T.C.

SELÇUK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY SİNİR AĞLARININ YAPAY ALG ALGORİTMASI İLE EĞİTİMİ

Bahaeddin TÜRKOĞLU

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Haziran-2018
KONYA

TEZ KABUL VE ONAYI

Bahaeddin TÜRKOĞLU tarafından hazırlanan "Yapay Sinir Ağlarının Yapay Alg Algoritması ile Eğitimi" adlı tez çalışması 27/06/2018 tarihinde aşağıdaki jüri tarafından oy birliği / ~~oy çokluğu~~ ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç.Dr. İsmail BABAOĞLU	
Danışman Dr.Öğr.Üyesi Ersin KAYA	
Üye Dr.Öğr.Üyesi Burak YILMAZ	

Yukarıdaki sonucu onaylarım.

Prof. Dr. Mustafa YILMAZ
FBE Müdürü

Bu tez çalışması Selçuk Üniversitesi Bilimsel Araştırma Projeleri Koordinatörlüğü tarafından 18201057 numaralı proje ile desteklenmiştir

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Bahaeddin Türkoğlu

Tarih: 27.06.2018

ÖZET

YÜKSEK LİSANS TEZİ

YAPAY SİNİR AĞLARININ YAPAY ALG ALGORİTMASI İLE EĞİTİMİ

Bahaeddin TÜRKOĞLU

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr.Öğr.Üyesi Ersin KAYA

2018, 52 Sayfa

Jüri
Doç.Dr. İsmail BABAOĞLU
Dr.Öğr.Üyesi Ersin KAYA
Dr.Öğr.Üyesi Burak YILMAZ

Yapay Sinir Ağları günümüzde yaygın olarak kullanılan bir problem çözme stratejisidir. Geçtiğimiz yıllar içerisinde sınıflandırma, örüntü tanıma, görüntü işleme gibi bir çok alanda başarıyla kullanılmıştır. Yapay Sinir Ağlarının en önemli ve en zor aşaması eğitim sürecidir. Eğitimdeki temel zorluk optimum ağ parametrelerinin bulunması sürecidir. Literatürde Yapay Sinir Ağlarının eğitimi için bir çok algoritma kullanılmıştır. Bu tez çalışmasında Yapay Sinir Ağlarının eğitimde ilk kez Yapay Alg Algoritması önerilmiştir. Yapay Alg Algoritması, hareketli mikro alglerin karakteristik özelliklerinden ve yaşam davranışlarından esinlenen metasezgisel bir optimizasyon algoritmasıdır. Bu algoritmanın geniş bir yelpazedeki optimizasyon problemlerini başarıyla çözebildiği kanıtlanmıştır. Önerilen yöntemin başarısının doğrulanması için UCI veri tabanından alınan farklı zorluk seviyelerine sahip on sınıflandırma problemi çözülmüştür. Yapay Alg Algoritması ile eğitilen Yapay Sinir Ağları; Parçacık Sürü Optimizasyonu, Balina Optimizasyon Algoritması, Güve Optimizasyon Algoritması, Çoklu Evren Algoritması, Yarasa Optimizasyon Algoritması, Guguk Kuşu Optimizasyon Algoritması ve Gri Kurt Optimizasyonu ile eğitilen Yapay Sinir Ağlarıyla sınıflandırma problemleri üzerinde kıyaslanmıştır. Algoritmaların sonuçları ortalama sınıflandırma başarısı, en iyi sınıflandırma başarısı, standart sapma değeri, lokal minimumdan kaçınma eğilimi, yakınsama eğrisi, istatistiksel testler ve eğitim süresi üzerinden değerlendirilmiştir. Çalışma sonuçları, Yapay Alg Algoritmasının Yapay Sinir Ağları eğitiminde güvenilir bir yaklaşım olduğunu göstermektedir.

Anahtar Kelimeler: Optimizasyon, Sinir ağlarının eğitimi, Yapay Alg Algoritması

ABSTRACT

MS THESIS

ARTIFICIAL ALGAE ALGORİTHM FOR TRAINING NEURAL NETWORKS

Bahaeddin TÜRKOĞLU

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF SELÇUK UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER ENGINEERING

Advisor: Asst.Prof.Dr. Ersin KAYA

2018, 52 Pages

Jury
Assoc.Prof.Dr. İsmail BABAOĞLU
Asst.Prof.Dr. Ersin KAYA
Asst.Prof.Dr. Burak YILMAZ

Artificial Neural Networks are a commonly used problem solving strategy. Over the years, many areas such as classification, pattern recognition, image processing have been successfully used. The most important and difficult phase of Artificial Neural Networks is the training process. The main difficulty in training is the process of finding optimal network parameters. Many algorithms have been used for the training of artificial neural networks in the literature. In this thesis Artificial Algae Algorithm for training Artificial Neural Network for the first time. Artificial Algae Algorithm is a nature-inspired metaheuristic optimization algorithm. Ten classification dataset with different difficulty levels from UCI database is utilized to benchmark the performance of the proposed method. For verification, the results compared to seven swarm based Algorithm. These Algorithms are Particle Swarm Optimization, Whale Optimization Algorithm, Mont Flame Optimization, Multi Verse Optimization, Bat Optimization Algorithm, Cuckoo Search Optimization and Grey Wolf Optimization. The results of the algorithms were evaluated on average and best classification success, standard deviation, local minimum avoidance tendency, convergence curve, statistical tests and duration of training. The results of the study show that Artificial Algae Algorithm is a reliable approach for training Artificial Neural Networks.

