırlarken aşağıdaki Üretken Yapay Zekâ (ÜYZ) araçlarından destek aldığımı beyan ederim:

Aşağıdaki ÜYZ araçlarını kullandım. (Birden fazla seçenek işaretlenebilir)

☐ Claude
☑ ChatGPT
☐ DeepSeek
☐ Gemini
☐ Copilot
☐ Diğer (Belirtiniz): .....

ÜYZ araçlarını tezimin/projemin aşağıdaki aşamalarında ilgili amaçlar için kullandım. (Birden fazla seçenek işaretlenebilir**.**

☑ Metin oluşturma
☑ Literatür tarama
☑ Çeviri
☐ Düzenleme/dil kontrolü
☑ Veri işleme ve analizi
☐ Veri görselleştirme
☐ Görüntü üretme
☐ Kaynakça düzenleme
☐ Diğer (Belirtiniz): ……

Bu çalışmada ÜYZ kullanımı ***Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Faaliyetlerinde Üretken Yapay Zekâ Kullanımına Dair Etik Rehber***'e ve akademik dürüstlük ilkelerine uygun şekilde gerçekleştirilmiş olup ÜYZ araçlarından alınan destek etik kurallar ve kaynak gösterme esaslarına uygun biçimde belirtilmiştir.

ÜYZ araçlarının sağladığı bilgilerin doğruluğunu kontrol ettiğimi, yanlış veya eksik bilgi bulunması halinde tüm sorumluluğun tarafıma ait olduğunu; ayrıca çalışmamla ilgili bu beyana aykırı bir durumun herhangi bir zamanda tespit edilmesi durumunda ortaya çıkabilecek akademik ve hukuki sonuçları kabul ettiğimi beyan ederim.

Tolun KELEŞ
19/02/2026

(İmza)

ÖZET

İnsan konuşmasının derin öğrenme ve doğal dil işleme yoluyla yapay olarak taklit edilmesini mümkün kılan Deepfake ses sentezi, bilgi güvenliği ve dijital güven açısından giderek artan riskler oluşturmaktadır. Üretilen seslerin yüksek gerçekçiliği ve çeşitliliği nedeniyle bu tür sentetik sesleri tespit etmek hâlâ zorlu bir görevdir. Bu çalışma, model performansını artırmak amacıyla sezgisel (heuristic) optimizasyon algoritmalarını derin öğrenme mimarileriyle bütünleştiren gelişmiş bir Deepfake ses tespit çerçevesi önermektedir. Özellikle, Tekrarlayan Sinir Ağı (RNN) ve Uzun Kısa Süreli Bellek (LSTM) modelleri; Genetik Algoritma (GA), Parçacık Sürüsü Optimizasyonu (PSO), Fark Evrimi (DE) ve Jaya dâhil olmak üzere on sezgisel ve meta-sezgisel algoritma kullanılarak optimize edilmiştir. 11.536 örnekten (8.275 sahte ve 3.260 gerçek) oluşan ADD 2022 veri seti, eğitim ve test süreçleri için spektrogram tabanlı sayısal özelliklere dönüştürülmüştür. Her sezgisel algoritma, öğrenme oranı, dropout oranı, girdi boyutu ve batch boyutu gibi hiperparametreleri optimize ederek tespit doğruluğunu en üst düzeye çıkarmayı hedeflemiştir. Deneysel sonuçlar, önemli performans artışlarını göstermektedir; RNN–PSO ve LSTM–DE kombinasyonları sırasıyla %99,0 ve %97,2 doğruluk oranlarına ulaşarak temel modele göre %20'nin üzerinde bir iyileşme sağlamıştır. Bu bulgular, sezgisel optimizasyonun Deepfake ses tespitinde genelleme ve yakınsama verimliliğini önemli ölçüde artırdığını göstermektedir. Önerilen yaklaşım, diğer derin öğrenme tabanlı medya doğrulama görevlerine de uyarlanabilir ölçeklenebilir bir optimizasyon çerçevesi sunmaktadır.

Anahtar Kelimeler: DeepFake Ses, Derin Öğrenme, Sezgisel Algoritmalar

SUMMARY

Deepfake voice synthesis, which enables the artificial replication of human speech through deep learning and natural language processing, poses increasing risks to information security and digital trust. Detecting such synthetic voices remains a challenging task due to the high realism and variability of generated speech. This study proposes an enhanced Deepfake voice detection framework that integrates heuristic optimization algorithms with deep learning architectures to improve model performance. Specifically, Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) models were optimized using nine heuristic and meta-heuristic algorithms, including Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), and Jaya. The ADD 2022 dataset—comprising 11,536 samples (8,275 fake and 3,260 real)—was preprocessed into spectrogram-based numerical features for training and testing. Each heuristic algorithm performed hyperparameter optimization over learning rate, dropout rate, input size, and batch size to maximize detection accuracy. Experimental results demonstrate substantial performance gains, with RNN–PSO and LSTM–DE combinations achieving 99.0% and 97.2% accuracy, respectively—an improvement of over 20% compared to the baseline model. These findings indicate that heuristic optimization significantly enhances generalization and convergence efficiency in Deepfake voice detection. The proposed approach contributes a scalable optimization framework adaptable to other deep learning-based media authentication tasks.

Keywords: DeepFakeVoice, Deep Learning, Heuristic Algorithms

İÇİNDEKİLER

Sayfa

ÖZET ..... **vii**
SUMMARY ..... **viii**
İÇİNDEKİLER ..... **ix**
ŞEKİLLER DİZİNİ ..... **xi**
ÇİZELGELER DİZİNİ ..... **xii**
SİMGELER VE KISALTMALAR DİZİNİ ..... **xiii**
1. GİRİŞ VE AMAÇ **1**
2. LİTERATÜR ARAŞTIRMASI **5**
2.1. DeepFake Teknolojileri ve Ses Tespiti 5
2.2. Derin Öğrenme Modelleri 9
2.2.1. RNN 9
2.2.2. LSTM 10
2.2.2. RNN ve LSTM Karşılaştırması 11
2.3. Sezgisel Algoritmalar ve Optimizasyon 12
2.3.1. Parçacık Sürü Optimizasyonu 15
2.3.2. Harris Hawks Optimizasyonu 16
2.3.3. Jaya Algoritması 17
2.3.4. Çiçek Tozlaşma Algoritması 18
2.3.5. Genetik Algoritma 19
2.3.6. Karınca Aslanı Algoritması 21
2.3.7. Diferansiyel Evrim Algoritması 22
2.3.8. Çapraz Entropi Metodu 23
2.3.9. Aritmetik Optimizasyon Algoritması 24
3. YÖNTEM **26**
3.1. Veri Kümesi ve Ön İşleme 26
3.2. Derin Öğrenme Modellerinin Tasarımı 26
3.3. Derin Öğrenme Modellerinin İşleyişi 26
2.3.1. RNN 26

İÇİNDEKİLER

Sayfa

2.3.2. LSTM 28
3.4. Sezgisel Algoritmalar ve Parametre Optimizasyonu 29
3.5. Sezgisel Algoritmaların İşleyişi 30
2.5.1. PSO 30
2.5.2. HHO 31
2.5.3. Jaya 33
2.5.4. FPA 33
2.5.5. GA 34
2.5.6. ALO 35
2.5.7. DE 36
2.5.8. CEM 37
2.5.9. AOA 38
3.4. Deney Ortamı 38
4. BULGULAR VE TARTIŞMA 40
5. SONUÇ VE ÖNERİLER 45
KAYNAKLAR DİZİNİ 46

ŞEKİLLER DİZİNİ

Sekil **Sayfa**

2.1. Meta sezgisel algoritmalar 14

2.2. RNN yapısı 27

2.3. LSTM yapısı 28

3.1. Akış diyagramı 39

4.1. Performans grafikleri 43

ÇİZELGELER DİZİNİ

Cizelge **Sayfa**

4.1. Çıkarılan öznitelikler 41

4.2. Hiperparametre değerleri… 41

4.3. Sonuç tablosu… 42

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
PSO	Particle Swarm Optimization
HHO	Harris-Hawks Optimization
FPA	Flower Pollination Algorithm
GA	Genetic Algorithm
ALO	Ant Lion Optimization
DE	Differantial Evolution
CEM	Cross-Entropy Method
AOA	Arithmetic Optimization Algorithm

1. GİRİŞ VE AMAÇ

Son yıllarda Deepfake teknolojileri hızla gelişerek, insan görünümünü, davranışını ve sesini taklit eden son derece gerçekçi sentetik medya üretimini mümkün hâle getirmiştir (Almutairi ve Elgibreen, 2022). Video, görüntü, metin ve ses gibi farklı biçimleri arasında **Deepfake ses**, hedef konuşmacının tonunu, prozodisini ve ritmini insan benzeri bir hassasiyetle taklit edebilme yeteneği nedeniyle özel bir ilgi görmektedir (Amezaga ve Hajek, 2022; Khanjani vd., 2023; McUba vd., 2023). Saldırganlar, konuşma kayıtları üzerinde derin öğrenme modelleri eğiterek, gerçek insan konuşmasından ayırt edilemeyen sentetik sesler üretebilmektedir (Khochare vd., 2022). Bu yenilik eğlence, destek teknolojileri ve sanal asistanlar gibi meşru kullanım alanlarına sahip olsa da aynı zamanda kimlik sahtekârlığı, yanlış bilgi yayılımı ve taklit temelli saldırılar gibi ciddi tehditleri de beraberinde getirmektedir (Almutairi ve Elgibreen, 2022; Dixit vd., 2023; Frank ve Schönherr, 2021; Yi vd., 2023).

Ses tabanlı kimlik doğrulama sistemlerinin ve sesli iletişim platformlarının artan kullanımı, bu tür sentetik seslerin oluşturduğu potansiyel riski daha da büyütmektedir (Dixit vd., 2023). Deepfake ses, telefon dolandırıcılıkları, politik dezenformasyon ve sosyal mühendislik saldırılarını kolaylaştırarak etik ve güvenlik açısından ciddi zorluklar doğurmaktadır (Frank ve Schönherr, 2021; Shaaban vd., 2023). Görsel Deepfake'lerde sıklıkla yüz kusurları veya tutarsızlıklar yoluyla tespit yapılabilirken, ses tabanlı Deepfake'lerde algısal ipuçlarının eksikliği nedeniyle tespit süreci çok daha zordur (Yi vd., 2023). Konuşma sinyallerindeki küçük manipülasyonların tespiti, frekans, faz ve zamansal özelliklerin karmaşık analizini gerektirmekte; mevcut yöntemler bu alanlarda hâlâ yetersiz kalmaktadır (Martín-Doñas ve Álvarez, 2022; Yi vd., 2023).

Geleneksel Deepfake ses tespit yaklaşımları, çoğunlukla Mel-Frekans Kepstrum Katsayıları (MFCC), spektrogramlar veya kromagramlar gibi elle çıkarılmış özellikler üzerinde eğitilmiş Evrişimsel Sinir Ağları (CNN) veya Tekrarlayan Sinir Ağları (RNN) kullanmaktadır. Bu modeller belirli başarılar elde etmiş olsa da, performansları hiperparametre yapılandırmalarına (örneğin öğrenme oranı, dropout oranı, katman boyutu) son derece duyarlıdır (McUba vd., 2023). Yanlış parametre ayarları, aşırı uyuma

(overfitting), yavaş yakınsama veya düşük tespit doğruluğu gibi sorunlara yol açmaktadır (Almutairi ve Elgibreen, 2022). Ayrıca, CNN tabanlı mimariler konuşmadaki uzun vadeli zamansal bağımlılıkları yakalamakta yetersiz kalmakta, daha çok yerel özelliklere odaklandıkları için yüksek tutarlılığa sahip sentetik sesler karşısında etkinlikleri sınırlı kalmaktadır (Yi vd., 2023).

Bu sınırlamaların üstesinden gelmek için hiperparametre optimizasyonu, modelin sağlamlığını ve genelleme kabiliyetini artırmada kritik bir adım hâline gelmiştir (Gu vd., 2015). Ancak klasik ızgara arama (grid search) veya rastgele arama (random search) yöntemleri, derin öğrenmeye özgü yüksek boyutlu arama uzaylarında hesaplama açısından maliyetli ve verimsizdir (Rawat ve Wang, 2017). Son dönemde Genetik Algoritma (GA), Parçacık Sürüsü Optimizasyonu (PSO), Harris Şahin Optimizasyonu (HHO), Fark Evrimi (DE) ve Jaya gibi sezgisel ve meta-sezgisel algoritmalar; geniş, doğrusal olmayan parametre uzaylarını verimli bir şekilde keşfetme konusundaki dikkat çekici yetenekleriyle öne çıkmıştır (Kumar ve Bawa, 2020). Bu algoritmalar, doğadaki veya sosyal sistemlerdeki süreçleri (örneğin evrim, sürü davranışı, tozlaşma) taklit ederek kapsamlı bir tarama yapmadan optimuma yakın çözümlere ulaşabilmektedir (Beheshti vd., 2013). Derin öğrenme mimarileriyle entegre edildiklerinde, tespit doğruluğunu ve kararlılığını en üst düzeye çıkaran hiperparametre yapılandırmalarını otomatik olarak bulabilmektedirler (Hussien vd., 2022).

Daha önce yapılan bazı çalışmalar, derin öğrenme yöntemleri kullanarak Deepfake ses tespitini incelemiştir. Almutairi ve Elgibreen, (2022), modern tespit yöntemlerini inceleyerek ASVspoof 2019 ve AR-DAD veri setlerinde CNN tabanlı modellerin %85,99 ile %94,33 arasında doğruluk sağladığını bildirmiştir (Almutairi ve Elgibreen, 2022). Dixit vd. (2023) ise CNN ve BiLSTM mimarilerini değerlendirerek, sırasıyla Arabic Diversity ve H-Voice veri setlerinde %94,33 ve %91,00 doğruluk oranlarına ulaşmıştır (Dixit vd., 2023). Bu çalışmalar önemli bir temel oluşturmakla birlikte, sezgisel optimizasyonun Deepfake tespit performansı üzerindeki etkisini sistematik biçimde inceleyen araştırmalar sınırlı kalmıştır. Bu nedenle, algoritmik optimizasyon ile uygulamaya özgü doğruluk artışı arasında metodolojik bir boşluk bulunmaktadır.

Bu çalışma, söz konusu boşluğu gidermeyi amaçlayarak, RNN ve LSTM mimarilerini sezgisel ve meta-sezgisel optimizasyon algoritmalarıyla entegre ederek Deepfake ses tespit performansını artırmayı hedeflemektedir. ADD 2022 veri seti kullanılarak gerçekleştirilen bu yaklaşım, öğrenme oranı, dropout oranı, girdi boyutu ve batch boyutunu kapsayan hiperparametre optimizasyonunu yinelemeli sezgisel arama yoluyla gerçekleştirmiştir. Çalışmanın temel amacı, bu algoritmaların modelin yakınsama hızına ve tespit doğruluğuna olan etkisini değerlendirmektir. Optimize edilmiş ve edilmemiş modeller karşılaştırılarak, sezgisel temelli optimizasyonun tespit güvenilirliğini önemli ölçüde artırırken hesaplama verimliliğini koruduğuna dair deneysel kanıtlar sunulmuştur.

Bu çalışmanın temel amacı, DeepFake Seslerin tespitinde kullanılan derin öğrenme algoritmalarının performansını artırmak için sezgisel algoritmalar kullanmaktır. Bu algoritmalar ile parametre optimizasyonu gerçekleştirilmiştir. Derin öğrenme algoritmalarının (LSTM, CNN, RNN vb.) DeepFake Ses tespitindeki etkinliğini değerlendirmek, sezgisel algoritmalar (GA, HHO, PSO vb.) kullanarak derin öğrenme model parametrelerini optimize etmek ve bu optimizasyonların performans üzerindeki etkilerini incelemek; DeepFake Ses Tespiti için geliştirilen yöntemlerin doğruluk, kesinlik, geri çağırma ve F1-skoru gibi performans ölçütleriyle değerlendirilmesi amaçlanmıştır.

Bu çalışmanın bir diğer amacı ise geleneksel sezgisel algoritmalar ile son yıllarda ortaya çıkan yeni nesil sezgisel algoritmaları aynı veri seti üzerinde karşılaştırarak performans analizi yapmaktır. Böylece hangi derin öğrenme yönteminde hangi sezgisel algoritmanın daha iyi parametre optimizasyonu yaptığı sorusunun cevabı aranarak bu alandaki çalışmalara ışık tutulacaktır. Bu araştırma, DeepFake ile geliştirilen yapay seslerin tam ve doğru bir şekilde tespit etmeye yönelik yenilikçi bir yaklaşım sunmayı hedeflemektedir.

Bu araştırmanın temel soruları aşağıdaki gibi sıralanabilir:

- Derin öğrenme algoritmalarından (LSTM, RNN vb.) hangisi, DeepFake ses tespitinde en yüksek doğruluğu sağlamaktadır?
- Sezgisel algoritmalar (GA, HHO, PSO vb.) kullanılarak yapılan parametre optimizasyonları, DeepFake ses tespitindeki model performansını nasıl etkiler?
- Derin öğrenme ve sezgisel algoritmaların birleşimi, yalnızca derin öğrenme algoritmalarına kıyasla daha iyi tespit sonuçları sunmakta mıdır?

Bu çalışmanın kapsamı, DeepFake seslerin tespitine yönelik derin öğrenme algoritmalarının ve sezgisel algoritmaların birlikte çalıştırılmasına dayanmaktadır. Araştırmada, ses verilerinden elde edilen özellikler kullanılarak LSTM ve RNN gibi derin öğrenme modelleri ile DeepFake Sesler tespit edilecektir. Bu modellerin hiperparametre optimizasyonu için ise GA, HHO ve PSO gibi sezgisel yöntemler kullanılacaktır. Çalışma, DeepFake Seslerin tespiti üzerine yapılan çalışmalara katkıda bulunmayı ve bu alandaki eksiklikleri gidermeyi amaçlamaktadır.