Keywords: Optimization, Artificial Algae Algorithm, Training neural network

ÖNSÖZ

Bu tezin yürütülmesi kapsamında destek ve yardımlarını esirgemeyen danışman hocam ERSİN KAYA'ya katkılarından dolayı teşekkür ederim.

Selçuk Üniversitesi Bilimsel Araştırma Projeleri Koordinatörlüğü'ne 18201057 numaralı proje desteğinden dolayı teşekkürlerimi sunarım.

Bahaeddin Türkoğlu
KONYA-2018

İÇİNDEKİLER

ÖZET ..... **iv**

ABSTRACT ..... **v**

ÖNSÖZ ..... **vi**

İÇİNDEKİLER ..... **vii**

SİMGELER VE KISALTMALAR ..... **viii**

1. GİRİŞ **1**

2. KAYNAK ARAŞTIRMASI **5**

2.1. Evrimsel Tabanlı Algoritmalar 5
2.2. Sürü Tabanlı Algoritmalar 6

3. MATERYAL VE YÖNTEM **9**

3.1. Yapay Alg Algoritması 9
3.1.1. Helisel Hareket 10
3.1.2. Evrimsel Süreç 12
3.1.3. Adaptasyon Süreci 12
3.1.4. Yapay Alg Algoritmasının Sözde Kodu ve Akış Şeması 13
3.2. İleri Beslemeli Yapay Sinir Ağları 16
3.3. Yapay Sinir Ağlarının Yapay Alg Algoritması ile Eğitimi 18
3.3.1. Yapay Alg Algoritması için Temsil Stratejisi 19
3.3.2. Uygunluk Fonksiyonunun Belirlenmesi 20
3.3.3. Önerilen Model 20

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA **24**

4.1. Çalışma Ortamının Hazırlanması 24
4.2. Deneysel Sonuçlar 27

5. SONUÇLAR VE ÖNERİLER **45**

5.1 Sonuçlar 45
5.2 Öneriler 46

KAYNAKLAR **47**

ÖZGEÇMİŞ **52**

SİMGELER VE KISALTMALAR

Simgeler

Ap	: AAA'da adaptasyon parametresi
D	: Problem boyutu
e	: AAA'da enerji kaybı parametresi
f	: Amaç fonksiyonu
G	: AAA'da alg kolonilerinin büyüklükleri
N	: Popülasyondaki alg koloni sayısı
Δ	: AAA'da kesme kuvveti katsayısı

Kısaltmalar

AAA	:Yapay alg algoritması (Artificial algae algorithm)
ABC	: Yapay arı kolonisi algoritması (Artificial bee colony algorithm)
ACO	: Karınca koloni algoritması (Ant colony optimization algorithm)
ANN	: Yapay sinir ağları (Artificial neural network)
BAT	: Yarasa optimizasyon algoritması (Bat optimization algorithm)
BP	: Geri yayılım algoritması (Back propagation algorithm)
CS	: Guguk kuşu arama algoritması (Cuckoo search algorithm)
DE	: Diferansiyel evrim algoritması (Differential evolution algorithm)
GA	: Genetik algoritma (Genetic algorithm)
GWO	: Gri kurt optimizasyonu (Gray wolf optimizer)
HC	: Tepe tırmanma algoritması (Hill climbimg algorithm)
MFO	: Güve optimizasyon algoritması (Mont flame optimization algorithm)
MLP	: Çok katmanlı algılayıcı (Multi layer perceptron)
MSE	: Ortalama karesel hata (Mean squared error)
MVO	: Çoklu evren optimizasyon algoritması (Multi verse optimization algorithm)
PSO	: Parçacık sürü optimizasyonu (Particle swarm optimization)
SA	: Benzetimli tavlama algoritması (Simulated annealing algorithm)
Std.	: Standart sapma
WOA	: Balina optimizasyon algoritması (Whale optimization algorithm)

1. GİRİŞ

Makine öğrenmesi ve yapay zeka literatüründeki en popüler çalışma konularından birisi Yapay Sinir Ağlarıdır (ANN). ANN, biyolojik sinir sisteminin çalışma prensibi model alınarak tasarlanan bir problem çözme stratejisidir. En genel anlamda bir ANN, bir çok yapay işlem elemanının (nöron) birbirlerine değişik etki seviyeleri ile bağlanması sonucu oluşan bir bilgi işlem mekanizmasıdır. ANN'lerin temeli, 1940'larda insan beyninin matematiksel olarak modelleme çalışmaları ile başlamıştır. 1980'lerde yapılan çalışmalar ile hız kazanarak disipline bir şekil almıştır (Yegnanarayana, 2009).

Geçtiğimiz yirmi yıl içerisinde ANN çok geniş bir yelpazede araştırılmış ve sınıflandırma, regresyon, tahmin problemleri, örüntü tanıma, robotik ve sinyal işleme gibi bir çok alana başarıyla uygulanmıştır (Efe ve Kaynak, 1999; Linggard ve ark., 2012; Rezaeianzadeh ve ark., 2014; Schmidhuber, 2015). ANN'nin özellikle mühendislik alanında geniş çapta kullanılmasının en önemli nedeni, çözümü zor problemler için etkin bir alternatif olmasıdır.

Literatürde bir çok ANN modeli önerilmiştir. Evrişimsel Sinir Ağları (Convulational neural network, CNN), İleri Beslemeli Sinir Ağları (Feedforward network, FNN), Kohonen Ağları (Kohonen self organizing network, SOM), Radyan Temelli Fonksiyon Ağları (Radial basis function network, RBFN) ve Tekrarlayan Sinir Ağları (Recurrent neural network, RNN) literatürdeki popüler sinir ağı modelleridir (Kohonen, 1990; Park ve Sandberg, 1993; Dorffner, 1996; Reed ve Marks, 1999; Krizhevsky ve ark., 2012). Bunların arasında en yaygın kullanılan model ileri beslemeli ANN'lerdir. İleri beslemeli ANN'lerde işlem elemanları olan nöronlar, katmanlar şeklinde dizayn edilmiştir. Bir katmandaki nöronların çıkışı bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Bilgi tek yönde çıkışa doğru aktarılır.

Farklı sinir ağı modelleri olsa da öğrenme süreci ortaktır. Literatürde denetimli (danışmanlı) öğrenme, denetimsiz (danışmansız) öğrenme ve takviyeli (pekiştirmeli) öğrenme olmak üzere üç yaygın öğrenme türü bulunmaktadır.

Denetimli öğrenmede sisteme giriş verileriyle birlikte, bu giriş verilerine karşılık gelen çıkış verileri de sunulur. Sistem, giriş ve çıkış arasındaki bağlantıları öğrenmeye çalışır. Sınıflandırma ve regresyon problemlerinde genellikle bu metot kullanılır (Reed ve Marks, 1999; Caruana ve Niculescu-Mizil, 2006).

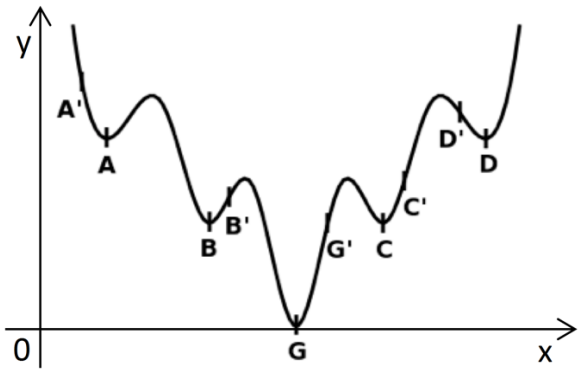
Denetimsiz öğrenmede ise ağa sadece giriş verileri sunulur. Giriş verilerine karşılık çıkış etiketleri yoktur. Bu verilerin saklı olduğu örüntüyü modellemek için

etiketlenmemiş giriş verileri kullanılır. Kümeleme problemlerinde sıklıkla bu yöntem kullanılır (Hinton ve Sejnowski, 1999; Wang, 2001).

Takviyeli öğrenmede de, denetimsiz öğrenme gibi sisteme herhangi bir çıkış verisi sunulmaz. Ancak sistemin ürettiği çıktıların iyi veya kötü olduğu harici bir kaynak tarafından değerlendirilerek sisteme sunulur. Sistem, deneme yanılma yöntemiyle öğrenmeye çalışır. Robotik çalışmalarında bu yöntem daha fazla tercih edilmektedir. (Sutton ve Barto, 1998).