2. LİTERATÜR ARAŞTIRMASI

2.1. DeepFake Teknolojileri ve Ses Tespiti

DeepFake, yapay zekâ ve derin öğrenme yöntemleri kullanılarak gerçekçi ses ve görüntüler üretebilen teknolojilerdir (McUba vd., 2023). Bu teknolojiler, özellikle ses alanında, belirli bir kişinin ses tonunu, tarzını ve özgün özelliklerini taklit edebilir (Amezaga ve Hajek, 2022). DeepFake sesleri genellikle eğlence, eğitim ve tehlikeli kullanımlar gibi farklı alanlarda kullanılmaktadır (Almutairi ve Elgibreen, 2022). Seslendirme ve film endüstrisinde karakterlere gerçekçi sesler verme, bankacılık ve müşteri hizmetlerinde sesli asistanlar, tarihi figürlerin seslerinin simüle edilmesiyle eğitim materyallerinin zenginleştirilmesi gibi olumlu uygulamaların yanı sıra, yanlış bilgilendirme, dolandırıcılık ve manipülasyon amaçlarıyla sahte ses üretimi gibi olumsuz kullanımlar da söz konusudur (Yi vd., 2023).

DeepFake ses teknolojileri genellikle iki temel aşamada işler (Yi vd., 2023). Birinci aşamada, ses verisinin analiz edilmesi ve orijinal sesin özellikleri (tonlama, hız, frekans) çıkarılmasıyla model oluşturulur (Amezaga ve Hajek, 2022). İkinci aşamada ise oluşturulan model yardımıyla sahte sesler üretilir (Shaaban vd., 2023). Son yıllarda geliştirilen teknolojiler yardımıyla, özellikle özgün seslerin birebir taklit edilmesi konusunda çok büyük ilerlemeler kaydetmiştir (Yi vd., 2023).

DeepFake konuşma üretimi TTS (metinden sese) teknolojileri ile de yapılabilmektedir. Tacotron, Google tarafından geliştirilen bir TTS (Text-to-Speech) sistemidir. Bu sistem, metin tabanlı bir girişi alır ve bunu insan sesi gibi duyulan bir çıktıya dönüştürür. Tacotron'ın temel bileşenleri encoder, decoder ve vocoder olarak öne çıkar. Encoder, girdi metin verilerini analiz ederek fonetik ve dilbilgisel özellikleri temsil eden bir gösterim yaratır. Decoder, bu gösterimi kullanarak ses spektrogramları üretir. Son aşamada vocoder, spektrogramlardan ses sinyallerini oluşturur ve sesi insan sesine benzetir. Tacotron, oldukça gerçekçi ve doğal sesler üretebilme yeteneğiyle dikkat çeker. Giriş metindeki duygu durumuna göre özgün ses tonu üretme becerisine sahiptir. Tacotron 2 gibi güncellenmiş

sürümleri, WaveNet gibi gelişmiş vocoder teknolojilerini kullanarak daha kaliteli çıktılar elde etmektedir (Khanjani vd., 2023).

WaveNet, yine Google tarafından geliştirilen ve insan sesine benzeyen çıktılar üretebilen bir yapay sinir ağı modelidir. Bu model, doğrudan ham ses sinyallerini üretir ve sesi dalga formunda taklit eder. WaveNet, ses sinyallerini bir dizi zaman serisi olarak ele alır ve her bir veri noktasını bir öncekine bağlı olarak tahmin eder. Karmaşık dalga formlarını modellemek için doğrusal olmayan aktivasyon fonksiyonları kullanır ve insan sesindeki ince detayları bile yakalayarak daha doğal çıktılar sağlar. WaveNet, geleneksel metinden-sese dönüşüm sistemlerine göre çok daha akıcı ve gerçekçi sesler üretebilmesiyle öne çıkar. Ayrıca farklı ses tonları ve şiveler üzerinde çalışabilme esnekliği sunar. Tacotron ve WaveNet gibi teknolojiler, DeepFake ses üretiminde çok etkili birer aracıdır. Bu modeller, insan sesindeki doğallığı ve çeşitliliği yakalayarak sahte seslerin gerçeğe çok yakın olmasını sağlar. Ancak bu yetenekler, zararlı kullanımlara da kapı açabileceği için etik ve yasal sorunları da beraberinde getirmektedir. Bu nedenle bu teknolojilerin tespit edilmesi ve kontrol altına alınması önem arz etmektedir (Dixit vd., 2023).

Mcuba v.d, (2023)'nin çalışmasında DeepFake ses sınıflandırma sorununu incelenmekte ve adli soruşturmalar için mevcut DeepFake ses algılama yöntemleri değerlendirilmektedir. Ses dosyası özellikleri çıkarılarak, MFCC, Mel-Spektrum, Chromagram ve Spektrogram temsilleri kullanılarak görsel olarak sunulmuş ve farklılıklar incelenmiştir. Mevcut literatürdeki farklı derin öğrenme tekniklerinden yararlanılarak beş yinelemeli test ile karşılaştırma yapılmış, ortalama doğruluk ve etkileri belirlenmiştir. Sonuçlar, Chromagram, Spektrogram ve Mel-spektrum görüntüleri için Özel Mimari'nin daha iyi sonuçlar verdiğini; MFCC görüntü özelliği için ise VGG-16 mimarisinin en iyi sonuçları sağladığını göstermiştir. Bu makale, adli soruşturmacılara sentetik ve gerçek sesleri ayırt etmede daha fazla yardımcı olmayı amaçlamaktadır.

Khanjani v.d, (2023)'nin çalışmasında farklı DeepFake kategorileri, bunların nasıl oluşturulup tespit edilebileceği, bu alandaki en yeni eğilimler ve tespit yöntemlerindeki eksiklikler, çalışmanın ana odağı olan DeepFake Seslerin nasıl oluşturulup tespit edildiğine dair daha ayrıntılı bilgilerin özeti sunulmuştur. Değerlendirmede, Generative Adversarial Networks (GAN), Convolutional Neural Networks (CNN) ve Deep Neural Networks (DNN)

gibi yöntemlerin DeepFake oluşturulması ve tespiti için yaygın bir şekilde kullanıldığı görülmüştür. İncelenen 140'tan fazla yöntemde, çalışmaların çoğunlukla video DeepFake'lere, özellikle de bunların üretilmesine odaklandığı tespit edilmiştir. Bu makalede DeepFake ile üretilen seslerin tespiti üzerine daha fazla araştırma yapılması gerekliliğini açıkça ortaya konmaktadır. DeepFake Ses, 3 kategoriye ayrılmıştır.

Tekrar Saldırıları (Replay Attacks), hedef bir konuşmacının ses kaydının yeniden oynatılması olarak tanımlanır. Bu saldırının, uzak alan algılama ve kes ve yapıştır algılama olarak iki alt türü bulunmaktadır. Uzak alan algılama saldırılarında, kurbanın sesi uzak bir mikrofonla kaydedilir ve bu kayıt bir telefonun hoparlöründe çalınarak tekrar test edilir. Kes ve yapıştır algılama saldırılarında, metin bağımlı bir sistemin gerektirdiği cümleleri sahte olarak üretmek için kısa kayıtlar kesilip yapıştırılarak sahte bir ses kaydı oluşturulur. Bu tür saldırılara karşı metin bağımlı konuşmacı doğrulama kullanılabilir. Güncel bir yöntem, uçtan uca tekrar saldırılarını tespit etmek için derin evrişimsel ağların (deep convolutional networks) kullanılmasıdır. Bu ağlar, ses kayıtlarının doğal olup olmadığını anlamak için güçlü bir araçtır.

Konuşma Sentezi (Speech Synthesis), insan konuşmasını yapay olarak yazılım veya donanım sistemi programları aracılığıyla üretme işlemidir. Konuşma sentezi, metni analiz ederek girilen metinle uyumlu bir şekilde konuşma sesi üreten ve metin bağımlı dilbilgisel kuralları kullanan Metinden Konuşmaya (TTS - Text-to-Speech) teknolojisini içerir. Metin okuma ve kişisel yapay zekâ asistanı olarak kullanılabilir. Farklı aksanlar ve sesler sunabilir, böylece önceden kaydedilmiş insan seslerine olan ihtiyacı ortadan kaldırır. Önde gelen konuşma sentezi şirketlerinden biri olan Lyrebird, derin öğrenme modelleri kullanarak saniyede 1.000 cümle oluşturabilmektedir.

Konuşma sentezleme sistemleri, kaliteli konuşma corpuslarına dayanır ve bu corpusların oluşturulması oldukça maliyetlidir. Bu sistemler nokta veya özel karakterleri tanımakta zorlanır. En büyük sorunlardan biri, yazılışı aynı ama anlamı farklı olan homograf kelimelerdeki belirsizliklerdir. Örneğin, İngilizce'de "lead" kelimesi hem "öncülük etmek" hem de "kurşun" anlamına gelir. Bu zorluklar, konuşma sentezi teknolojisinin daha gelişmiş hale gelmesi için çözülmesi gereken sorunlardır.

Yi vd., (2023)'nin çalışmasında daha fazla DeepFake türüne yer verilmektedir. Duygu sahtekarlığı (Emotion Fake), konuşmanın içeriği ve konuşmacının kimliği gibi diğer bilgileri sabit tutarken, sesin duygusunu değiştirmeyi hedefler. Örneğin, konuşmacı B tarafından mutlu bir duyguyla söylenen bir ifadeyi alıp, aynı ifadeyi üzüntülü bir duyguya dönüştürmek bu kapsama girer. Ancak, sesin duygusunu değiştirmek genellikle anlamda değişikliklere yol açabilir. Bu tür duygu dönüştürme işlemleri için iki yöntem kullanılmaktadır. Paralel veri tabanlı yöntemler, orijinal ses ile dönüştürülen ses arasında birebir eşleşme bulunan veri kümelerine dayanır. Paralel olmayan veri tabanlı yöntemler: Birebir eşleşen verilerin olmadığı, daha esnek veri kümelerini kullanır. Bu yöntemler, duygusal ses dönüşümü (VC - Voice Conversion) alanında "emotion fake" olarak bilinen uygulamalarda kullanılır. Bir diğer kullanım alanı ise psikolojide hastaların konuşmalarından hangi duygu durumunda olduklarının tespit edilmesidir.

Sahne Sahtekarlığı (Scene Fake), konuşmanın içeriği ve konuşmacının kimliği değişmeden, orijinal ifadenin akustik sahnesinin konuşma iyileştirme teknolojileriyle başka bir sahneye dönüştürülmesi işlemidir. Örneğin, gerçek bir ifadenin akustik sahnesi "Ofis" iken, sahte bir ifadenin akustik sahnesi "Havaalanı" olabilir. Eğer bir orijinal ses kaydının sahnesi başka bir sahneyle manipüle edilirse, sesin özgünlüğü ve bütünlüğünün doğrulanması güvenilmez hale gelir ve hatta orijinal sesin anlamsal içeriği değişebilir. Bu, sahne manipülasyonunun yaratabileceği ciddi güvenlik ve doğruluk sorunlarını ortaya koyar.

Kısmi sahtekarlık (Partially Fake), bir ifadede sadece birkaç kelimenin değiştirilmesine odaklanır. Sahte ifade, orijinal ifadelerin gerçek veya sentezlenmiş ses klipleriyle manipüle edilmesiyle oluşturulur. Orijinal ifade ile sahte kliplerin konuşmacısı aynı kişidir. Sentezlenmiş ses klipleri, konuşmacının kimliğini değiştirmeden ifade oluşturur. Bu yöntem, genellikle konuşmanın genel yapısını ve kimliğini koruyarak küçük değişiklikler yapmayı amaçlar.

Almutairi ve Elgibreen, (2022)'in çalışmalarında mevcut Audio DeepFake (AD) tespit yöntemleri gözden geçirilmiş ve sahte ses veri kümelerine ilişkin karşılaştırmalı bir inceleme yapılmıştır. Makale, öncelikle AD saldırı türlerini tanıtarak, ardından taklit ve sentetik tabanlı DeepFake tespit yöntemlerini ve veri kümelerini detaylandırıp analiz etmektedir. Makale, AD tespit yöntemlerinin benzerliklerini ve farklılıklarını

özetleyerek, yöntem türünün performansı ses özelliklerinden daha fazla etkilediğini göstermektedir. Bu durum, doğruluk ve ölçeklenebilirlik arasında önemli bir ödünleşme olduğunu ortaya koymaktadır. Ayrıca, makalenin sonunda, DeepFake tespit yöntemlerinin potansiyel araştırma yönleri ve zorlukları ele alınmıştır. Araştırma, AD tespiti aktif bir alan olsa da mevcut boşlukların giderilmesi için daha fazla çalışmaya ihtiyaç olduğunu vurgulamaktadır. Bu makale, araştırmacılar için AD literatürünün mevcut durumunu anlamak ve aksanlı sesler veya gerçek dünya gürültüleri içeren sahte sesleri bile tespit edebilecek daha sağlam tespit modelleri geliştirmek için bir başlangıç noktası olabilir.

2.2. Derin Öğrenme Modelleri

2.2.1 RNN

Recurrent Neural Network (RNN), sıralı ve zaman bağımlı veriler üzerinde çalışan özel bir yapay sinir ağı türüdür. RNN'nin temel çalışma prensibi, geçmişteki bilgileri kullanarak mevcut adımda tahmin yapabilmesidir. Bu özellik, özellikle sıralı veri analizi, zaman serisi tahmini, doğal dil işleme (NLP), konuşma tanıma ve video işleme gibi görevlerde oldukça etkili olmasını sağlar. Girdiler arasında bağımlılıkların olduğu durumlarda, RNN'ler geçmiş bilgiyi geri besleme (feedback) bağlantıları sayesinde saklar ve mevcut hesaplamalarda kullanır (Lipton vd., 2015).

RNN'nin avantajlarından biri, önceki zaman adımlarındaki bilgiyi mevcut adıma taşıyabilmesi ve sıralı bağımlılıkları öğrenebilmesidir. Giriş dizisinin uzunluğuna bağlı olarak dinamik bir şekilde çalışması, modeli esnek hale getirir. Zaman serisi tahmini, dil modelleme, makine çevirisi gibi görevlerde etkin bir şekilde kullanılabilir. Ancak, RNN'nin dezavantajları da bulunmaktadır. Geleneksel RNN'ler, uzun diziler üzerinde çalışırken gradyan kaybolması (vanishing gradient) veya patlaması (exploding gradient) gibi sorunlarla karşılaşabilir. Bu durum, modelin uzun vadeli bağımlılıkları öğrenmesini zorlaştırır ve bu nedenle genellikle LSTM (Long Short-Term Memory) veya GRU (Gated Recurrent Unit) gibi daha gelişmiş RNN varyantları tercih edilir. Ayrıca, RNN'lerin sıralı veriler üzerinde adım adım işlem yapması gerektiğinden, diğer ağ türlerine kıyasla daha yavaş olabileceği de unutulmamalıdır (Kaur ve Mohta, 2019).

RNN'nin uygulama alanları oldukça geniştir. Doğal dil işleme alanında makine çevirisi, metin oluşturma ve soru-cevaplama gibi görevlerde kullanılabilir. Zaman serisi tahmini için hisse senedi fiyatları ve hava durumu tahmininde, konuşma tanıma için sesli komutların işlenmesinde ve video analizinde hareket tanıma ve olay algılama gibi görevlerde etkili bir şekilde uygulanabilir. Sanat ve müzik alanlarında ise şiir yazma, müzik besteleme gibi yaratıcı süreçlere katkıda bulunabilir.

Geleneksel RNN'lerin sınırlamalarını aşmak için geliştirilen bazı modeller mevcuttur. LSTM, uzun vadeli bağımlılıkları daha etkili bir şekilde öğrenirken, GRU benzer bir yapıya sahip olmasına rağmen daha az parametre içerir ve bu sayede daha hızlı öğrenme sağlar. İleri ve geri yönde veri işleme yeteneği sunan Bi-directional RNN ise bağlam bilgisini artırarak performansı geliştirebilir. Bunlara ek olarak, attention mekanizması, RNN'lerde belirli zaman adımlarının daha önemli olduğunu belirlemeye yardımcı olur. Bu özellik, özellikle makine çevirisi gibi görevlerde modelin odaklanmasını sağlar ve performansı artırır (Mao ve Sejdic, 2023).

2.2.2. LSTM

LSTM (Long Short-Term Memory), zaman serisi veya sıralı veri analizi gibi uzun süreli bağımlılıkları modellemek için tasarlanmış bir tür tekrarlayan sinir ağı (Recurrent Neural Network, RNN) mimarisidir. 1997 yılında Hochreiter ve Schmidhuber tarafından önerilen LSTM, geleneksel RNN'lerin uzun vadeli bağımlılıkları öğrenme konusundaki sınırlamalarını aşar. Özellikle sıralı verilerdeki (zaman serisi, doğal dil işleme, vb.) uzun süreli ilişkileri yakalamada üstün performans gösterir (Van Houdt vd., 2020).

LSTM'nin temel özelliklerinden biri, "hücre durumu" (cell state) adı verilen bir yapıya sahip olmasıdır. Bu yapı, uzun vadeli bilgileri depolamak ve gerektiğinde aktarmak için kullanılır. LSTM, hangi bilginin depolanacağını, unutulacağını ve çıktı olarak kullanılacağını kontrol etmek için kapı mekanizmaları kullanır. Bu kapılar üçe ayrılır: Unutma Kapısı (Forget Gate), hangi bilginin unutulacağını belirler; Giriş Kapısı (Input Gate), hangi yeni bilginin belleğe yazılacağını belirler; Çıkış Kapısı (Output Gate) ise hangi bilginin sonraki duruma aktarılacağını belirler. LSTM hem kısa vadeli hem de uzun vadeli

bağımlılıkları aynı anda modelleyebilir. Geleneksel RNN'lerde uzun diziler üzerinde gradyanların kaybolması veya patlaması problemi sık görülürken, LSTM'nin hücre durumu ve kapı mekanizmaları bu sorunu büyük ölçüde azaltır.

LSTM, sıradaki olayların çok önceki durumlardan etkilenebileceği durumlarda etkili bir şekilde çalışır. Zaman serisi tahmini, dil modelleme, konuşma tanıma ve video analizi gibi geniş bir uygulama yelpazesinde kullanılabilir. Ancak, geleneksel RNN'lere kıyasla daha fazla parametreye sahip olması nedeniyle daha yüksek hesaplama maliyetine sahiptir ve eğitim süresi daha uzun olabilir.