Denetimli öğrenmede, ağın kendisine sunulan giriş verileri için doğru çıkış değerleri üretecek hale gelmesi bir diğer ifadeyle eğitilmesi gerekmektedir. Eğitim, bir ağın tecrübe edinme işlemi, sisteme sunulan verilerden öğrenme sürecidir. Bu süreç aslında, probleme uygun ağ parametrelerinin bulunması işlemi olduğundan bir optimizasyon problemidir. Eğitim süreci, sistemin beklenen çıktısı ile sistemin ürettiği çıktı arasındaki farkın minimize edilme prensibine dayanır. ANN'de öğrenmeyi sağlayan sisteme eğitici denir. Bir eğitici, ağa sunulan girdiler kümesi için çıkışa en yüksek başarıyı verecek şekilde parametreleri ayarlamakla sorumludur. Eğitici, sistemin başarısını geliştirmek için her örnek setinde, diğer ifadeyle her iterasyonda (çevrim) ağın ağırlık parametrelerini yeni örnekleri de kapsayacak şekilde değiştirir. ANN'lerin eğitim süreci ağın başarısını doğrudan etkilediği için sistemin en önemli kısmıdır.

Literatürde ileri beslemeli ANN'nin eğitimi için deterministik yöntemler önerilmiştir. Deterministik yöntemler, rassallık içermezler ve aynı başlangıç parametreleriyle çalıştırıldığında aynı sonucu üretirler. Deterministik yöntemler; hesaplamalı yöntemler ve gradyan tabanlı yöntemler olarak iki kısma ayrılırlar. Hesaplamalı yöntemler gradyan fonksiyonu hesaplamalarına gereksinim duymazlar. Ancak oldukça yavaş ve etkisiz yöntemlerdir. Gradyan tabanlı ise yöntemler amaç fonksiyonunun türevinden faydalanırlar. Bu yöntemler amaç fonksiyonunun yerel optimumlara sahip olduğu problemlerde global optimum çözümü bulmayı garanti etmez. Geri Yayılım Algoritması ve varyantları, gradyan tabanlı yöntemlerin standart örnekleridir (Kim ve Jung, 2015; Wang ve ark., 2015; Hertz, 2018). Gradyan tabanlı algoritmaların en büyük avantajı, basitliği ve hızıdır. Bu algoritmalar genellikle bir çözümle başlar ve çözümü bir optimuma doğru yönlendirmeye çalışır. Yakınsama çok hızlıdır. Ancak elde edilen çözümüm kalitesi büyük ölçüde başlangıç parametrelerine bağlıdır. Ayrıca yerel optimumlara takılma olasılığı yüksektir. Yerel optimum, global optimum aranırken global olmayan bir çözümün global gibi varsayılmasıdır. Şekil 1.1'de dört yerel, bir global minimuma sahip bir fonksiyon gösterilmektedir.



Şekil 1.1. Dört yerel (A, B, C, D) bir global (G) minimuma sahip bir fonksiyon

Şekildeki fonksiyonun global minimum olan sıfır noktası G ile gösterilmektedir. Gradyan tabanlı algoritmalar, global minimumu bulmak için farklı başlangıç parametreleriyle yeniden çalıştırılması gerekir. Örneğin şekildeki fonksiyon için gradyan tabanlı algoritmalar, sırasıyla A', B', C', D' noktalarından başlatılırsa A, B, C, D noktalarını global minimum olarak bulacaktır. Ancak G' noktasından başlatıldığında gerçek global minimum olan G noktasını bulur. Genel olarak gradyan tabanlı yöntemlerin avantajları hız ve basitlik, dezavantajları ise yerel optimumlara takılma eğilimi, başlangıç parametrelerine yüksek bağımlılık, erken ve yavaş yakınsamadır (Karaboga ve ark., 2007; Rakitianskaia ve Engelbrecht, 2012; Wang ve ark., 2014; Mirjalili, 2015; Aljarah ve ark., 2018).

Optimumu aranan fonksiyonların, tek modlu ve çok modlu olmak üzere iki farklı türü bulunmaktadır. Tek modlu fonksiyonlar sadece bir yerel minimuma sahiptir. Çok modlu fonksiyonlar ise birden fazla yerel minimuma sahiptir ve tek-modlu fonksiyonlara göre çözülmeleri daha zordur. Deterministik yöntemlerle yüksek boyutlu, çok modlu ve türevlenemeyen problemleri çözmek oldukça zordur. Hatta bazı problemler için imkansızdır (Ebrahimi ve Khamehchi, 2016).

Literatürde gradyan tabanlı yaklaşımlara alternatif olarak, çok katmanlı ANN'leri eğitmek için sezgisel algoritmalar önerilmiştir. Sezgisel algoritmalar eğitim sürecine rasgele çözüm veya çözümlerden başlar ve iterasyonlar boyunca çözümü hatayı azaltacak şekilde geliştirirler. Metasezgisel algoritmalar, hem başlangıç parametrelerinin belirlenmesinde hem de yerel aramadan global aramaya geçmek için prensip olarak rassallığı içermektedir. Bu nedenle global optimizasyon için daha uygundur (Yang, 2010). Bu tür yöntemlerin avantajı yerel optimumlardan kaçınmadaki yüksek başarılarıdır. Fakat genellikle deterministik yaklaşımlardan çok daha yavaştır (Wang ve

ark., 2014). Gradyan tabanlı algoritmalarının aksine, metasezgisel yöntemler, yerel minimumların önlenmesinde yüksek verimlilik gösterdiği için literatürde daha fazla ilgi görmektedir. (Črepinšek ve ark., 2013; Xu, 2013).

Tezin devamı aşağıdaki gibi organize edilmiştir:

İkinci bölümde ANN'lerin eğitiminde kullanılan çeşitli evrimsel ve sürü tabanlı optimizasyon algoritmalarının literatürü özetlemiştir. Bu bölümde ayrıca önerilen yöntemdeki optimizasyon algoritmaları ile ilgili genel bilgilendirme yapılmıştır.

Üçüncü bölümün birinci kısımda alglerin yaşam davranışlarından esinlenen AAA'nın tanıtımı, aşamaları, matematiksel formülleri izah edilmiştir. Bu bölümün ikinci kısmında, ileri beslemeli ANN'nin çalışma şekli hakkında genel bilgilendirmeler matematiksel formülleriyle yapılmıştır. Bu bölümün son kısmında AAA'nın, ANN eğitiminde nasıl modellenip kullanılacağı detaylandırılmıştır.

Dördüncü bölümün birinci kısmında yapılacak olan testler için veri setlerinin ön hazırlık aşamaları izah edilerek programın kodlanacağı ve çalışacağı platform hakkında açıklamalar yapılmıştır. Bu bölümün ikinci kısmında ise ANN eğitimi için önerilen AAA'nın farklı zorluk seviyelerine sahip veri setleri üzerindeki performans analizi yapılarak yedi farklı sürü zekası algoritmasıyla kıyaslanmıştır.

Beşinci bölümde ise bu çalışmada elde edilen sonuçlar değerlendirilmiş ve gelecekte yapılabilecek çalışmalara öneriler sunulmuştur.

2. KAYNAK ARAŞTIRMASI

ANN'nin eğitim süreci makine öğrenmesindeki en zor problemlerden birisidir (Aljarah ve ark., 2018). Eğitim süreci uygun parametre değerlerinin bulunması işlemi olması dolayısıyla bir optimizasyon problemidir. Bu problemi çözebilmek için literatürde bir çok stokastik optimizasyon algoritması önerilmiştir.

Stokastik tabanlı algoritmaların tek çözüm tabanlı algoritmalar ve çok çözüm (popülasyon) tabanlı algoritmalar olarak iki kategorisi mevcuttur.

Tek çözüm tabanlı algoritmalarda eğitim rastgele oluşturulmuş ağırlık parametrelerine sahip tek bir sinir ağı ile başlar. Bir sonlanma kriteri algoritmayı durduruncaya kadar ANN parametreleri iterasyonlar boyunca, hatayı azaltarak başarıyı artıracak şekilde güncellenmeye devam eder. Sonlanma kriteri genelde belirli bir iterasyon sayısı veya belirli bir hata miktarıdır. Tek çözüm tabanlı algoritmalara Benzetilmiş Tavlama (SA) ve Tepe Tırmanma (HC) algoritmaları örnek olarak gösterilebilir. (Van Laarhoven ve Aarts; Mitchell ve ark., 1994).

Çok çözüm tabanlı algoritmalar ise bir dizi çözüm ile başlar. Sonlanma kriteri sağlanana kadar devam eder. Stokastik tabanlı algoritmaların evrimsel tabanlı algoritmalar ve sürü tabanlı algoritmalar olarak iki temel kategorisi bulunur.

2.1. Evrimsel Tabanlı Algoritmalar

Genetik algoritma (GA) evrimsel algoritmaların klasik bir örneğidir ve ANN eğitiminde en çok kullanılan yöntemlerden birisidir. (Holland, 1992). GA, doğada gözlemlenen seleksiyon sürecinden ilham alan bir arama ve optimizasyon tekniğidir. Karmaşık çok boyutlu problemlerin optimizasyonunda, çözüm uzayındaki global en iyi çözümü bulmaya çalışır (Holland ve Goldberg, 1989; Mitchell, 1998). GA, Holland tarafından 1992'de önerilmiştir Seiffert 2001'de yaptığı çalışmada ANN'yi GA ile eğitmiş, problemin boyutu arttığında ve daha karmaşık olduğunda, GA'nın BP algoritmasından daha iyi olduğunu savunmuştur (Seiffert, 2001). Benzer bir çalışmayı, Gupta ve Randall gerçekleştirmiş, GA'nın kaotik zaman serisi problemleri için BP algoritması ile karşılaştırıldığında etkinlik, kullanım kolaylığı ve verimlilik bakımından üstün olduğunu göstermişlerdir. (Gupta ve Sexton, 1999). İleri beslemeli ANN eğitiminde GA'nın kullanılması ile ilgili benzer bir çok çalışma literatürde mevcuttur (Whitley ve ark., 1990; Sexton ve Gupta, 2000; Ding ve ark., 2011).