2.2.3. RNN ve LSTM Karşılaştırılması

Recurrent Neural Networks (RNN) ve Long Short-Term Memory (LSTM) ağları, derin öğrenmede farklı veri türleri ve görevler için tasarlanmış iki temel mimaridir. Bu modellerin güçlü ve zayıf yönleri, performans farklılıkları ve kullanım alanları, görev gereksinimlerine göre değişiklik göstermektedir.

RNN'ler, sıralı verilerde zamansal bağımlılıkları öğrenme yeteneğiyle dikkat çeker. Zaman serisi analizi, metin işleme ve konuşma tanıma gibi görevlerde önceki adımlardan gelen bilgiyi kullanarak etkili tahminler yapabilir. Bununla birlikte, uzun sekanslar üzerinde çalışırken karşılaşılan vanishing gradient ve exploding gradient problemleri, modelin performansını düşürebilir. Ayrıca, sıralı işlem yapısı nedeniyle RNN'ler genellikle daha uzun eğitim ve çıkarım sürelerine sahiptir.

LSTM'ler, RNN'lerin uzun vadeli bağımlılıkları öğrenme konusundaki sınırlamalarını aşmak için tasarlanmıştır. Gating mekanizmaları sayesinde, uzun sekanslardaki ilişkileri etkili bir şekilde öğrenebilir ve vanishing gradient problemlerini azaltabilir. Bu özellikleriyle LSTM'ler, konuşma tanıma, dil modelleme ve uzun sekanslı zaman serisi tahmini gibi görevlerde öne çıkar. Ancak, LSTM'lerin daha karmaşık yapıları, daha fazla hesaplama gücü ve bellek gereksinimi anlamına gelir. Ayrıca, bu ağların eğitimi, hiperparametre ayarlarında daha fazla özen gerektirdiği için genellikle daha uzun sürebilir.

Sıralı veri analizlerinde LSTM'ler, uzun vadeli bağımlılıkları etkili bir şekilde öğrenme kabiliyeti nedeniyle daha başarılıdır. Eğitim süreleri açısından CNN'ler, paralel işlem yetenekleri sayesinde daha hızlı sonuçlar üretirken, RNN ve LSTM gibi sıralı işlem gerektiren modeller daha uzun süreler alır. LSTM'lerin karmaşık yapısı, bu modeli daha fazla kaynak ve dikkatli bir optimizasyon gerektirir.

Bu mimarilerin seçiminde veri türü, görev gereksinimleri ve kaynak kısıtlamaları önemli rol oynar. Örneğin, DeepFake ses tespiti gibi bir görevde, spektrogramlardan özellik çıkarımı için CNN'lerin ve zamansal desenlerin tanınması için LSTM'lerin kombinasyonu, her iki mimarinin güçlü yönlerinden yararlanmayı sağlayabilir. Bu tür hibrit yaklaşımlar, farklı modellerin avantajlarını bir araya getirerek daha yüksek performans sunabilir.

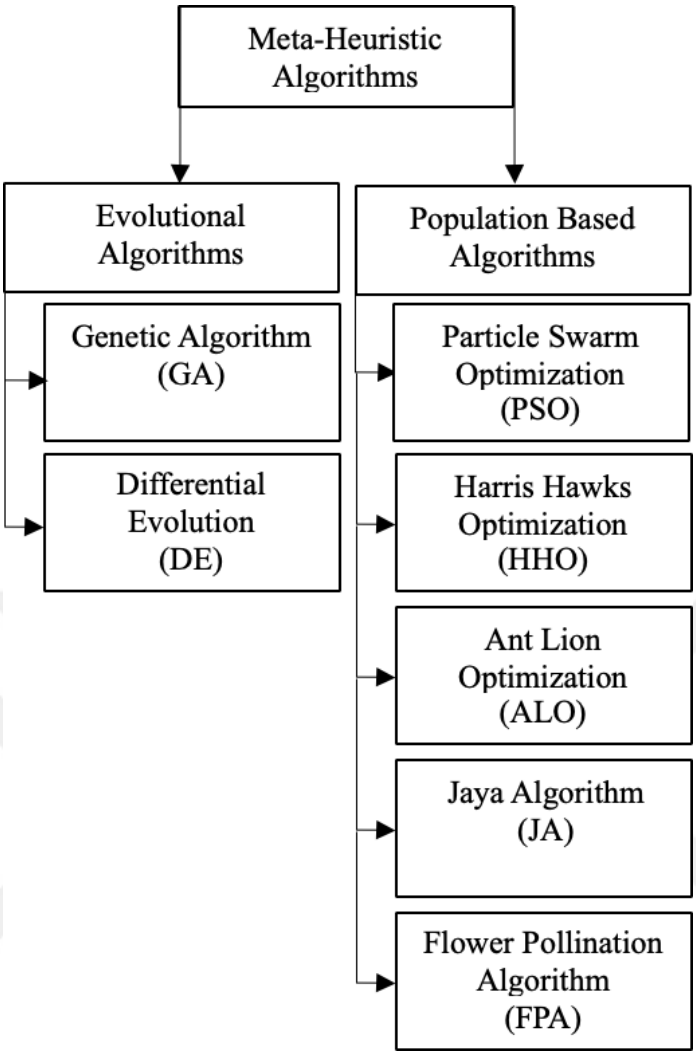
2.3. Sezgisel Algoritmalar ve Optimizasyon

Sezgisel algoritmalar, karmaşık ve çözülmesi zor problemlerde kesin çözüm garantisi sunmayan ancak pratikte kabul edilebilir ve etkili çözümler üreten yöntemlerdir (Desale vd., 2015). Bu algoritmalar, belirli bir süre veya kaynak sınırı dahilinde tatmin edici bir çözüm bulmaya odaklanır (Preuss vd., 2024). "Deneme-yanılma," yaklaşık çözümleme ve problem bazlı ipuçları kullanarak çalışan sezgisel algoritmalar, büyük veri kümeleri ve karmaşık problem yapılarıyla başa çıkma yetenekleriyle öne çıkar (Gandomi vd., 2013). Ancak, elde edilen çözümlerin her zaman en iyi veya optimal olmadığını unutmamak gerekir (Wong ve Ing Ming, 2019).

Sezgisel algoritmaların temel özellikleri, hız ve esneklik etrafında şekillenir (Kenny vd., 2014). Özellikle tam ve gerçek sonucu hesaplamanın çok uzun sürdüğü durumlarda hızlı bir şekilde kabul edilebilir çözümler sunabilir (Preuss vd., 2024). Bu algoritmalar, problemin doğasına özgü ipuçları ve sezgilerden faydalanarak optimum sonuca yakın bir sonuç üretir. Bununla birlikte, global optimum yerine yerel optimal takılma riski taşıdıkları gibi, her problem için özel bir sezgi veya yönteme ihtiyaç duyarlar (Hopper ve Turton, 1999).

Bu algoritmalar, özellikle NP-zor ve NP-tam problemler gibi çözümü zor olan problemlerde geniş bir uygulama alanına sahiptir (Kokash, t.y.). Örneğin, optimizasyon problemlerinde olan Gezgin Satıcı Problemi (TSP), Çizelgeleme Problemi, İnşaat Topolojisi gibi problemlerde etkili çözümler sunarlar (Desale vd., 2015). Makine öğrenmesinde model parametrelerinin optimize edilmesi, yapay zekâ ve robotikte yol planlama ve karar verme süreçleri, mühendislik tasarımında yapı ve sistem optimizasyonu, büyük veri analitiğinde karmaşık veri kümelerinden anlamlı bilgi çıkarma gibi birçok alanda kullanılmaktadır (Preuss vd., 2024).

Sezgisel algoritmalar, kullanılan yöntemlere göre farklı kategorilere ayrılır (Preuss vd., 2024). Yerel arama (Local Search) algoritmaları, mevcut bir çözümden başlayarak küçük değişikliklerle daha iyi çözümler arar (Baghel vd., 2012). Tepelik Tırmanma (Hill Climbing) bu tür algoritmalara örnek olarak verilebilir (Preuss vd., 2024). Evrimsel ve doğadan ilham alan algoritmalar, biyolojik süreçlerden ve doğa olaylarından esinlenir (Beheshti vd., 2013). Genetik Algoritmalar ve Parçacık Sürüsü Optimizasyonu gibi yöntemler bu kategoriye girer (Agrawal vd., 2021). Bu algoritmalar, doğanın evrimsel ve sosyal etkileşim mekanizmalarını taklit ederek optimizasyon problemlerinde etkili çözümler sunar (Ezugwu vd., 2020).



2.1. Meta sezgisel algoritmalar

Sezgisel algoritmalar ile meta-sezgisel algoritmalar arasındaki temel fark, kapsam ve uygulama alanıdır (Desale vd., 2015). Sezgisel algoritmalar, problemin özelliklerine özgüdür ve daha hızlı sonuç verir (Rajpurohit vd., 2017). Buna karşılık, meta-sezgisel algoritmalar daha genel bir çerçeve sunar ve farklı türdeki problemlere uygulanabilir (Preuss vd., 2024). Meta-sezgisel algoritmalar genellikle evrimsel algoritmaları ve doğadan esinlenen yöntemleri içerir (Gandomi vd., 2013).

Örneğin, TSP'nin çözümünde Yakın Komşu Algoritması (Nearest Neighbor Algorithm) gibi sezgisel yöntemler kullanılabilir (Preuss vd., 2024). Bu algoritmada, rastgele bir başlangıç noktası seçilir ve henüz ziyaret edilmemiş en yakın nokta seçilerek süreç tüm noktalar ziyaret edilene kadar devam eder (Preuss vd., 2024). Bu basit yaklaşım, hızlı bir çözüm sunsa da genellikle optimal olmaktan uzaktır (Preuss vd., 2024).

Sonuç olarak, sezgisel algoritmalar, mühendislik, finans, yapay zekâ ve lojistik gibi alanlarda karmaşık problemlere hızlı ve etkili çözümler sunarak kritik bir rol oynar (Preuss vd., 2024). Kesin çözüm gerektirmeyen veya çözüm süresinin sınırlı olduğu durumlarda tercih edilen bu yöntemler, optimizasyon süreçlerinde önemli bir yere sahiptir (Beheshti vd., 2013). Sezgisel algoritmaların kullanımı, problemin doğasına ve gereksinimlerine göre özelleştirildiğinde, gerçek dünya uygulamalarında büyük avantajlar sağlayabilir (Preuss vd., 2024).

2.3.1. Parçacık Sürü Optimizasyonu

Parçacık Sürü Optimizasyonu (PSO), 1995 yılında James Kennedy ve Russell Eberhart tarafından geliştirilmiş, biyolojik sürü davranışlarından ilham alınarak tasarlanmış bir optimizasyon algoritmasıdır (Wang vd., 2018). Doğadaki sürü hareketleri, örneğin kuş sürülerinin, balık sürülerinin ya da arı kolonilerinin yiyecek arama davranışları, PSO'nun temel ilham kaynağını oluşturmuştur (Song ve Gu, 2004).

PSO, bir grup parçacığın (her biri bir aday çözümü temsil eder) bir arama alanı içinde hareket ederek en iyi çözümü aradığı sezgisel bir optimizasyon yöntemidir. Her parçacık, bir pozisyon ve bir hız vektörüne sahiptir (Jain vd., 2018). Parçacıklar, arama alanında hareket ederken iki tür bilgiyi kullanarak bir sonraki adımlarını belirlerler: kendi önceki en iyi konumları (kişisel en iyi pozisyon, pbest) ve grubun bulduğu en iyi konum (global en iyi pozisyon, gbest). Bu bilgiler doğrultusunda hızlarını ve pozisyonlarını güncelleyerek optimum çözüme doğru ilerlemeye çalışırlar.

PSO’nun avantajları arasında basitliği, uygulanabilirliği ve global optimuma yakın çözümler bulma konusundaki etkinliği sayılabilir. Bu özellikler, PSO’yu yapay zeka, makine öğrenimi, mühendislik, ekonomi gibi çeşitli alanlarda optimizasyon problemlerini çözmek için güçlü bir araç haline getirmektedir.

2.3.2. Harris Hawks Optimizasyonu

Harris Hawks Optimizasyonu (Harris Hawks Optimization - HHO), 2019 yılında Ali Asghar Heidari ve ekibi tarafından geliştirilen, Harris şahinlerinin avlanma stratejilerine dayalı bir sezgisel optimizasyon algoritmasıdır (Heidari vd., 2019). Bu algoritma, Harris şahinlerinin doğadaki dinamik ve işbirlikçi avlanma davranışlarını modelleyerek, karmaşık optimizasyon problemlerini çözmeye odaklanır (Alabool vd., 2021).

Harris şahinleri, özellikle tavşanları avlarken kooperatif (işbirlikçi) ve dinamik bir avlanma stratejisi uygularlar (Hussien vd., 2022). HHO, bu stratejiyi taklit eder ve global optimum çözüme ulaşmak için çevreyi keşfetme (exploration) ve çözümü iyileştirme (exploitation) süreçlerini dengeler (Shehab vd., 2022). Şahinler, avın davranışına ve koşullara göre stratejilerini değiştirirler, bu da algoritmanın esnek ve uyarlanabilir olmasını sağlar (Elgamal vd., 2020).

HHO, keşif ve sömürme aşamalarını dinamik bir şekilde dengeleyerek, çözüm uzayında hem geniş çaplı arama yapar hem de uygun çözümlere yaklaştıkça daha dar bir odakta çözümü iyileştirir (C. Y. Li vd., 2021). Algoritma, avın durumuna göre strateji değiştirebildiğinden, farklı problemler için uyarlanabilir ve esnek bir yapıya sahiptir (Kamboj vd., 2020). Yapısal olarak basit olması, HHO'nun anlaşılmasını ve uygulanmasını kolaylaştırır. (Heidari vd., 2019) Algoritma, optimizasyon problemlerinde hızlı sonuçlar verebilme kapasitesine sahiptir (Heidari vd., 2019).

Harris Hawks Optimizasyonu, mühendislik tasarımları, enerji yönetimi, makine öğrenmesi, veri madenciliği ve çeşitli optimizasyon gerektiren uygulamalarda kullanılmaktadır (Qu vd., 2020). Karmaşık ve çok boyutlu optimizasyon problemlerine etkin ve verimli çözümler sunabilir (Heidari vd., 2019).

2.3.3. Jaya Algoritması

Jaya Algoritması, 2016 yılında R. Venkata Rao tarafından geliştirilen, basit ama etkili bir sezgisel optimizasyon algoritmasıdır (Pandey, 2016). Sanskritçe bir kelime olan "Jaya", başarı veya zafer anlamına gelir (Zitar vd., 2022). Algoritma, çözüm uzayında global optimuma ulaşmayı hedeflerken, yalnızca en iyi çözümü bulmaya ve kötü çözümlerden kaçınmaya odaklanır (Warid vd., 2016).

Jaya algoritmasının çalışması için herhangi bir algoritmik parametre ayarına gerek yoktur (Venkata Rao ve Saroj, 2017). Bu, uygulama sürecini basitleştirir ve parametre ayarlaması gerektiren diğer algoritmalara göre avantaj sağlar (Rao vd., 2016). Algoritma, sadece mevcut en iyi çözümü iyileştirmeyi ve kötü çözümlerden uzaklaşmayı temel aldığı için sade bir yapıya sahiptir (Yu vd., 2019).

Algoritma tasarımı ve uygulanması kolaydır. Diğer optimizasyon algoritmalarında kullanılan çaprazlama, mutasyon, hız veya öğrenme oranı gibi parametrelerin ayarlanması gerekmez. Çözüm bulma süreci hızlıdır ve geniş bir uygulama alanına sahiptir. Algoritma, global optimuma ulaşmada güçlüdür ve lokal optimuma takılma olasılığı düşüktür (da Silva vd., 2023).

Algoritma, bazen çok hızlı bir şekilde iyi bir çözüme ulaşmaya çalıştığından, çözüm uzayında daha geniş bir keşif yapamayabilir. Çok karmaşık veya yüksek boyutlu problemlerde ince ayar gerektirebilir (Zhang vd., 2021).

Mekanik, elektronik ve yapısal tasarım problemleri, yapay zekada model parametre optimizasyonu, yenilenebilir enerji kaynaklarının optimizasyonu, makine öğrenmesinde özellik seçimi ve parametre optimizasyonu alanlarında kullanılmaktadır (Pandey, 2016).

Jaya algoritması, basitliği ve etkili sonuçlar sunma potansiyeli sayesinde optimizasyon problemleri için giderek daha popüler hale gelmektedir (Pandey, 2016).

2.3.4. Çiçek Tozlaşma Algoritması

Çiçek Tozlaşma Algoritması (Flower Pollination Algorithm - FPA), doğadan ilham alan bir meta-sezgisel optimizasyon algoritmasıdır (Yang, 2014). 2012 yılında Xin-She Yang tarafından geliştirilmiştir ve bitkilerdeki doğal tozlaşma süreçlerini model alır. Algoritma, küresel ve yerel arama süreçlerini birleştirerek karmaşık optimizasyon problemlerini çözmeyi hedefler (Abdel-Basset ve Shawky, 2019).

Tozlaşma, bitkilerde genetik materyalin (polenin) bir çiçekten diğerine taşınmasıyla gerçekleşir ve bu süreç, bitki üremesi için kritik öneme sahiptir (Yang, 2014).

Polenler uzun mesafelerde taşınır. Bu süreç, rüzgâr, su veya böcekler (örneğin arılar ve kelebekler) gibi taşıyıcılar tarafından sağlanır. Küresel tozlaşma, optimizasyon bağlamında geniş bir çözüm arama stratejisiyle ilişkilidir. Polenler, aynı bitkinin çiçekleri arasında veya yakın çevresindeki çiçekler arasında taşınır. Yerel tozlaşma, optimizasyon bağlamında komşu çözümlerin detaylı araştırılması anlamına gelir. Daha kokulu veya renkli çiçeklerin tozlaşma olasılığı daha yüksektir. Bu durum, uygunluk (fitness) fonksiyonunun bir optimizasyon sürecinde nasıl çalıştığını simgeler (Yang, 2013).