Diferansiyel Gelişim (DE) algoritması evrimsel tabanlı bir algoritmadır. Bu algoritma 1996 yılında Storn ve Price tarafından önerilmiş, popülasyon tabanlı, çaprazlama, mutasyon ve seçim operatörlerini içeren bir stokastik optimizasyon algoritmasıdır (Storn ve Price, 1997). 2008'de yapılan çalışmada ileri beslemeli ANN eğitimi için DE algoritması önerilmiş, sonuçları literatürdeki diğer gradyan ve stokastik yöntemlerle kıyaslanarak analizi yapılmıştır. (Wdaa ve Sttar, 2008). Bu konuda yapılan başka bir çalışma ANN eğitiminde DE'nin gradyan tabanlı algoritmalara kıyasla öğrenme oranı veya çözüm kalitesi açısından herhangi bir belirgin avantaj sağlamaz gibi görünürken, BP ile ulaşılan optimum çözümün doğrulanmasında ve doğrusal olmayan (non-lineer) transfer fonksiyonlarının geliştirilmesinde kullanılabileceğini gösterilmiştir (Ilonen ve ark., 2003). Yapılan bir başka çalışmada ANN'nin eğitimi için DE algoritması önerilmiş, kontrol parametrelerinin adaptif olarak seçimi algoritmaya dahil edilmiştir. Parity-p probleminin sınıflandırılması için ANN, önerilen algoritma kullanılarak eğitilmiştir. Bu çalışmada önerilen algoritma kullanılarak elde edilen sonuçlar diğer evrimsel yöntemler kullanılarak elde edilen sonuçlarla ve BP algoritması ve Levenberg-Marquardt yöntemi gibi gradyan tabanlı eğitim yöntemleri ile karşılaştırılmıştır. Çalışmada, ANN eğitiminde diferansiyel evrim algoritmasının uygulanmasının diğer eğitim yöntemlerine alternatif olabileceği gösterilmiştir (Slowik ve Bialko, 2008).

2.2. Sürü Tabanlı Algoritmalar

Metasezgisel algoritmaların başka bir ailesi doğada bulunan canlılardan esinlenen diğer bir deyişe doğadan ilhamlı sürü (popülasyon) tabanlı stokastik optimizasyon algoritmalıdır. Bu algoritmalar oluşturan rastgele başlangıç çözümlerini matematiksel modellerle güncellerler.

Sürü zekası; kuşlar, böcekler, balıklar gibi aralarında etkileşim olan sürülerin davranışlarını modelleyerek, kısıtlamalar altında karmaşık optimizasyon problemlerini makul süre içerisinde çözmeye çalışan bir yapay zeka metodudur. Sürü zekasındaki önemli nokta, sürüdeki bireylerin bireysel olarak değil sürü olarak zeki davranışları sergilemesi, merkezi bir kontrol mekanizması olmadan kendi kendine organize olabilme yeteneğine sahip olmasıdır (Yang, 2010; Karaboğa, 2014). Kuş ve balık sürülerinin konum ve hız bilgilerini birbirleri ile paylaşmaları, karıncaların yuva ve yiyecek kaynağı arasına feromon bırakarak birbirleri ile haberleşmeleri, arıların buldukları besin kaynaklarının yerlerini kovan etrafında birbirlerine iletmeleri, balinaların okyanusta

bulduğu besinlerin bilgilerini spiral dalgalar şeklinde diğer balinalara aktarmaları sürü zekasına örnek gösterilebilen zeki davranışlardır.

Literatürde bir çok sürü zekası optimizasyon algoritması önerilmiştir. Bunlardan popüler olanlarından bazıları burada sunulmuş ve sinir ağlarının eğitiminde yapılan çalışmaları için kısa bir literatür özeti verilmiştir.

Parçacık Sürü Optimizasyonu (PSO), 1995 yılında Eberhart ve Kennedy tarafından geliştirilen, kuş ve balık sürülerinin sosyal davranışından esinlenen, popülasyon tabanlı bir optimizasyon algoritmasıdır. Evrimsel tabanlı algoritmalardan GA ile karşılaştırıldığında, PSO'nun kolay uygulanabilir olması ve ayarlanması gereken parametre sayısının az olması gibi avantajları vardır. PSO fonksiyon optimizasyonu, ANN eğitimi, bulanık sistem kontrolü gibi birçok alana başarıyla uygulanmıştır (Kennedy, 2011). Mendes, Cortez ve arkadaşları PSO'yu ANN eğitiminde önermiş, sınıflandırma ve regresyon problemleri üzerinde önerdikleri yöntemin başarısını göstermişlerdir (Mendes ve ark., 2002).