Sürekli, ayrık ve çok modlu optimizasyon problemlerine uygulanabilir. Lévy uçuşları ve yerel arama süreçleri sayesinde çözüm uzayı etkili bir şekilde taranır. Kolayca uygulanabilir ve uyarlanabilir bir yapıya sahiptir (Yang vd., 2014).

Algoritmanın başarısı, parametrelerin (örneğin, p değeri) doğru bir şekilde ayarlanmasına bağlıdır. Bazı durumlarda algoritma, yerel optimumlara sıkışabilir (Yang, 2014).

Mühendislikte mekanik ve elektronik sistemlerin optimizasyonu; yapay zekada model parametre ayarlamaları, yenilenebilir enerji kaynaklarının optimizasyonu, lojistik ve rota planlama, genetik veri analizinde kullanılmaktadır (Yang, 2014).

Çiçek Tozlaşma Algoritması, doğadan ilham alan etkili bir optimizasyon yöntemi olarak öne çıkar (Yang, 2014). Küresel ve yerel arama süreçleri arasındaki denge, algoritmayı birçok farklı problem için uygun hale getirir (Yang, 2014).

2.3.5. Genetik Algoritma

Genetik Algoritma (Genetic Algorithm - GA), doğadan ilham alan bir meta-sezgisel optimizasyon algoritmasıdır. 1975 yılında John Holland tarafından geliştirilmiştir. Bu algoritma, biyolojik evrim süreçlerini (doğal seçilim, çaprazlama, mutasyon) taklit ederek karmaşık optimizasyon problemlerini çözmeyi hedefler (Forrest, 1996).

Genetik Algoritma, Charles Darwin'in "Doğal Seçilim" ilkelerine dayanır. Biyolojik evrim sürecinde türler, çevreye en iyi uyum sağlayan bireylerin genetik materyalini gelecek nesillere aktararak hayatta kalır. GA, bu süreci matematiksel olarak simüle eder (Holland, 1992).

Her birey bir çözümü temsil eder. Bu bireyler, kromozom olarak adlandırılır ve genlerden oluşur. Kromozomlar, bir çözümün kodlanmış hali olup genellikle ikili (binary), tamsayı veya gerçek sayı dizileri şeklindedir. Her bireyin ne kadar "iyi" bir çözüm olduğunu değerlendirmek için kullanılır. Problemin çözüm hedefi doğrultusunda bireylere uygunluk puanı atanır. Daha yüksek uygunluk puanına sahip bireylerin, yeni nesiller oluşturma sürecine katılma olasılığı daha yüksektir (Mitchell, 1995).

Yaygın seçilim yöntemleri; Rulet Tekerleği Seçimi, Turnuva Seçimi ve Elitizm olarak verilebilir. İki ebeveyn bireyin genetik bilgileri birleştirilerek yeni bireyler (çocuklar) oluşturulur. Örneğin, iki birey arasındaki genlerin rastgele bölüşümü veya birleşimiyle yapılır. Çaprazlama türleri, Tek Noktalı Çaprazlama, Çok Noktalı Çaprazlama, Birleştirme Çaprazlaması olarak sıralanabilir (Sastry vd., 2014).

Mutasyon (Mutation) ile rastgele genetik değişiklikler yapılarak çeşitlilik artırılır ve çözüm uzayında farklı bölgeler araştırılır. Mutasyon, bireylerin yerel optimumlardan kaçmasına ve küresel optimuma ulaşmasına yardımcı olur (Mathew, t.y.).

Algoritma, belirli bir durma kriteri gerçekleştiğinde sonlanır. Maksimum iterasyon sayısına ulaşmak. Uygunluk fonksiyonu değerinin belirli bir eşiğe ulaşması. Popülasyonun homojen hale gelmesi.

Sürekli, ayrık ve çok modlu problemler için uygundur. Yerel optimumlardan kaçabilir ve küresel optimuma yaklaşabilir. Popülasyondaki bireylerin bağımsız olarak değerlendirilebilmesi, algoritmanın paralelleştirilmesine olanak tanır.

Büyük popülasyonlar ve çok iterasyonlu süreçler, zaman ve kaynak açısından maliyetli olabilir. Çaprazlama oranı, mutasyon oranı ve popülasyon büyüklüğü gibi parametrelerin doğru seçilmesi önemlidir. Uygun parametreler ayarlanmazsa algoritma, yerel optimuma sıkışabilir.

Mühendislikte makine tasarımı, ağ optimizasyonu, elektrik devrelerinde, yapay zekada model seçimi ve hiperparametre optimizasyonunda, ekonomide portföy optimizasyonu ve risk analizinde, gen sekans analizi ve protein yapı tahmininde; yörünge planlama, kontrol optimizasyonu.

Bir seyahat eden satıcı problemi (Traveling Salesman Problem - TSP) çözmek için Genetik Algoritma kullanılabilir. Şehirlerin bir permütasyonu. Toplam seyahat mesafesi (ne kadar düşük, o kadar iyi). Çaprazlama ve mutasyon, daha kısa yollar keşfetmek için kullanılır.

Genetik Algoritma, geniş bir çözüm uzayını tarayabilme ve karmaşık problemleri çözebilme yeteneği ile güçlü bir optimizasyon yöntemidir. Ancak, doğru parametre seçimi ve hesaplama maliyeti dikkate alınmalıdır.

2.3.6. Karınca Aslanı Algoritması

Ant Lion Optimizer (ALO) veya Türkçe adıyla Karınca Aslanı Algoritması, doğadan ilham alan bir meta-sezgisel optimizasyon algoritmasıdır (Mirjalili, 2015). 2015 yılında Seyedali Mirjalili tarafından geliştirilmiştir. Bu algoritma, karınca aslanlarının (doodlebug olarak da bilinir) avlanma davranışlarını modelleyerek optimizasyon problemlerini çözmeyi amaçlar (Abualigah vd., 2021).

Karınca aslanları, çöl veya kumluk bölgelerde yaşayan yırtıcı böceklerdir. Bu böcekler, avlanmak için kumdan huni şeklinde çukurlar kazarlar. Çukura düşen böcekler (örneğin, karıncalar) çukurun eğimi ve karınca aslanının çabalarıyla kolayca yakalanır. ALO, bu avlanma sürecini taklit ederek optimizasyon sürecini gerçekleştirir (Mirjalili vd., 2017).

Rastgele bir başlangıç popülasyonu oluşturulur (karıncalar ve karınca aslanları). Her bireyin uygunluk (fitness) değeri hesaplanır. Karıncalar rastgele yürüyüş yapar ve karınca aslanlarının oluşturduğu huniye doğru çekilir. Yeni konumlarda uygunluk değerleri hesaplanır. Karınca aslanlarının en iyi konumu (elit çözüm) belirlenir ve diğer karınca aslanlarının bu çözüme yakınlaşması sağlanır. İstenilen durma kriterine ulaşılana kadar (örneğin, belirli bir iterasyon sayısı) süreç tekrarlanır (Mouassa vd., 2017).

Rastgele yürüyüş ve elitizm mekanizmaları sayesinde küresel optimumu bulma olasılığı yüksektir. Birden fazla optimum çözümü olan karmaşık problemlerde etkilidir. Algoritma kolayca uygulanabilir ve farklı problemler için uyarlanabilir (Thawkar vd., 2021).

Büyük boyutlu problemler ve yüksek popülasyon sayısı için maliyetli olabilir. Yetersiz rastgelelik veya uygun olmayan parametre ayarları, algoritmanın yerel optimumlara sıkışmasına neden olabilir (Dinkar ve Deep, 2017).

Mühendislikte parametre optimizasyonu ve ağ tasarımında; yapay zekada model optimizasyonu ve hiperparametre seçiminde; ekonomide portföy optimizasyonu ve tahmin modellerinde, lojistikte yol planlama ve kontrol sistemlerinde; biyolojide genetik veri analizi, protein yapı tahmininde kullanılır.

Bir seyahat eden satıcı problemi (Traveling Salesman Problem - TSP) için ALO şöyle çalışabilir: Karıncalar, şehirleri ziyaret eden farklı rotaları temsil eder; karınca aslanları, mevcut en iyi rotaları temsil eder; algoritma, karıncaların zamanla en kısa rotaya yakınlaşmasını sağlar.

Ant Lion Algoritması, esnek ve güçlü bir optimizasyon aracıdır. Karmaşık ve çok modlu problemleri çözmek için etkili bir yöntemdir. Ancak, doğru parametre ayarı ve problem özelliklerine uygun olarak tasarlanması önemlidir.

2.3.7. Diferansiyel Evrim Algoritması

Diferansiyel Evrim Algoritması (Differential Evolution - DE), 1995 yılında Storn ve Price tarafından geliştirilmiş bir meta-sezgisel optimizasyon yöntemidir. Sürekli ve ayrık optimizasyon problemlerini çözmek için kullanılır ve özellikle gerçek değerli parametre optimizasyonu için güçlü bir araçtır (Opara ve Arabas, 2019).

DE, optimizasyon sürecinde bir grup çözümden (bireylerden) oluşan bir popülasyon kullanır. Bu bireyler çözüm uzayında farklı noktaları temsil eder. Algoritma, evrimsel stratejileri (mutasyon, çaprazlama ve seçim) kullanarak çözümleri iyileştirir. Çözümler, diğer bireyler arasındaki farklardan yararlanılarak iyileştirilir. Bu, rastgele seçilen bireylerin arasındaki farklılıkların yeni çözümler üretmek için kullanıldığı anlamına gelir (Tasoulis vd., 2004).

Matematiksel olarak anlaşılır ve farklı problemlere kolayca uyarlanabilir. Diferansiyel mutasyon, geniş çözüm uzayını taramada etkili bir mekanizmadır. Ölçekleme

faktörü (F) ve çaprazlama oranı (CR) gibi birkaç parametreyle çalışır. Popülasyondaki bireyler bağımsız olarak değerlendirilebildiğinden paralel işlemeye uygundur (M ve Koumoutsakos, t.y.).

F ve CR değerlerinin iyi seçilmesi gerekir. Yanlış ayarlar algoritmanın başarısını düşürebilir. Bazı durumlarda, özellikle düşük çeşitlilikte popülasyonlarla, algoritma yerel optimumda sıkışabilir. Büyük boyutlu popülasyonlar ve yüksek iterasyon sayısı maliyeti artırabilir (M ve Koumoutsakos, t.y.).

Parametre ayarı, sistem tasarımı. Model optimizasyonu, hiperparametre ayarı. Genetik veri analizi, biyolojik modelleme. Portföy optimizasyonu, risk analizi. Kontrol sistemleri, rota planlama.

Diferansiyel Evrim Algoritması, güçlü ve esnek bir optimizasyon yöntemidir. Sürekli optimizasyon problemlerinde genellikle tercih edilir ve doğru parametre ayarlarıyla küresel optimuma ulaşma şansı yüksektir.

2.3.8. Çapraz Entropi Metodu

Çapraz Entropi Metodu (Cross-Entropy Method - CEM), Monte Carlo temelli bir optimizasyon ve nadir olaylar simülasyonu tekniğidir (Kroese vd., 2013). İlk olarak 1997 yılında Rubinstein tarafından geliştirilmiştir ve başlangıçta nadir olayları simüle etmek için kullanılmıştır. Daha sonra, optimizasyon problemlerine uygulanacak şekilde genişletilmiştir. CEM, olasılık dağılımları üzerinden iyi çözümlerin bulunmasını sağlayan bir yöntemdir ve sürekli ya da ayrık problemler için kullanılabilir (De Boer vd., 2005).

Çapraz Entropi Metodu, bir çözüm uzayında en iyi çözümleri bulmak için iki temel fikre dayanır. Çözüm uzayından rastgele örnekler alınır ve bu örnekler uygunluk (fitness) değerine göre değerlendirilir. İyi çözümler, olasılık dağılımının güncellenmesi için kullanılır. Böylece sonraki örnekler daha iyi çözümlere odaklanır. Bu süreç, Kullback-

Leibler Divergence (KL divergence) veya diğer adıyla çapraz entropiyi minimize ederek gerçekleştirilir. Bu, hedef bir dağılıma (iyi çözümlere odaklanan) yakınsamayı ifade eder (Rubinstein, 1999a).

Algoritma matematiksel olarak anlaşılır ve kolay uygulanabilir. Hem sürekli hem de ayrık optimizasyon problemleri için uygulanabilir. İyi bir olasılık dağılımı tasarımıyla, küresel optimuma yakınsama şansı yüksektir. Çözüm örneklemeleri ve uygunluk hesaplamaları paralel olarak yapılabilir (Mannor vd., 2005).

Büyük çözüm uzaylarında ve yüksek boyutlu problemler için maliyetli olabilir. Örnekleme boyutu (N) ve seçilen iyi çözümlerin oranı (k) algoritmanın performansını etkiler. Yetersiz örnekleme veya kötü bir başlangıç olasılık dağılımı, algoritmanın yerel optimumlara sıkışmasına neden olabilir (Rubinstein, 1999b).

Yapay zekâda model parametre optimizasyonu ve hiperparametre seçimi; lojistikte rota optimizasyonu ve tedarik zinciri problemleri, yol planlama ve kontrol sistemlerinde; biyolojide protein yapı tahmini ve genetik analiz; ekonomide risk analizi ve portföy optimizasyonunda kullanılır.

Çapraz Entropi Metodu, basit bir fikre dayanan güçlü bir optimizasyon yöntemidir (Kroese vd., 2013). Monte Carlo örneklemesi ve olasılık dağılımı güncellemeleriyle çözüm uzayında etkili bir şekilde hareket eder ve nadir olay simülasyonu gibi çeşitli alanlarda uygulanabilir (Kroese vd., 2013).

2.3.9. Aritmetik Optimizasyon Algoritması

Aritmetik Optimizasyon Algoritması (Arithmetic Optimization Algorithm - AOA), 2021 yılında Abualigah vd., (2021) tarafından geliştirilmiş bir meta-sezgisel optimizasyon yöntemidir (Abualigah vd., 2021). Bu algoritma, aritmetik operatörlerin (toplama, çıkarma, çarpma, bölme) matematiksel işleyişinden esinlenerek tasarlanmıştır ve küresel

optimizasyon problemlerini çözmek için kullanılır. AOA, hem sürekli hem de ayrık optimizasyon problemlerine uygulanabilir (Kaveh ve Biabani Hamedani, 2022).

AOA, matematiksel aritmetik operatörlerin özelliklerini ve bu operatörlerin problem çözme süreçlerinde oynadığı rolü kullanır. Aritmetik işlemler, çözüm uzayında çeşitli yönlerde arama yapmaya olanak tanır. Toplama ve Çıkarma, çözüm uzayında yerel ve küresel arama dengesi oluşturur. Çarpma ve Bölme, çözümler arasında çeşitlilik sağlar ve uygunluk fonksiyonuna göre daha iyi çözümlere yakınsamayı destekler (Hu vd., 2022).

Sürekli ve ayrık optimizasyon problemlerine kolayca uygulanabilir. Matematiksel olarak anlaşılır ve uygulanması kolaydır. Exploration ve exploitation süreçleri dengeli bir şekilde yönetilir. Çözüm uzayını kapsamlı bir şekilde tarar ve küresel optimuma ulaşma şansı yüksektir (Agushaka ve Ezugwu, 2021).

Çok karmaşık veya büyük boyutlu problemler için yüksek iterasyon sayısı gerekebilir. Algoritmanın performansı, başlangıç parametrelerinin seçimine bağlı olabilir (X. D. Li vd., 2022).

Makine öğrenmesinde hiperparametre optimizasyonu ve model seçim süreçlerinde; mühendislikte sistem tasarımı ve enerji optimizasyonunda, lojistikte yol planlama ve kontrol algoritmalarında; biyolojide genetik analiz ve biyolojik modellemede, ekonomide risk yönetimi ve portföy optimizasyonunda kullanılır.

3. YÖNTEM

3.1. Veri Kümesi ve Ön İşleme

Kullanılan veri seti, DeepFake ve normal ses verilerinin Spektogram, Chromagram gibi grafik verilerinin sayısal veriye dönüşümlerini içeren bir CSV dosyası formatındadır. Veri setine, ön işleme olarak normalizasyon işlemi uygulanmıştır.

3.2. Derin Öğrenme Modellerinin Tasarımı

Temel yapılandırmada (yani sezgisel optimizasyon uygulanmadan), hem RNN hem de LSTM modelleri aynı genel mimariyi paylaşmıştır. Giriş katmanı, ADD 2022 veri setinden çıkarılan spektrogram tabanlı özellik vektörlerini almıştır. Bunu iki adet tekrarlayan (recurrent) katman izlemiştir: RNN modelinde bunlar standart RNN katmanları iken, LSTM modelinde LSTM katmanlarıdır. Her bir tekrarlayan katman sabit sayıda nöron içermektedir (birinci katmanda 128 birim, ikinci katmanda 64 birim) ve tanh aktivasyon fonksiyonu kullanılmıştır. Tekrarlayan katmanların ardından tam bağlantılı (dense) bir katman, bir dropout katmanı ve son olarak "REAL" ve "FAKE" sınıflarına karşılık gelen iki nöronlu bir Softmax çıkış katmanı yer almıştır.

3.3. Derin Öğrenme Algoritmalarının İşleyişi

3.3.1. RNN

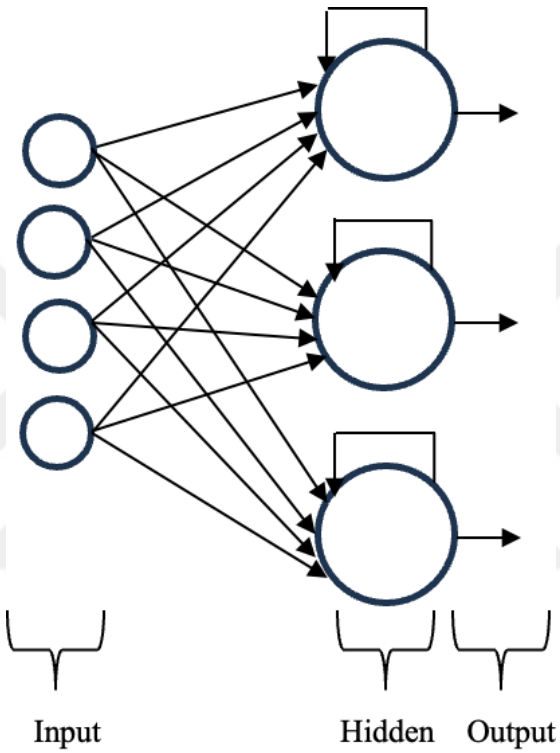
RNN'nin en önemli mekanizması, gizli durum (hidden state) adı verilen bir yapının her zaman adımında güncellenmesidir. Gizli durum, mevcut giriş verisinden (x_t) ve bir önceki gizli durumdan (h_{t-1}) etkilenir. Bu matematiksel olarak şu şekilde ifade edilir:

$$h_t = f\ (W_h \cdot h_{t-1} + W_x \cdot x_t + b)$$

Burada W_h gizli durum ağırlıklarını, W_x giriş ağırlıklarını, b ise bias terimini temsil eder. f, aktivasyon fonksiyonu, genellikle tanh veya ReLU olarak seçilir. Gizli durum, çıktıyı üretmek için kullanılır ve bu işlem

$$y_t = g\ (W_y \cdot h_t + b_y)$$

denklemiyle ifade edilir. Bu yapı, ağın sıralı veri bağımlılıklarını öğrenmesini mümkün kılar (Mienye vd., 2024).



3.1. RNN yapısı

Her iki mimari için de temel hiperparametreler sabit tutulmuş ve manuel olarak seçilmiştir: öğrenme oranı 0.0001, dropout oranı 0.3–0.5, giriş boyutu zaman adımı başına 32–64 özellik, batch boyutu 32 ve eğitim epoch sayısı 20 olarak ayarlanmıştır (Çizelge: 3.2.'de özetlenmiştir). Tüm deneyler boyunca Adam optimizasyon algoritması ve kategorik çapraz entropi kayıp fonksiyonu kullanılmıştır. Bu yapılandırma, Bölüm 3'te sezgisel olarak optimize edilen modellerin karşılaştırıldığı optimize edilmemiş referans noktası olarak kullanılmaktadır.

Projede çalışılan kod, Kaggle platformunda yayımlanmış DeepFake_Voice projesinden alınmış ve sezgisel algoritmalar uygulanabilmesi için modifiye edilmiştir. Orijinal kod içeriğinde derin öğrenme parametreleri sabit olup sezgisel algoritmaların uygulanması ile her iterasyonda dinamik olarak verilmiştir.

3.3.2. LSTM

LSTM'nin matematiksel temelleri şu şekildedir:

Unutma Kapısı:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big) \quad (1)$$

Giriş: h_{t-1} (önceki gizli durum) ve x_t (giriş vektörü).

Çıkış: 0 ile 1 arasında bir değer. 0, bilginin tamamen unutulacağını; 1, bilginin tamamen korunacağını gösterir.

Giriş Kapısı:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = tanh(W_C \cdot [h_t - 1, x_t] + b_C) \quad (3)$$

i_t: Yeni bilginin hücre durumuna ne kadar ekleneceğini belirler, $\tilde{C}_t$: Yeni aday bilgi.

Hücre Durumu Güncellemesi:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

Eski hücre durumu (C_{t-1}) unutma kapısına, yeni bilgi ($\tilde{C}_t$) giriş kapısına göre güncellenir.

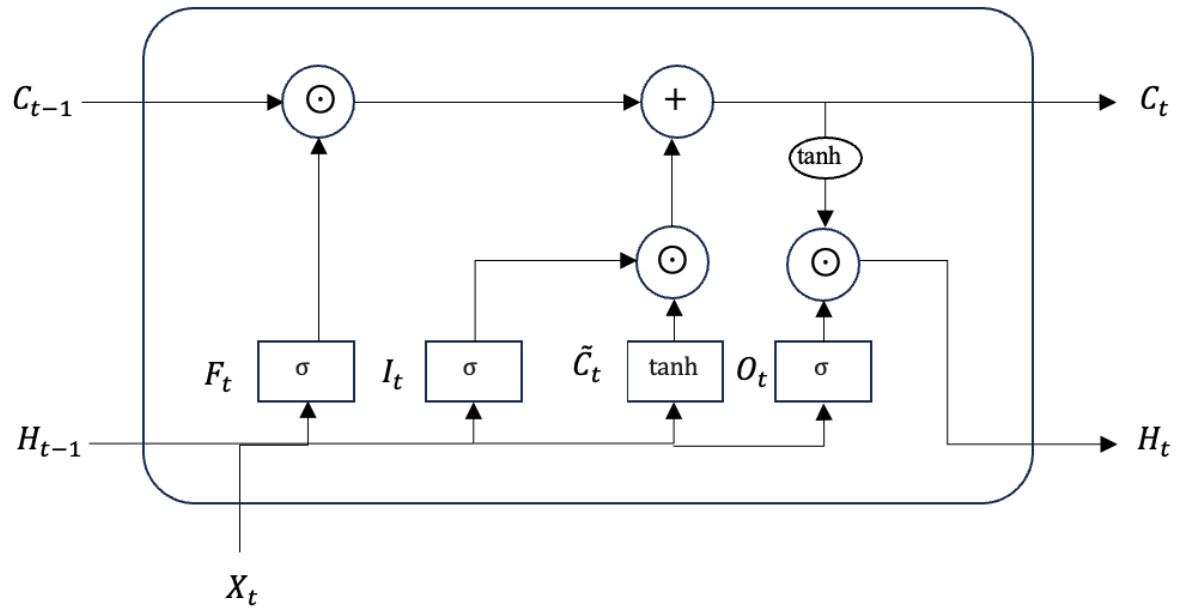
Çıkış Kapısı:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * tanh(C_t) \quad (6)$$

Giriş: h_{t-1} (önceki gizli durum) ve x_t (giriş vektörü).

Çıkış: 0 ile 1 arasında bir değer. 0, bilginin tamamen unutulacağını; 1, bilginin tamamen korunacağını gösterir.



3.2 LSTM yapısı

3.4. Sezgisel Algoritmalar ile Parametre Optimizasyonu

Derin öğrenme modellerinde parametre optimizasyonu yapmak, modelin performansını en üst düzeye çıkarmak için kritik bir adımdır. Parametreler, modelin öğrenme kapasitesini doğrudan etkiler ve doğru ayarlandığında model daha iyi genelleme yapabilir. Aksi takdirde, yanlış ayarlanmış parametreler, modelin overfitting (aşırı öğrenme) veya underfitting (yetersiz öğrenme) gibi sorunlarla karşılaşmasına yol açabilir. Örneğin, öğrenme oranı çok yüksek olduğunda model, en iyi çözümü bulamadan "atlayabilir," çok düşük olduğunda ise eğitim süreci gereksiz yere uzar. Benzer şekilde, modelin katman sayısı ve her bir katmandaki nöron sayısı, modelin karmaşıklığını belirler ve doğru ayarlandığında, modelin hem daha doğru hem de daha hızlı öğrenmesini sağlar.

Optimum parametreler, modelin eğitim sürecini daha verimli hale getirir. Yanlış ayarlanmış parametreler, eğitim süresinin uzamasına ve dolayısıyla daha fazla hesaplama kaynağının harcanmasına yol açabilir. Ancak doğru parametrelerle, eğitim süresi kısalır ve model daha hızlı bir şekilde öğrenir. Bu, özellikle büyük veri setleri ile çalışan derin öğrenme modelleri için önemli bir avantaj sağlar. Ayrıca, doğru hiper-parametrelerle modelin yalnızca eğitim verisi üzerinde değil, test verileri üzerinde de yüksek performans göstermesi sağlanabilir. Bu, modelin genelleme kabiliyetini artırır ve overfitting'i engeller. Overfitting, modelin eğitim verisi üzerinde çok iyi performans sergilerken test verisi üzerinde kötü sonuçlar vermesi durumudur. Parametre optimizasyonu, bu durumu engelleyerek, modelin daha sağlam ve güvenilir sonuçlar üretmesini sağlar.

Derin öğrenme modelleri, çok sayıda parametre içerdiğinden, eğitim sırasında zaman alıcı ve hesaplama açısından pahalı olabilir. Ancak parametre optimizasyonu, bu hesaplama kaynaklarını daha verimli kullanmaya yardımcı olur. Sezgisel algoritmalar gibi optimizasyon teknikleri, çözüm alanını daha hızlı keşfederek gereksiz hesaplamalardan kaçınmayı sağlar. Bu sayede modelin öğrenme süreci hızlanır ve daha verimli bir şekilde çalışır. Parametre optimizasyonunun bir diğer avantajı da farklı veri setlerine uyum sağlama yeteneğidir. Her veri seti, farklı özelliklere ve yapılara sahip olabilir; dolayısıyla her veri seti için aynı parametre ayarları etkili olmayabilir. Optimum parametreler, her veri setine özgü en uygun hiperparametrelerin bulunmasını sağlar ve bu da modelin farklı veri setlerinde yüksek performans göstermesini mümkün kılar.

Sonuç olarak, derin öğrenme modellerinde parametre optimizasyonu yapmak, modelin doğruluğunu artırmak, eğitim süresini kısaltmak, genelleme kabiliyetini iyileştirmek ve overfitting'i engellemek için kritik bir adımdır. Bu süreç, doğru hiperparametre ayarlarını bulmak için önemlidir çünkü bu ayarlar, modelin daha hızlı ve etkili bir şekilde öğrenmesini sağlarken, daha düşük hata oranlarıyla daha güvenilir sonuçlar elde edilmesine yardımcı olur.

Sezgisel algoritmalar, klasik optimizasyon yöntemlerinin kullanılamadığı veya verimli çalışmadığı durumlarda tercih edilen problem çözme stratejileridir. Bu algoritmalar, global optimuma ulaşmak için çok sayıda çözüm arar ve lokal minimumdan kaçınmaya çalışır. Derin öğrenme modellerindeki hiperparametre optimizasyonunda kullanılan sezgisel algoritmalar, modelin performansını iyileştirmede önemli bir rol oynar. Bu algoritmalar, çözüm alanını hızlı bir şekilde keşfederek, modelin doğruluğunu artırmaya ve eğitim süresini kısaltmaya yardımcı olur.

Sezgisel algoritmalar, derin öğrenme modellerinde hiperparametre optimizasyonunu sağlamak için oldukça etkilidir çünkü bu modellerde parametrelerin optimal ayarlarını bulmak genellikle karmaşık ve çok boyutludur. Geleneksel yöntemler, tüm parametre kombinasyonlarını manuel olarak denemek zor olduğu için yetersiz kalabilir. Sezgisel algoritmalar ise, büyük veri setlerine ve karmaşık modellerin parametre uzayına hızla uyum sağlayabilen, lokal minimumlardan kaçınarak daha geniş bir çözüm alanını keşfetmelerini sağlar.

3.5. Sezgisel Algoritmaların İşleyişi

3.5.1. PSO

PSO, ilk olarak, parçacıkların pozisyonları ve hızları rastgele bir şekilde başlatılır. Ardından her bir parçacık, optimizasyon probleminin fitness fonksiyonu aracılığıyla değerlendirilir (Shi ve Eberheart, 1999). Her bir parçacık, kendi kişisel en iyi pozisyonunu (pbest) ve grubun en iyi pozisyonunu (gbest) kaydeder.

$$V_i^{t+1} = \omega V_i^t + c_1 r_1 (pbest_i - V_i^t) + c_2 r_2 (gbest - X_i^t) \quad (7)$$

Sonraki adımda, parçacıklar hızlarını ve pozisyonlarını, kendi pbest ve gbest pozisyonlarına göre güncellerler. Algoritma, belirli bir iterasyon sayısına ulaşana kadar veya istenilen çözüm kalitesine elde edilene kadar devam eder.

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (8)$$

X_i^t: $i.$ parçacığın t iterasyonundaki konumu, V_i^t: $i.$ parçacığın hızı, $pbest_i$: $i.$ parçacığın şimdiye kadarki en iyi bireysel konumu, $gbest$: sürünün şimdiye kadarki en iyi global konumu, ω: atalet ağırlığı (exploration–exploitation dengesi), c_1: bilişsel katsayı (parçacığın kendi deneyimi), c_2: sosyal katsayı (sürünün deneyimi), $r_1, r_2 \sim U(0,1)$: rastgele sayılar.

3.5.2. HHO

Algoritma, şahinlerin başlangıç pozisyonlarını rastgele olarak belirler. Bu pozisyonlar, çözüm uzayındaki olası çözümleri temsil eder. Algoritmanın ilk aşamalarında şahinler, geniş bir çözüm uzayında optimum çözümü arar. Bu süreçte, şahinler birbirlerinden bağımsız olarak geniş alanlarda dolaşır ve tavşanın (global optimumun) yerini belirlemeye çalışır. Avın (optimum çözümün) savunma yeteneği dikkate alınır. Tavşanın ne kadar yorgun veya savunmasız olduğu, şahinlerin saldırı stratejilerini etkiler. Eğer tavşan zayıf hale gelmişse, şahinler bir çevreleme ve hızlı saldırı stratejisi uygular. Tavşan kaçamaz hale geldiğinde, şahinler hızlı bir şekilde çözüme ulaşmaya çalışırlar. Eğer tavşan güçlü ise ve kaçma yeteneğine sahipse, şahinler daha karmaşık ve koordine bir strateji izleyerek tavşanı yavaşlatmaya çalışırlar. Bu aşamada, şahinler farklı saldırı yollarını dener ve bu da algoritmanın farklı arama stratejilerini uygulamasına olanak tanır. Tavşan yorgun düştükçe, şahinler avı daha yakından takip eder ve daha dar bir aralıkta iyileştirme yapar. Bu süreçte, tavşanın kaçış yollarını sınırlayarak optimum çözüm bulunmaya çalışılır. Algoritma, belirlenen bir iterasyon sayısına ulaşıldığında ya da en iyi çözüm yeterince iyi olduğunda sonlanır (Heidari vd., 2019).

HHO Aşamaları:

- Faz Geçişi

HHO'da faz geçişi, avın (tavşanın) kaçma enerjisi E ile belirlenir:

$$E = 2E_0\left(1 - \frac{t}{T}\right) \quad (9)$$

$E_0 \sim U(-1,1)$: başlangıç enerjisi, t: mevcut iterasyon, T: maksimum iterasyon sayısı, $|E| \geq 1 \rightarrow$ Keşif (Exploration), $|E| < 1 \rightarrow$ Sömürü (Exploitation)

- Keşif (Exploration) Fazı

$$X_{t+1} = \begin{cases} X_{rand} - r_1|X_{rand} - 2r_2X_t|, & q \geq 0.5 \\ (X_{rabbit} - X_{mean}) - r_3(LB + r_4(UB - LB), & q < 0.5 \end{cases} \quad (10)$$

X_{rand}: rastgele seçilen şahin, X_{rabbit}: en iyi çözüm (av), X_{mean}: popülasyon ortalaması

$$r_1, r_2, r_3, r_4, q \sim U(0,1) \quad (11)$$

- Sömürü

Bu faz, avın kaçma olasılığı r ve enerji E değerine göre 4 alt stratejiye ayrılır.

- Yumuşak Kuşatma

$$X_{t+1} = \Delta X_t - E|JX_{rabbit} - X_t| \quad (12)$$

$$\Delta X_t = X_{rabbit} - X_t \quad (13)$$

- Sert Kuşatma (Hard Besiege)

$$X_{t+1} = X_{rabbit} - E|\Delta X_i| \quad (14)$$

- Yumuşak Kuşatma + Hızlı Dalış (Levy Flight)

$$Y = X_{rabbit} - E|JX_{rabbit} - X_t| \quad (15)$$

$$Z = Y + S.LF(D) \quad (16)$$

- Sert Kuşatma + Hızlı Dalış

$$Y = X_{rabbit} - E|\Delta X_t| \quad (17)$$

$$Z = Y + S.LF(D) \quad (18)$$

- Levy Flight

$$LF(D) = 0.01 \times \frac{u}{|v|^{\frac{1}{\beta}}}, u \sim \mathcal{N}(0, \sigma^2), v \sim \mathcal{N}(0,1) \quad (19)$$

Genellikle:

$$\beta = 1.5$$

- Atlayış Gücü (Jump Strength)

$$J = 2(1 - r), r \sim U(0,1) \quad (20)$$

- Güncelleme Kuralı

$$X_{t+1} = \begin{cases} Y, f(Y) < f(X_t) \\ Z, f(Z) < f(X_t) \end{cases} \quad (21)$$

- Temel Parametreler

Popülasyon boyutu: N, Maksimum iterasyon: T, Alt/üst sınırlar: LB, UB

3.5.3. Jaya

Problem için bir amaç fonksiyonu belirlenir. Çözüm uzayında rastgele başlangıç çözümleri oluşturulur. Çözüm sayısı (örneğin, popülasyon büyüklüğü) belirlenir. Her çözüm, verilen amaç fonksiyonuna göre değerlendirilir. En iyi çözüm (best) ve en kötü çözüm (worst) popülasyondan seçilir. Her çözüm için, mevcut pozisyonu şu şekilde güncellenir:

$$X_{new} = X_{current} + r_1(X_{best} - |X_{current}|) - r_2(X_{worst} - |X_{current}) \quad (22)$$

Burada:

$X_{current}$: Mevcut çözüm, X_{best}: En iyi çözüm, X_{worst}: En kötü çözüm, r_1, r_2 : 0 ile 1 arasında rastgele üretilen değerler.

Güncellemeler sırasında, çözüm en iyiye yaklaşmaya çalışır ve en kötüden uzaklaşır. Yeni oluşturulan çözümler, mevcut çözümlerden daha iyiyse eski çözümleri değiştirir. En iyi ve en kötü çözümler yeniden belirlenir. Belirlenen bir iterasyon sayısına ulaşıldığında veya çözüm iyileştirmesi artık gözlenmediğinde algoritma durur (Yu vd., 2019).