2003'de doğrusal olmayan bir fonksiyonun optimizasyonu problemi hem PSO eğitimli ANN hem de BP eğitimli ANN ile çözülmüştür. Eğitim algoritmaları kıyaslandığında, PSO algoritmasının BP algoritmasından daha hızlı global optimuma ulaştığını savunulmuştur. (Gudise ve Venayagamoorthy, 2003).

2008'de Evrim Stratejisi algoritmasından esinlenilerek temel PSO geliştirilmiş ve üç katmanlı bir ANN'nin yapısının tasarımında ve ağırlık parametrelerinin optimizasyonunda geliştirilen PSO önerilmiştir. Önerilen yöntemin başarısı iki gerçek dünya problemi üzerinde doğrulanmıştır. (Yu ve ark., 2008).

Karınca Koloni Algoritması (ACO), besin kaynağı ve yuvası arasında en optimum yolu bulmaya çalışan karıncaların davranışlarından esinlenen metasezgisel bir optimizasyon algoritmasıdır. Diego tarafından 1992 'de literatüre kazandırılmıştır. 2005 yılında Blum ve Socha tarafından İleri beslemeli ANN eğitiminde ACO algoritması önerilmiştir. Önerilen yöntem, tıp alanındaki sınıflandırma veri setleri üzerinde test edilerek literatürdeki genel amaçlı optimizasyon tekniklerinden daha avantajlı olduğu gösterilmiştir. (Blum ve Socha, 2005). İki yıl sonra aynı araştırmacılar ileri beslemeli sinir ağlarında önerdikleri ACO algoritmasının performansını, gradyan tabanlı algoritmalarla ve evrimsel algoritmalarla kıyaslamışlardır (Socha ve Blum, 2007).

Yapay Arı Kolonisi (ABC) algoritması bal arılarının yiyecek arama davranışlarından esinlenen popülasyon tabanlı bir metasezgisel optimizasyon algoritmasıdır (Karaboga, 2005). 2007 yılında ileri beslemeli ANN'nin eğitimi için ABC

algoritması önerilmiştir. Önerilen model XOR problemli, 3-Bit parity problemi ve 4-Bit encoder problemi üzerinde Diferansiyel Gelişim (DE) ve PSO algoritmaları ile kıyaslanmış, ABC'nin daha üstün başarısı olduğu savunulmuştur (Karaboga ve ark., 2007). 2011 yılında yapılan bir çalışmada ABC algoritması, Levenberq-Marquardt (LM) algoritması ile melezlenmiş, ANN'yi eğitmek için kullanılmıştır. Önerilen model ABC'nin global arama metodu ile LM'nin geleneksel teknikleri melezleştirerek başarımı artırmaya amaçlamıştır (Ozturk ve Karaboga, 2011).

Gri Kurt Optimizasyon (GWO) algoritması, kurtların sosyal liderliklerinden ve avlanma davranışlarından esinlenen popülasyon tabanlı metasezgisel bir optimizasyon algoritmasıdır. 2014 yılında Mirjalili tarafından geliştirilmiştir (Mirjalili ve ark., 2014a). 2015 yılında yapılan çalışmada ANN'nin eğitiminde GWO kullanılmış, dört sınıflandırma veri seti ve üç doğrusal olmayan fonksiyon üzerinde başarısı gösterilmiştir (Mirjalili, 2015).

Balina optimizasyon algoritması (WOA), kambur balinaların sosyal davranışlarından ve avlanma stratejilerinden esinlenen popülasyon tabanlı metasezgisel bir optimizasyon algoritmasıdır (Mirjalili ve Lewis, 2016). 2018'de WOA, ANN'nin eğitiminde kullanılmış, sınıflandırma veri setleri üzerinde test edilmiştir. Önerilen yöntem, BP algoritması ve altı popülasyon tabanlı algoritma ile karşılaştırılarak ANN eğitimindeki başarısı gösterilmiştir.(Aljarah ve ark., 2018).

Literatürdeki çok çözüm tabanlı algoritmaların, tek çözümlü algoritmalara kıyasla yerel optimumlardan kaçınmada daha üstün başarısı olduğu savunulmuştur (Mirjalili ve ark., 2012). Fakat çok çözümlü tabanlı algoritmalar da yerel minimumlara takılma eğilimi göstermektedir. Literatürde ANN eğitimi için geniş bir yelpazede evrimsel ve sürü tabanlı algoritmalar mevcuttur, ancak yerel minimum problemi halen açıktır. No-free-lunch (NFL) teoremine göre, bir optimizasyon algoritması optimizasyon problemlerinin hepsinde başarı (üstünlük) sağlayamaz (Wolpert ve Macready, 1997; Ho ve Pepyne, 2002).

3. MATERYAL VE YÖNTEM

3.1. Yapay Alg Algoritması

Yapay alg algoritması (AAA) hareketli mikro alglerin karakteristik özelliklerinden ve yaşam davranışlarından ilham alan, metasezgisel bir optimizasyon algoritmasıdır (Uymaz ve ark., 2015).