3.5.4. FPA

Çözüm uzayında rastgele bir başlangıç popülasyonu oluşturulur. Her birey bir "çiçek" olarak kabul edilir. Lévy uçuşları ile modellenir ve geniş çözüm uzayını keşfetmeyi hedefler.

$$X_i^{t+1} = t_i^t + L \cdot (x_i^t - g^*) \quad (23)$$

Burada:

L: Lévy dağılımından türetilmiş adım büyüklüğü, X_i^t: Mevcut bireyin çözümü, g^*: Mevcut en iyi çözüm.

Çiçeklerin yakın çözümler arasında polen değişimi yaptığı süreci simüle eder.

$$X_i^{t+1} = X_i^t + \epsilon\left(x_j^t - x_k^t\right) \quad (24)$$

Burada:

X_j^k ve X_k^t: Rastgele seçilen iki farklı çözüm, ϵ: Rastgele bir ağırlık katsayısı.

Algoritma, küresel ve yerel arama arasında geçiş yapmak için bir olasılık (p) kullanır. Genellikle $p = 0.8$ olarak alınır. Her iterasyonda mevcut çözümler değerlendirilir ve

uygunluk fonksiyonuna göre en iyi çözüm güncellenir. Algoritma, durma kriterine (örneğin, maksimum iterasyon sayısı veya belirli bir hata toleransı) ulaşılana kadar çalışmaya devam eder (Alyasseri vd., 2018).

3.5.5. GA

Rastgele bireylerden oluşan başlangıç popülasyonu oluşturulur. Her bireyin uygunluk değeri hesaplanır. Seçilimde, en iyi çözümleri temsil eden bireyler seçilir. Çaprazlamada, seçilen bireyler birleştirilerek yeni bireyler oluşturulur. Mutasyonda, yeni bireylerin bazı genleri rastgele değiştirilir. Popülasyon Güncellemesi yapılırken, yeni bireyler, eski popülasyonun yerine geçer. Durma kriterine ulaşılana kadar süreç tekrarlanır (Frenzel, 1993).

- Popülasyon Tanımı

$$\mathcal{P}^{(t)} = \left\{X_1^{(t)}, X_2^{(t)}, \dots, X_N^{(t)}\right\} \quad (25)$$

N: popülasyon büyüklüğü

X_i: i. bireyin kromozomu (çözüm vektörü)

i: jenerasyon (iterasyon) sayısı

- Uygunluk (Fitness) Fonksiyonu

$$f: \mathbb{R}^D \to \mathbb{R}, f(X_i) = bireyin\ uygunluk\ değeri$$

Amaç:

Minimizasyon**:** $\min f(x)$

Maksimizasyon**:** $\max f(x)$

- Seçilim (Selection)

Rulet Tekerleği Seçilimi

$$P_i = \frac{f(X_i)}{\sum_{j=1}^{N} f(X_j)} \quad (26)$$

Turnuva Seçilimi

$$X_{best} = arg \min_{X \epsilon \mathcal{T}} f(x) \quad (27)$$

$\mathcal{T}$: rastgele seçilen birey alt kümesi

- Çaprazlama (Crossover)

Tek Noktalı Çaprazlama

$$o_1 = (x_1, \dots, x_k, y_{k+1}, \dots, y_D) \quad (28)$$

$$o_2 = (y_1, \dots, y_k, x_{k+1}, \dots, x_D) \quad (29)$$

$$o_1, o_2 = \begin{cases} Crossover(x,y), r < P_c \\ x, y, \quad\quad\quad\quad r \geq P_c \end{cases}, r \sim U(0,1) \quad (30)$$

k: çaprazlama noktası, P_c: çaprazlama olasılığı

Aritmetik Çaprazlama (Gerçek Değerli GA)

$$o_1 = \alpha x + (1-\alpha)y \quad (31)$$

$$o_2 = (1-\alpha)x + \alpha, \alpha \sim U(0,1) \quad (32)$$

- Mutasyon (Mutation)

  Bit-Flip Mutasyonu

$$x_i = \begin{cases} 1 - x_i, olasılık\ P_m \\ x_i \quad\quad , aksi\ halde \end{cases} \quad (33)$$

  Gaussian Mutation

$$x_i = x_i + N(0, \sigma^2) \quad (34)$$

  P_m: mutasyon olasılığı

- Elitizm (Elitism)

  En iyi birey(ler) doğrudan bir sonraki nesle aktarılır:

$$x_{elite}^{(t+1)} = x_{best}^{(t)} \quad (35)$$

- Popülasyon Güncelleme

$$P^{(t+1)} = Selection + Crossover + Mutation + Elitizm \quad (36)$$

- Durdurma Kriterleri

  Maksimum jenerasyon sayısına ulaşması

  Yakınsama

$$\left(\left| f_{best}^{(t)} - f_{best}^{(t-1)} \right| < \varepsilon \right) \quad (37)$$

- Temel Parametreler

  Popülasyon boyutu N

  Çaprazlama olasılığı $P_c \in [0.6, 0.9]$

  Mutasyon olasılığı $P_m \in [0.001, 0.1]$

3.5.6. ALO

Her karınca, optimizasyon probleminin bir potansiyel çözümünü temsil eder. Karıncalar çözüm uzayında rastgele dolaşır. Karınca aslanları, mevcut en iyi çözümleri temsil eder. Karınca aslanları, karıncaları yakalamak için çukur stratejisini simgeler. Karıncaların hareketi, bir karınca aslanı tarafından oluşturulan huniye doğru çekilme sürecine dayanır. Karıncaların dolaşım alanı zamanla daralır, bu da çözümlerin

optimizasyonda iyileşmesini sağlar. Karıncalar, çözüm uzayında rastgele yürüyüş modeliyle hareket eder. Bu hareket, başlangıçta geniş bir arama alanını temsil eder ve zamanla daralır.

$$X(t) = [cumsum(2r_1 - 1), cumsum(2r_2 - 1), cumsum(2r_d - 1)] \quad (38)$$

$cumsum$: Kümülatif toplam, r_i: Rastgele [0, 1] aralığında değerler, d: Problem boyutu.

Karıncaların çözüm uzayında dolaşma sınırları zamanla daralır.

$$X_i(t) = \frac{(Ub - Lb) \cdot (X_{rand} - Lb)}{10^t} + Lb \quad (39)$$

Ub, Lb: Üst ve alt sınırlar, X_{rand}: Rastgele bir konum, t: İterasyon sayısı.

Karıncalar, karınca aslanlarının etkisi altında bir noktaya doğru hareket eder:

$$X_i(t+1) = \frac{X_i(t) + X_{antlion}}{2} \quad (40)$$

$X_{antlion}$: Karınca aslanının konumu.

Elitizm ile en iyi çözüm sürekli izlenir ve karıncaların bu çözüme çekilmesi sağlanır (Wu vd., 2017).

3.5.7. DE

Rastgele bir başlangıç popülasyonu (NP) oluşturulur. Her birey çözüm uzayında bir konumu temsil eder ve alt/üst sınırlar göz önünde bulundurularak başlatılır. Her birey için uygunluk fonksiyonu değeri hesaplanır. Her birey için diferansiyel mutasyon ve çaprazlama işlemleri uygulanarak deneme bireyler oluşturulur. Deneme bireyler ile mevcut bireyler karşılaştırılır. Daha iyi çözümler bir sonraki nesile aktarılır. Belirli bir durma kriterine (örneğin, maksimum iterasyon sayısı veya uygunluk fonksiyonunun belirli bir değere ulaşması) ulaşılana kadar 3. ve 4. adımlar tekrarlanır (Bilal vd., 2020).

Başlangıçta rastgele oluşturulan NP bireyden oluşur. Her birey bir D-boyutlu vektördür.

Örneğin, $X_i = [x_{i1}, x_{i2}, , x_{iD}]$.

Uygunluk Fonksiyonu (Fitness), her bireyin "ne kadar iyi" bir çözüm olduğunu değerlendirmek için kullanılır. Amaç, bu uygunluk değerini en iyi (maksimum veya minimum) hale getirmektir.

Diferansiyel Mutasyon ile rastgele seçilen üç farklı birey (X_a, X_b, X_c) kullanılarak bir hedef birey (X_i) için mutant birey (V_i) oluşturulur:

$$V_i = X_a + F \cdot (X_b - X_c) \quad (41)$$

Burada:

F: Ölçekleme faktörü, genellikle [0, 2] aralığında bir sabit.

Çaprazlama (Crossover) ise hedef birey (X_i) ile mutant birey (V_i) arasında gen değişimi yapılır.

Bu süreçte yeni bir deneme birey (U_i) oluşturulur:

$$U_{ij} = \begin{cases} V_{ij}, & eğer\ r_j \leq CR\ veya\ j = j_{rand} \\ X_{ij}, & diğer\ durumlarda \end{cases} \quad (42)$$

Burada:

CR: Çaprazlama oranı (genellikle [0.1, 0.9] arasında bir değer), r_j: Rastgele [0, 1] arasında bir sayı, j_{rand}: Rastgele seçilen bir gen pozisyonu (en az bir genin mutant bireyden gelmesini garanti eder).

Seçilim (Selection) deneme birey (U_i) ile hedef birey (X_i) karşılaştırılır. Daha iyi uygunluk değerine sahip olan birey bir sonraki nesile aktarılır:

$$X_i^{t+1} = \begin{cases} U_i, & eğer\ f(U_i) \leq f(X_i) ise \\ X_i, & diğer\ durumlarda \end{cases} (Neri\ ve\ Tirronen, 2010) \quad (43)$$

3.5.8. CEM

Çözüm uzayı için başlangıçta bir olasılık dağılımı ($P(x|\theta)$) tanımlanır. Parametre vektörü θ, bu olasılık dağılımını kontrol eder (örneğin, normal dağılım için ortalama ve standart sapma). Tanımlanan olasılık dağılımından N adet rastgele çözüm ($x_1, x_2, , x_N$) üretilir. Her çözüm bir uygunluk (fitness) fonksiyonu $f(x)$ ile değerlendirilir. Çözümler, uygunluk değerlerine göre sıralanır ve en iyi k çözüm (örneğin, en iyi %10) seçilir. İyi çözümlere dayalı olarak θ parametresi güncellenir. Yeni olasılık dağılımı, gelecekte daha iyi çözümler üretmek için optimize edilir. Belirlenen durma kriterine (örneğin, maksimum iterasyon sayısı veya yeterli yakınsama) kadar süreç tekrarlanır.

CEM'in amacı, aşağıdaki problemi çözmektir:

$$x^* = arg\ max\ f(x)\ , x \quad (44)$$

$f(x)$, çözümün uygunluk fonksiyonu

Güncellenmiş parametreler, çapraz entropiyi minimize edecek şekilde seçilir:

$$\theta^* = arg\ min\ D_{KL}\big(P(x|\theta^*)\big|P(x|\theta)\big) \quad (45)$$

D_{KL}, Kullback-Leibler uzaklığıdır ve iki dağılım arasındaki farklılığı ölçer (Jamin ve Humeau-Heurtier, 2020).

3.5.9. AOA

Rastgele bir başlangıç popülasyonu (P) oluşturulur. Her birey, çözüm uzayında bir potansiyel çözümü temsil eder. Yerel ve küresel arama süreçleri, aritmetik operatörler kullanılarak modellenir. Toplama, çıkarma, çarpma ve bölme işlemleri, mevcut çözümleri iyileştirmek için kullanılır. Arama süreci, aritmetik yoğunluk (Arithmetic Intensity - AI) adlı bir parametre ile kontrol edilir. AI, iterasyon ilerledikçe yerel aramaya (exploitation) daha fazla ağırlık verir, başlangıçta ise küresel aramayı (exploration) destekler.

$$AI = \frac{Iter}{MaxIter}, \quad (46)$$

$Iter$, mevcut iterasyon numarası, $MaxIter$, maksimum iterasyon sayısı

Çözüm iyileştirme, aritmetik operatörlerin ağırlıklı kombinasyonu ile yapılır.

$$NewSolution = CurrentSolution + r_1 \cdot (BestSolution - CurrentSolution), \quad (47)$$

Burada r_1, rastgele bir sayı (0 ile 1 arasında) ve $BestSolution$, o ana kadarki en iyi çözümü temsil eder. Her bireyin uygunluk değeri, problem tanımına göre hesaplanır. Daha iyi çözümler popülasyona dahil edilir. Belirli bir durma kriterine (örneğin, maksimum iterasyon sayısı veya uygunluk fonksiyonunun istenen değere ulaşması) kadar 3. ila 5. adımlar tekrarlanır.

$$NewSolution = CurrentSolution \pm r_1 \cdot (r_2 \cdot BestSolution - CurrentSolution), \quad (48)$$

r_1 ve r_2, rastgele sayı dağılımları, $\pm$, rastgele seçilen toplama veya çıkarma işlemi

$$NewSolution = CurrentSolution \times r3 \; veya \; CurrentSolution/r3, \quad (49)$$

r_3, bir rastgele sayı.

Arama Ağırlığı (AI), küresel arama ve yerel arama arasında geçiş yapmak için kullanılır (Dhal vd., 2023).

3.6. Deney Ortamı

Sezgisel optimizasyonun derin öğrenme eğitimine entegrasyonu, çok aşamalı bir iş akışı üzerinden gerçekleştirilmiştir (Şekil: 3.1.):

- Başlatma (Initialization):

Sezgisel algoritma, aday hiperparametre kümelerinden oluşan bir popülasyonu başlatır.

- Değerlendirme (Evaluation):

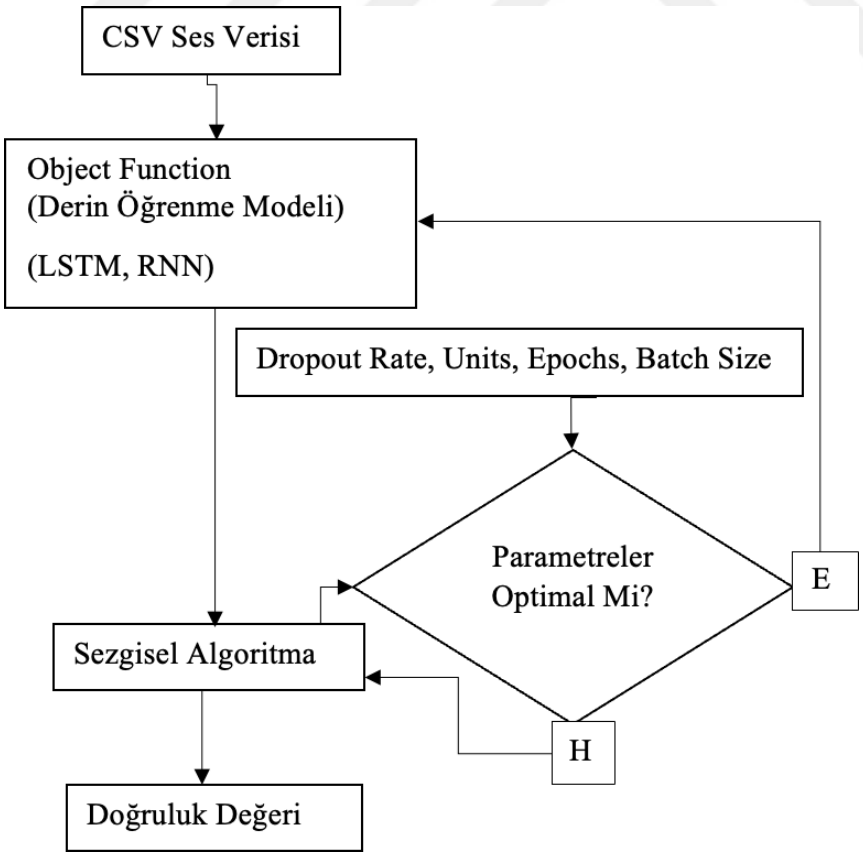
Her bir aday yapılandırma, RNN veya LSTM modelini sabit bir epoch sayısı (20) boyunca eğitmek için kullanılır. Ardından doğrulama (validation) doğruluğu hesaplanır.

- Seçim ve Güncelleme (Selection and Update):

Sezgisel algoritma, her adayın uygunluk (fitness) değerini değerlendirir ve popülasyonu kendi özel evrimsel veya sosyal kurallarına göre (örneğin mutasyon, çaprazlama, çekim veya iş birliği) günceller.

- Sonlandırma (Termination):

Süreç, maksimum yineleme sayısına (100) ulaşılana veya yakınsama (convergence) sağlanana kadar tekrarlanır. Bu optimizasyon süreci, her yinelemede model parametrelerini dinamik olarak ayarlayarak daha iyi yakınsama davranışı ve azalmış aşırı uyum (overfitting) sağlar. Çerçeve, optimizasyon rutinleri için pyMetaheuristic kütüphanesi kullanılarak MacBook Pro (Apple M1 Pro, 16 GB RAM) üzerinde PyCharm IDE ortamında uygulanmıştır.



Şekil: 3.3. Akış diyagram

4. BULGULAR VE TARTIŞMA

Bu bölümde, Deepfake ses tespiti için RNN ve LSTM mimarileri ile sezgisel ve meta-sezgisel algoritmaların entegre edilmesiyle elde edilen deneysel sonuçlar sunulmaktadır. Optimizasyon süreci; öğrenme oranı, dropout oranı, giriş boyutu ve batch boyutu olmak üzere dört temel hiperparametre üzerinde, 9 farklı sezgisel algoritma kullanılarak gerçekleştirilmiştir. Kullanılan algoritmalar, popülerlikleri ve farklı sezgisel algoritma kategorilerinden olmaları sebebi ile tercih edilmişlerdir. Her bir deney, 100 iterasyon boyunca 50 popülasyon üyesi ile yürütülmüş ve model performansı temel olarak doğrulama doğruluğu (validation accuracy) üzerinden değerlendirilmiştir.

Sabit ve manuel olarak seçilmiş hiperparametrelerle eğitilen temel Deepfake tespit modeli ortalama %77,1 doğruluk elde etmiştir. Sezgisel optimizasyonun uygulanmasının ardından, tüm modellerde belirgin performans artışları gözlemlenmiş ve bazı yapılandırmalar %99'un üzerinde doğruluğa ulaşmıştır.

Görece düşük olan temel doğruluk oranı, optimize edilmemiş mimari ve eğitim parametrelerine bağlanabilir. Özellikle 0.0001 olarak sabitlenen öğrenme oranı ve nispeten yüksek dropout değerleri (0.3–0.5), yakınsama verimliliğini sınırlamış ve alt-optimal yerel minimumlara takılma olasılığını artırmıştır. Buna ek olarak, küçük giriş boyutları (32–64), spektrogram özellikleri üzerinde aşırı boyut indirgemesine neden olarak, gerçek ve Deepfake konuşmayı ayırt etmek için gerekli olan ince ayrıntılı zaman–frekans yapılarının yakalanmasını zorlaştırmıştır. ADD 2022 veri setindeki sınıf dengesizliği (8.275 sahteye karşılık 3.260 gerçek örnek) de özellikle azınlık sınıfında genelleme yeteneğinin azalmasına katkıda bulunmuştur.

DeepFake ve gerçek ses verilerinin, sinyal verileri kullanılarak, Librosa kütüphanesi ile öznitelik çıkarımı yapılmıştır. Çıkarılan özniteliklerin, kendi içlerindeki ortalama değerleri hesaba katılmıştır. Çizelge 4.1.'de öznitelikler ve sayıları belirtilmiştir.

Çizelge: 4.1. Çıkarılan öznitelikler

Öznitelikler	Sayısı
Chroma STFT, RMS, Spectral Centroid, Spectral Bandwidth, Rolloff, Zero Crossing Rate	1
MFCC	20

Buna karşılık, sezgisel optimizasyon prosedürleri temel hiperparametreleri dinamik olarak ayarlayarak modellerin daha etkili biçimde yakınsamasını sağlamış ve temel yapılandırmada mevcut olan sınırlamaların aşılmasına olanak tanımıştır. Ortaya çıkan performans sonuçları Çizelge: 4.2.'de özetlenmektedir. Çizelge: 4.1., optimizasyon süreci boyunca incelenen hiperparametre aralıklarını özetlemektedir.

Çizelge: 4.2. Hiperparametre Değerleri

Hiperparametre	Arama Aralığı
Öğrenme oranı (Learning rate)	0.0001 – 0.1
Dropout oranı (Dropout rate)	0.1 – 0.5
Girdi boyutu (Input size)	32 – 256
Batch boyutu (Batch size)	16 – 128

Sezgisel algoritmalar tarafından belirlenen en iyi hiperparametre kümeleri, doğruluk oranı, kesinlik oranı ve AUC değerleri Çizelge 4.3.'de gösterilmektedir. Sonuçlar, sezgisel optimizasyon sonrasında doğruluk oranında belirgin bir iyileşme olduğunu ortaya koymaktadır. Tüm yapılandırmalar genelinde:

- RNN modelleri, aynı optimizasyon koşulları altında LSTM modellerinden tutarlı biçimde daha iyi performans göstermiştir. Bu durum, yinelemeli durum aktarımının (recurrent state propagation), konuşma verilerindeki zamansal bağımlılıkların öğrenilmesi için daha kararlı bir temel sağladığını göstermektedir.
- Test edilen tüm algoritmalar arasında, Fark Evrimi (DE) LSTM modeli için en yüksek doğruluk oranını %97,16, Parçacık Sürüsü Optimizasyonu (PSO) ise RNN modeli için en yüksek doğruluk oranını %99,03 ile elde etmiştir.
- Optimizasyon, yinelemeler boyunca eğitim doğruluğundaki varyansı önemli ölçüde azaltmış, bu da yakınsama kararlılığının arttığını göstermektedir.
- Jaya ve Genetik Algoritma (GA) gibi bazı algoritmalar, 15 yinelemeden daha kısa

sürede neredeyse doygun (saturated) doğruluk oranlarına ulaşarak CEM veya ALO gibi stokastik yöntemlere göre daha hızlı yakınsama göstermiştir.

Genel olarak, sezgisel algoritmaların entegrasyonu, temel modele kıyasla %18–22 oranında doğruluk artışı sağlamıştır. Bu sonuçlar, sezgisel tabanlı hiperparametre ayarlamasının, hesaplama maliyetini önemli ölçüde artırmadan modelin genelleme kabiliyetini büyük ölçüde iyileştirdiğini doğrulamaktadır.

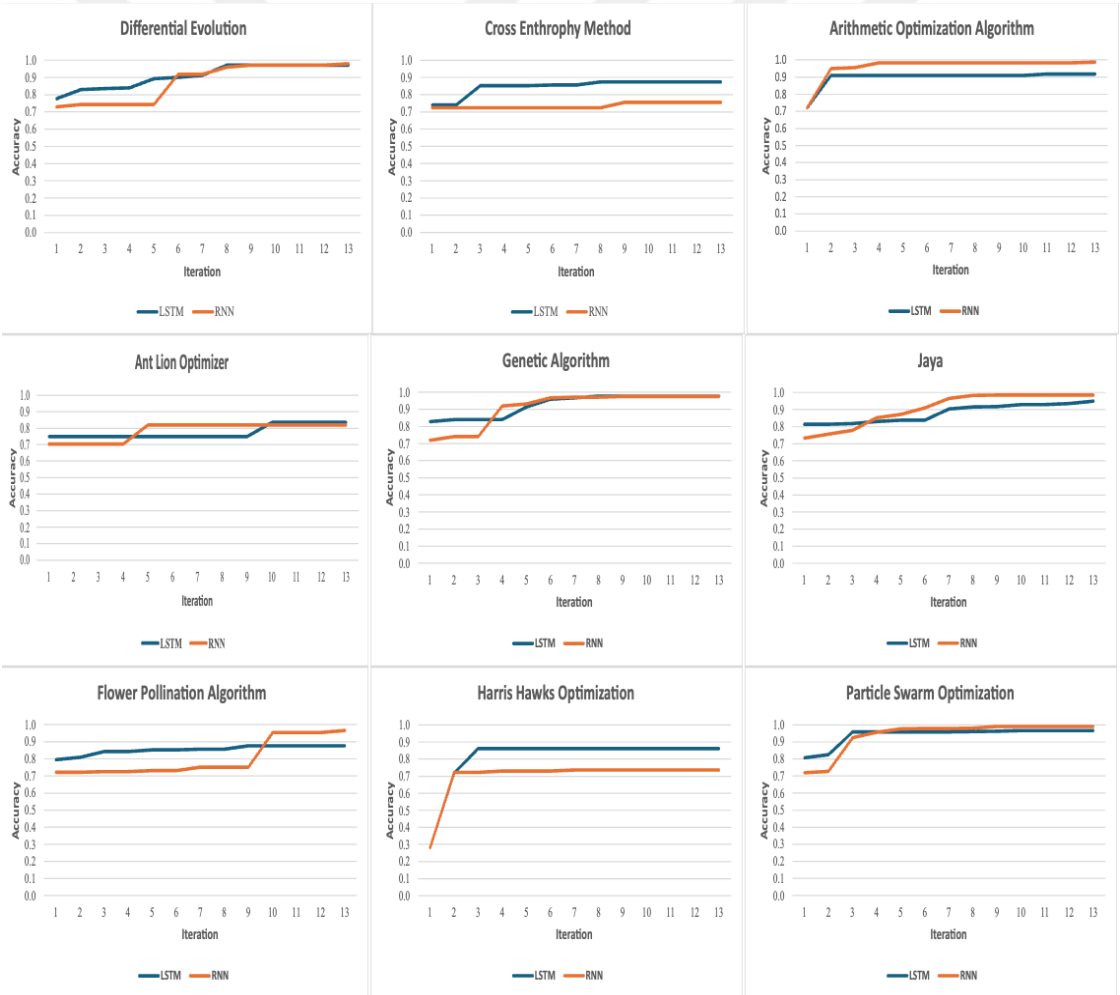
Çizelge 4.3. Sonuç tablosu

Sezgisel Algoritma	**Model**	**Öğrenme Oranı**	**Dropout Oranı**	**Girdi Boyutu**	**Batch Boyutu**	**Test Doğruluk Oranı**	**Kesinlik Oranı**	**AUC**
DE	LSTM	0.0031	0.1178	182	113	0.9716	0.973	0.986
	RNN	0.0001	0.3481	256	49	0,9798	0.982	0.991
CEM	LSTM	0.0007	0.2290	161	79	0.8737	0.875	0.910
	RNN	0.0186	0.2919	148	87	0.7544	0.751	0.801
AOA	LSTM	0.0001	0.1	256	16	0.9176	0.911	0.944
	RNN	0.0001	0.1	256	16	0.9873	0.988	0.995
ALO	LSTM	0.0161	0.1	42	128	0.8356	0.822	0.882
	RNN	0.0089	0.1	39	124	0.8202	0.809	0.869
GA	LSTM	0.0011	0.3812	213	16	0.9766	0.974	0.988
	RNN	0.0001	0.1962	255	93	0.9750	0.971	0.987
Jaya	LSTM	0.0031	0.1349	70	117	0.9499	0.945	0.968
	RNN	0.0001	0.1	256	16	0,9858	0.986	0.994
FPA	LSTM	0.0098	0.3938	110	96	0.8765	0.866	0.912
	RNN	0.0001	0.5	256	64	0.9651	0.958	0.982
HHO	LSTM	0.0081	0.2220	116	128	0.8623	0.853	0.899
	RNN	0.1	0.5	122	64	0.7356	0.718	0.780
PSO	LSTM	0.0009	0.5	256	128	0.9669	0.961	0.983
	RNN	0.0001	0.1	256	16	0.9903	0.991	0.997
Baseline	**LSTM**	**0.0001**	**0.3, 0.5**	**32, 64**	**32**	**0.7715**	**0.764**	**0.845**
	RNN	**0.0001**	**0.3, 0.5**	**32, 64**	**32**	**0.7503**	**0.734**	**0.771**

Her bir sezgisel algoritmanın yakınsama davranışı, iterasyon–doğruluk grafikleri Şekil: 4.1. kullanılarak görselleştirilmiştir. Her bir grafik, modelin en uygun çözüme ilerleyişini göstermekte olup, daha düşük uygunluk (fitness) değerleri daha yüksek doğruluğa karşılık gelmektedir.

Görsel analizlerin tamamında, DE, GA ve PSO gibi algoritmaların düzgün ve monoton bir iyileşme sergilediği gözlemlenmiş; bu durum, keşif–sömürü (exploration–exploitation) dengesinin etkili bir şekilde sağlandığını göstermektedir. Buna karşılık, ALO ve HHO gibi yöntemler daha yüksek salınımlar göstermiş olup, bu durum başlangıç popülasyon çeşitliliğine olan duyarlılığı yansıtmaktadır.

Elde edilen sonuçlar, derin öğrenme ile sezgisel optimizasyonun birleştirilmesinin Deepfake ses tespitinde kayda değer performans artışları sağladığını ortaya koymaktadır. Sezgisel algoritmalar, manuel ayarlama veya grid search ile belirlenmesi pratikte mümkün olmayan en uygun öğrenme yapılandırmalarını otomatik olarak tespit edebilmiştir.



4.1 Performans Grafikleri

Tüm görsel analizler genelinde, DE, GA ve PSO gibi algoritmalar düzgün ve monoton bir gelişim sergileyerek etkili bir keşif–sömürü (exploration–exploitation) dengesi kurduklarını göstermiştir. Buna karşılık, ALO ve HHO yöntemleri daha yüksek salınım (oscillation) göstermiştir. Bu da başlangıç popülasyonundaki çeşitliliğe olan duyarlılığı yansıtmaktadır.

Sonuçlar, derin öğrenme ile sezgisel optimizasyonun birleştirilmesinin Deepfake ses tespitinde önemli performans artışları sağladığını göstermektedir. Sezgisel algoritmalar, manuel ayarlama veya grid search yöntemleriyle belirlenmesi neredeyse imkânsız olan en uygun öğrenme yapılandırmalarını otomatik olarak keşfedebilmiştir.

Özellikle:

- PSO, işbirlikçi sürü dinamikleri sayesinde RNN modellerinde güçlü bir uyum yeteneği ve hızlı yakınsama göstermiş; öğrenme oranı ve dropout oranlarını verimli biçimde ayarlamıştır.
- DE, özellikle LSTM modellerinde etkili olmuş; mutasyon–yeniden kombinasyon (recombination) mekanizması sayesinde keşif ve hassas hiperparametre ayarı (örneğin öğrenme oranı) arasında denge kurmuştur.
- GA ve Jaya, doğruluk açısından biraz daha düşük sonuçlar verse de daha az yinelemede dengeye ulaşmış, bu da büyük ölçekli uygulamalara uygun hesaplama verimliliğini göstermiştir.

Farklı algoritmalarda gözlemlenen tutarlı iyileşmeler, sezgisel temelli optimizasyonun model mimarileri arasında genelleştirilebildiğini ve hem tespit doğruluğunu hem de eğitim verimliliğini artırdığını doğrulamaktadır. Önceki çalışmalar, CNN veya BiLSTM ağlarıyla %85 ila %94 doğruluk elde ederken (Almutairi ve Elgibreen, 2022; Dixit vd., 2023), bu çalışmadaki optimize edilmiş RNN–PSO yapılandırması ADD 2022 veri setinde %99,0 doğruluğa ulaşmıştır (Almutairi ve Elgibreen, 2022; Dixit vd., 2023). Bu oran, benzer deneysel koşullar altında rapor edilen en yüksek Deepfake ses tespit sonuçlarından biridir. Özetle, hiperparametre optimizasyonu için sezgisel algoritmaların uygulanması, RNN ve LSTM modellerinin performansını önemli ölçüde artırmıştır.

Temel gözlemler şunlardır:

- Optimizasyon, temel doğruluk oranını %77,1'den %99,0'a kadar yükseltmiştir.
- Fark Evrimi (DE) ve Parçacık Sürüsü Optimizasyonu (PSO) en kararlı ve en doğru sonuçları vermiştir.
- Çoğu algoritmada yakınsama genellikle ilk 10–15 yineleme içinde sağlanmıştır.
- Entegrasyon çerçevesi, hesaplama açısından uygulanabilir olup birden fazla model mimarisine uyarlanabilir niteliktedir.

5. SONUÇ VE ÖNERİLER

Gelecek çalışmalarda, önerilen optimizasyon çerçevesinin farklı yönlerden geliştirilmesi önerilmektedir. Bu kapsamda; gerçek dünya ses koşullarındaki dayanıklılığın değerlendirilebilmesi amacıyla çok dilli ve gürültü içeren veri kümeleri üzerinde alanlar arası (cross-domain) doğrulama çalışmalarının gerçekleştirilmesi önem arz etmektedir. Ayrıca, keşif ve sömürü (exploration–exploitation) yeteneklerinin tamamlayıcı güçlü yönlerinden yararlanmak amacıyla DE-PSO veya GA-Jaya gibi hibrit optimizasyon yaklaşımlarının uygulanması önerilmektedir. Bununla birlikte, daha üst düzey konuşma temsillerinin elde edilebilmesi için Conformer ve Wav2Vec tabanlı modeller gibi gelişmiş derin öğrenme mimarileri ile entegrasyon sağlanması uygun olacaktır. Son olarak, optimize edilmiş modellerin daha önce karşılaşılmamış sentetik örüntülere ve adversarial manipülasyonlara karşı tepkilerinin incelendiği model yorumlanabilirliği ve adversarial dayanıklılık analizlerinin gerçekleştirilmesi, sistem güvenilirliği açısından önemli katkılar sunacaktır.

Sonuç olarak, bu çalışmada RNN ve LSTM mimarileri kullanılarak Deepfake ses tespiti problemine yönelik sezgisel (heuristic) tabanlı bir hiperparametre optimizasyon çerçevesi geliştirilmiştir. Dokuz farklı sezgisel algoritma aracılığıyla parametre uzaylarının sistematik biçimde araştırılması sonucunda, hesaplama verimliliği korunarak literatürdeki güncel yöntemlerle karşılaştırılabilir düzeyde yüksek doğruluk performanslarına ulaşılmıştır. Elde edilen bulgular, optimizasyon süreçlerinin, sentetik konuşma tespitinde derin öğrenme modellerinin potansiyelini en üst düzeye çıkarmada belirleyici bir rol oynadığını ortaya koymaktadır. Ses Deepfake uygulamalarının ötesinde, önerilen yöntemin diğer medya doğrulama ve sinyal adli analiz (signal forensics) alanlarında da genellenebilir bir performans artırma yaklaşımı sunduğu değerlendirilmektedir.

KAYNAKLAR DİZİNİ

Abdel-Basset, M., ve Shawky, L. A. (2019, Aralık 1). Flower pollination algorithm: a comprehensive review. *Artificial Intelligence Review*, C. 52, ss. 2533-2557. Springer Netherlands. https://doi.org/10.1007/s10462-018-9624-4

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., ve Gandomi, A. H. (2021). The Arithmetic Optimization Algorithm. *Computer Methods in Applied Mechanics and Engineering*, *376*. https://doi.org/10.1016/j.cma.2020.113609

Abualigah, L., Shehab, M., Alshinwan, M., Mirjalili, S., ve Elaziz, M. A. (2021). Ant Lion Optimizer: A Comprehensive Survey of Its Variants and Applications. *Archives of Computational Methods in Engineering*, *28*(3), 1397-1416. https://doi.org/10.1007/s11831-020-09420-6

Agrawal, P., Abutarboush, H. F., Ganesh, T., ve Mohamed, A. W. (2021). Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019). *IEEE Access*, *9*, 26766-26791. https://doi.org/10.1109/ACCESS.2021.3056407

Agushaka, J. O., ve Ezugwu, A. E. (2021). Advanced arithmetic optimization algorithm for solving mechanical engineering design problems. *PLoS ONE*, *16*(8 August). https://doi.org/10.1371/journal.pone.0255703

Alabool, H. M., Alarabiat, D., Abualigah, L., ve Heidari, A. A. (2021, Ağustos 1). Harris hawks optimization: a comprehensive review of recent variants and applications. *Neural Computing and Applications*, C. 33, ss. 8939-8980. Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/s00521-021-05720-5

Almutairi, Z., ve Elgibreen, H. (2022, Mayıs 1). A Review of Modern Audio Deepfake Detection Methods: Challenges and Future Directions. *Algorithms*, C. 15. MDPI. https://doi.org/10.3390/a15050155

Alyasseri, Z. A. A., Khader, A. T., Al-Betar, M. A., Awadallah, M. A., ve Yang, X. S. (2018). Variants of the flower pollination algorithm: A review. Içinde *Studies in Computational Intelligence* (C. 744, ss. 91-118). Springer Verlag. https://doi.org/10.1007/978-3-319-67669-2_5

Amezaga, N., ve Hajek, J. (2022). Availability of Voice Deepfake Technology and its Impact for Good and Evil. *SIGITE 2022 - Proceedings of the 23rd Annual Conference on Information Technology Education*, 23-28. Association for Computing Machinery, Inc. https://doi.org/10.1145/3537674.3554742

KAYNAKLAR DİZİNİ

Baghel, M., Agrawal, S., ve Silakari, S. (2012). Survey of Metaheuristic Algorithms for Combinatorial Optimization. Içinde *International Journal of Computer Applications* (C. 58).

Beheshti, Z., Mariyam, S., ve Shamsuddin, H. (2013). A Review of Population-based Meta-Heuristic Algorithm. Içinde *Int. J. Advance. Soft Comput. Appl* (C. 5). Geliş tarihi gönderen www.i-csrs.org

Bilal, Pant, M., Zaheer, H., Garcia-Hernandez, L., ve Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence*, *90*. https://doi.org/10.1016/j.engappai.2020.103479

da Silva, L. S. A., Lúcio, Y. L. S., Coelho, L. dos S., Mariani, V. C., ve Rao, R. V. (2023). A comprehensive review on Jaya optimization algorithm. *Artificial Intelligence Review*, *56*(5), 4329-4361. https://doi.org/10.1007/s10462-022-10234-0

De Boer, P.-T., Kroese, D. P., ve Rubinstein, R. Y. (2005). A Tutorial on the Cross-Entropy Method. Içinde *Annals of Operations Research* (C. 134). Geliş tarihi gönderen http://www.cemethod.org/

Desale, S., Rasool, A., Andhale, S., ve Rane, P. (2015). Heuristic and Meta-Heuristic Algorithms and Their Relevance to the Real World: A Survey. *INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS*, *2*, 296-304. Geliş tarihi gönderen http://www.ijcert.org

Dhal, K. G., Sasmal, B., Das, A., Ray, S., ve Rai, R. (2023). A Comprehensive Survey on Arithmetic Optimization Algorithm. *Archives of Computational Methods in Engineering*, *30*(5), 3379-3404. https://doi.org/10.1007/s11831-023-09902-3

Dinkar, S. K., ve Deep, K. (2017). Opposition based Laplacian Ant Lion Optimizer. *Journal of Computational Science*, *23*, 71-90. https://doi.org/10.1016/j.jocs.2017.10.007

Dixit, A., Kaur, N., ve Kingra, S. (2023). Review of audio deepfake detection techniques: Issues and prospects. *Expert Systems*, *40*(8). https://doi.org/10.1111/exsy.13322

Elgamal, Z. M., Yasin, N. B. M., Tubishat, M., Alswaitti, M., ve Mirjalili, S. (2020). An improved harris hawks optimization algorithm with simulated annealing for feature selection in the medical field. *IEEE Access*, *8*, 186638-186652. https://doi.org/10.1109/ACCESS.2020.3029728

KAYNAKLAR DİZİNİ

Ezugwu, A. E., Adeleke, O. J., Akinyelu, A. A., ve Viriri, S. (2020). A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems. *Neural Computing and Applications*, *32*(10), 6207-6251. https://doi.org/10.1007/s00521-019-04132-w

Forrest, S. (1996). Genetic Algorithms. Içinde *ACM Computing Surveys* (C. 28).

Frank, J., ve Schönherr, L. (2021). *WaveFake: A Data Set to Facilitate Audio Deepfake Detection*. Geliş tarihi gönderen http://arxiv.org/abs/2111.02813

Frenzel, james F. (1993). *Genetic algorithms*.

Gandomi, A. H., Yang, X. S., Talatahari, S., ve Alavi, A. H. (2013). Metaheuristic Algorithms in Modeling and Optimization. Içinde *Metaheuristic Applications in Structures and Infrastructures* (ss. 1-24). Elsevier Inc. https://doi.org/10.1016/B978-0-12-398364-0.00001-2

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., … Chen, T. (2015). *Recent Advances in Convolutional Neural Networks*. Geliş tarihi gönderen http://arxiv.org/abs/1512.07108

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., ve Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849-872. https://doi.org/10.1016/j.future.2019.02.028

Holland, J. H. (1992). *Genetic Algorithms*. *267*(1), 66-73. https://doi.org/10.2307/24939139

Hopper, E., ve Turton, B. (1999). *Theory and Methodology An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem*. Geliş tarihi gönderen www.elsevier.com/locate/dsw

Hu, G., Zhong, J., Du, B., ve Wei, G. (2022). An enhanced hybrid arithmetic optimization algorithm for engineering applications. *Computer Methods in Applied Mechanics and Engineering*, *394*. https://doi.org/10.1016/j.cma.2022.114901

Hussien, A. G., Abualigah, L., Zitar, R. A., Hashim, F. A., Amin, M., Saber, A., … Gandomi, A. H. (2022, Haziran 1). Recent Advances in Harris Hawks Optimization: A Comparative Study and Applications. *Electronics (Switzerland)*, C. 11. MDPI. https://doi.org/10.3390/electronics11121919

KAYNAKLAR DİZİNİ

Jain, N. K., Nangia, U., ve Jain, J. (2018, Ağustos 1). A Review of Particle Swarm Optimization. *Journal of The Institution of Engineers (India): Series B*, C. 99, ss. 407-411. Springer. https://doi.org/10.1007/s40031-018-0323-y

Jamin, A., ve Humeau-Heurtier, A. (2020, Ocak 1). (Multiscale) cross-entropy methods: A review. *Entropy*, C. 22, s. 45. MDPI AG. https://doi.org/10.3390/e22010045

Kamboj, V. K., Nandi, A., Bhadoria, A., ve Sehgal, S. (2020). An intensify Harris Hawks optimizer for numerical and engineering optimization problems. *Applied Soft Computing Journal*, *89*. https://doi.org/10.1016/j.asoc.2019.106018

Kaur, M., ve Mohta, A. (2019). *A Review of Deep Learning with Recurrent Neural Network*. IEEE.

Kaveh, A., ve Biabani Hamedani, K. (2022). Improved arithmetic optimization algorithm and its application to discrete structural optimization. *Structures*, *35*, 748-764. https://doi.org/10.1016/j.istruc.2021.11.012

Kenny, V., Nathal, M., Saldana ChE, S., Yue, D., ve You, F. (2014). *Heuristic algorithms From optimization*. Geliş tarihi gönderen https://optimization.mccormick.northwestern.edu/index.php?title=Heuristic_algorithmsveprintable=yes

Khanjani, Z., Watson, G., ve Janeja, V. P. (2023). *How Deep Are the Fakes? Focusing on Audio Deepfake: A Survey*.

Khochare, J., Joshi, C., Yenarkar, B., Suratkar, S., ve Kazi, F. (2022). A Deep Learning Framework for Audio Deepfake Detection. *Arabian Journal for Science and Engineering*, *47*(3), 3447-3458. https://doi.org/10.1007/s13369-021-06297-w

Kokash, N. (t.y.). *An introduction to heuristic algorithms*.

Kroese, D. P., Rubinstein, R. Y., Cohen, I., Porotsky, S., ve Taimre, T. (2013). Cross-Entropy Method. Içinde *Encyclopedia of Operations Research and Management Science* (ss. 326-333). Springer US. https://doi.org/10.1007/978-1-4419-1153-7_131

Kumar, A., ve Bawa, S. (2020). A comparative review of meta-heuristic approaches to optimize the SLA violation costs for dynamic execution of cloud services. *Soft Computing*, *24*(6), 3909-3922. https://doi.org/10.1007/s00500-019-04155-4

Li, C. Y., Li, J., Chen, H. L., Jin, M., ve Ren, H. (2021). Enhanced Harris hawks optimization with multi-strategy for global optimization tasks. *Expert Systems with Applications*, *185*. https://doi.org/10.1016/j.eswa.2021.115499

KAYNAKLAR DİZİNİ

Li, X. D., Wang, J. S., Hao, W. K., Zhang, M., ve Wang, M. (2022). Chaotic arithmetic optimization algorithm. *Applied Intelligence*, *52*(14), 16718-16757. https://doi.org/10.1007/s10489-021-03037-3

Lipton, Z. C., Berkowitz, J., ve Elkan, C. (2015). *A Critical Review of Recurrent Neural Networks for Sequence Learning*. Geliş tarihi gönderen http://arxiv.org/abs/1506.00019

M, S. D., ve Koumoutsakos, P. (t.y.). *A Parameter Study for Differential Evolution*.

Mannor, S., Peleg, B., ve Rubinstein, R. (2005). The cross entropy method for classification. *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*, 561-568. https://doi.org/10.1145/1102351.1102422

Mao, S., ve Sejdic, E. (2023). A Review of Recurrent Neural Network-Based Methods in Computational Physiology. *IEEE Transactions on Neural Networks and Learning Systems*, *34*(10), 6983-7003. https://doi.org/10.1109/TNNLS.2022.3145365

Martín-Doñas, J. M., ve Álvarez, A. (2022). THE VICOMTECH AUDIO DEEPFAKE DETECTION SYSTEM BASED ON WAV2VEC2 FOR THE 2022 ADD CHALLENGE. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2022-May*, 7937-7941. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ICASSP43922.2022.9747768

Mathew, T. V. (t.y.). *Genetic Algorithm*.

McUba, M., Singh, A., Ikuesan, R. A., ve Venter, H. (2023). The effect of deep learning methods on deepfake audio detection for digital investigation. *Procedia Computer Science*, *219*, 211-219. Elsevier B.V. https://doi.org/10.1016/j.procs.2023.01.283

Mienye, I. D., Swart, T. G., ve Obaido, G. (2024). Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications. *Information*, *15*(9), 517. https://doi.org/10.3390/info15090517

Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software*, *83*, 80-98. https://doi.org/10.1016/j.advengsoft.2015.01.010

Mirjalili, S., Jangir, P., ve Saremi, S. (2017). Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems. *Applied Intelligence*, *46*(1), 79-95. https://doi.org/10.1007/s10489-016-0825-8

Mitchell, M. (1995). Genetic Algorithms: An Overview 1. Içinde *Complexity* (C. 1).

KAYNAKLAR DİZİNİ

Mouassa, S., Bouktir, T., ve Salhi, A. (2017). Ant lion optimizer for solving optimal reactive power dispatch problem in power systems. *Engineering Science and Technology, an International Journal*, *20*(3), 885-895. https://doi.org/10.1016/j.jestch.2017.03.006

Neri, F., ve Tirronen, V. (2010). Recent advances in differential evolution: A survey and experimental analysis. *Artificial Intelligence Review*, *33*(1-2), 61-106. https://doi.org/10.1007/s10462-009-9137-2

Opara, K. R., ve Arabas, J. (2019). Differential Evolution: A survey of theoretical analyses. *Swarm and Evolutionary Computation*, *44*, 546-558. https://doi.org/10.1016/j.swevo.2018.06.010

Pandey, H. M. (2016). *Jaya a Novel Optimization Algorithm: What, How and Why?* IEEE.

Preuss, N., Chen, W.-H., Xiao, T., ve Hu, G. (2024). *Heuristic algorithms Author: Zemin Mi (zm287), Boyu Yang (by274) (ChemE 6800 Fall 2024)*. Geliş tarihi gönderen https://optimization.cbe.cornell.edu/index.php?title=Heuristic_algorithms

Qu, C., He, W., Peng, X., ve Peng, X. (2020). Harris Hawks optimization with information exchange. *Applied Mathematical Modelling*, *84*, 52-75. https://doi.org/10.1016/j.apm.2020.03.024

Rajpurohit, J., Kumar Sharma, T., ve Abraham, A. (2017). Glossary of Metaheuristic Algorithms. Içinde *International Journal of Computer Information Systems and Industrial Management Applications* (C. 9). Geliş tarihi gönderen www.mirlabs.net/ijcisim/index.html

Rao, R. V., Rai, D. P., Ramkumar, J., ve Balic, J. (2016). A new multi-objective Jaya algorithm for optimization of modern machining processes. *Advances in Production Engineering And Management*, *11*(4), 271-286. https://doi.org/10.14743/apem2016.4.226

Rawat, W., ve Wang, Z. (2017, Eylül 1). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, C. 29, ss. 2352-2449. MIT Press Journals. https://doi.org/10.1162/NECO_a_00990

Rubinstein, R. (1999a). The Cross-Entropy Method for Combinatorial and Continuous Optimization. Içinde *Methodology and Computing in Applied Probability* (C. 1). Kluwer Academic Publishers.

KAYNAKLAR DİZİNİ

Rubinstein, R. (1999b). The Cross-Entropy Method for Combinatorial and Continuous Optimization. Içinde *Methodology and Computing in Applied Probability* (C. 1). Kluwer Academic Publishers.

Sastry, K., Goldberg, D. E., ve Kendall, G. (2014). Genetic algorithms. Içinde *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, Second Edition* (ss. 93-118). Springer US. https://doi.org/10.1007/978-1-4614-6940-7_4

Shaaban, O. A., Yildirim, R., ve Alguttar, A. A. (2023). Audio Deepfake Approaches. *IEEE Access*, *11*, 132652-132682. https://doi.org/10.1109/ACCESS.2023.3333866

Shehab, M., Mashal, I., Momani, Z., Shambour, M. K. Y., AL-Badareen, A., Al-Dabet, S., … Abualigah, L. (2022, Kasım 1). Harris Hawks Optimization Algorithm: Variants and Applications. *Archives of Computational Methods in Engineering*, C. 29, ss. 5579-5603. Springer Science and Business Media B.V. https://doi.org/10.1007/s11831-022-09780-1

Shi, Y., ve Eberheart, R. C. (1999). *Empirical Study of Particle Swarm Optimization*.

Song, M.-P., ve Gu, G.-C. (2004). *RESEARCH ON PARTICLE SWARM OPTIMIZATION: A REVIEW*.

Staudemeyer, R. C., ve Morris, E. R. (2019). *Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks*. Geliş tarihi gönderen http://arxiv.org/abs/1909.09586

Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P., ve Vrahatis, M. N. (2004). *Parallel Differential Evolution*.

Thawkar, S., Sharma, S., Khanna, M., ve Singh, L. kumar. (2021). Breast cancer prediction using a hybrid method based on Butterfly Optimization Algorithm and Ant Lion Optimizer. *Computers in Biology and Medicine*, *139*. https://doi.org/10.1016/j.compbiomed.2021.104968

Van Houdt, G., Mosquera, C., ve Nápoles, G. (2020). A review on the long short-term memory model. *Artificial Intelligence Review*, *53*(8), 5929-5955. https://doi.org/10.1007/s10462-020-09838-1

Venkata Rao, R., ve Saroj, A. (2017). A self-adaptive multi-population based Jaya algorithm for engineering optimization. *Swarm and Evolutionary Computation*, *37*, 1-26. https://doi.org/10.1016/j.swevo.2017.04.008

KAYNAKLAR DİZİNİ

Wang, D., Tan, D., ve Liu, L. (2018). Particle swarm optimization algorithm: an overview. *Soft Computing*, *22*(2), 387-408. https://doi.org/10.1007/s00500-016-2474-6

Warid, W., Hizam, H., Mariun, N., ve Abdul-Wahab, N. I. (2016). Optimal power flow using the Jaya algorithm. *Energies*, *9*(9). https://doi.org/10.3390/en9090678

Wong, W., ve Ing Ming, C. (2019). *A Review on Metaheuristic Algorithms: Recent Trends, Benchmarking and Applications*. IEEE.

Wu, Z., Yu, D., ve Kang, X. (2017). Parameter identification of photovoltaic cell model based on improved ant lion optimizer. *Energy Conversion and Management*, *151*, 107-115. https://doi.org/10.1016/j.enconman.2017.08.088

Yang, X.-S. (2013). *Flower Pollination Algorithm for Global Optimization*. https://doi.org/10.1007/978-3-642-32894-7_27

Yang, X.-S. (2014). *Flower Pollination Algorithm: An Introduction*.

Yang, X.-S., Karamanoglu, M., ve He, X. S. (2014). *Flower Pollination Algorithm: A Novel Approach for Multiobjective Optimization*. https://doi.org/10.1080/0305215X.2013.832237

Yi, J., Tao, J., Fu, R., Yan, X., Wang, C., Wang, T., … Li, H. (2023). *ADD 2023: the Second Audio Deepfake Detection Challenge*. Geliş tarihi gönderen http://arxiv.org/abs/2305.13774

Yi, J., Wang, C., Tao, J., Zhang, X., Zhang, C. Y., ve Zhao, Y. (2023). *Audio Deepfake Detection: A Survey*. Geliş tarihi gönderen http://arxiv.org/abs/2308.14970

Yu, K., Qu, B., Yue, C., Ge, S., Chen, X., ve Liang, J. (2019). A performance-guided JAYA algorithm for parameters identification of photovoltaic cell and module. *Applied Energy*, *237*, 241-257. https://doi.org/10.1016/j.apenergy.2019.01.008

Zhang, Y., Chi, A., ve Mirjalili, S. (2021). Enhanced Jaya algorithm: A simple but efficient optimization method for constrained engineering design problems. *Knowledge-Based Systems*, *233*. https://doi.org/10.1016/j.knosys.2021.107555

Zitar, R. A., Al-Betar, M. A., Awadallah, M. A., Doush, I. A., ve Assaleh, K. (2022, Mart 1). An Intensive and Comprehensive Overview of JAYA Algorithm, its Versions and Applications. *Archives of Computational Methods in Engineering*, C. 29, ss. 763-792. Springer Science and Business Media B.V. https://doi.org/10.1007/s11831-021-09585-8