AAA'da gerçek alglerin hayatta kalması için uygun çevreye doğru hareketleri ve gelişerek çoğalıp yeni nesli üretmesi model alınmıştır. AAA'daki temel öğe alg kolonileri ve hareketleridir.

Alg popülasyonu aşağıdaki gibi bir dizi alg kolonisinden oluşur (Denklem 3.1).

$$\text{Alg kolonisi popülasyonu} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \quad (3.1)$$

Her bir alg kolonisi, bir çözümün elemanları (boyutları) olarak kabul edilen bir grup alg hücresi içerir. Bir kolonideki tüm alg hücreleri, besin kaynağı bulunan uygun çevre şartlarına sahip yere doğru birlikte hareket eden bir bütün olarak düşünülür. Koloni ideal konuma ulaştığında optimum çözüm elde edilmiş olur. Alg kolonileri bir araya gelerek alg popülasyonunu oluşturur. Alg popülasyonu, popülasyondaki birey (alg kolonisi) sayısı kadar çözümün oluşturduğu kümedir. Alg kolonisi kendini hareket ettirerek, geliştirerek ve adapte ederek daha iyi bir konuma geçmeye çalışır.

AAA'daki her alg kolonisi belirli bir büyüklüğe sahiptir. Gerçek alglere benzer şekilde yapay alg kolonileri de bulunduğu çevreye göre büyüklüğünde değişiklik gösterir. Elverişli yaşam koşulları, alg kolonisinin büyüklüğünü artırır. Kötü ve verimsiz çevre koşulları da, alg hücrelerinin ölümüne ve alg kolonisinin küçülmesine yol açar.

Alglerin koloni büyüklüğünü S_i, i = 1, 2, ..., n. olarak temsil edilirse S_i i. Alg kolonisinin koloni büyüklüğünü, n ise popülasyondaki koloni sayısını gösterir. S_i, ilk aşamada 1 olarak ayarlanır ve i. alg kolonisinin uygunluk değerinin değişmesiyle değişir. Daha iyi uyguluk fonksiyonu $f(x_i)$, daha büyük S_i değerinin oluşmasını sağlar. S_i, aşağıdaki Denklem 3.2 – 3.4 kullanılır gibi verilen biyolojik büyüme sürecine göre güncellenir.

$$S_i = size(x_i) \tag{3.2}$$

$$\mu_i = \frac{S_i + 4f(x_i)}{S_i + 2f(x_i)} \tag{3.3}$$

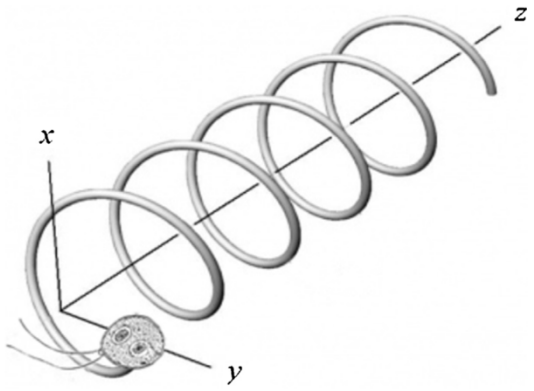
$$S_i^{t+1} = \mu_i S_i^t, \quad i = 1, 2, \ldots, n \tag{3.4}$$

Burada $f(x_i)$ uygunluk fonksiyonu, μ_i S_i*'nin* güncelleme katsayısı, t ise mevcut iterasyonu temsil eder.

AAA helisel hareket, evrimsel süreç, ve adaptasyon süreci olarak üç temel bölümden oluşmaktadır.

3.1.1. Helisel Hareket

Algler, yeterli ışık ve besin maddelerine sahip çevre koşullarında belirli hareketler yaparlar. AAA'da, her bir alg kolonisi, en büyük boyut veya en iyi uygunluk fonksiyon değerine sahip olan en iyi alg kolonisine doğru hareket eder. Gerçek alglerin üç boyuttaki hareketine benzer şekilde, alg kolonisi de üç boyutta hareket eder (Şekil 3.1).



Şekil 3.1. Alglerin üç boyuttaki helisel hareketi (Uymaz, 2015)

Ancak, yapay algler bu hareketi turnuva metoduyla belirlediği bir alg kolonisinin rastgele üç farklı alg hücresi seçip, pozisyonlarını değiştirerek yapar. Denklem 3.5 birinci boyuttaki hareketi temsil eder ve tek boyutlu problemlerde sadece bu denklem kullanılır. İki boyutlu problemlerde Denklem 3.5 ve 3.6 kullanılır. Üç ve daha fazla boyutlu problemlerde Denklem 3.5 – 3.7 kullanılır .

$$x_{im}^{t+1} = x_{im}^{t} + (x_{jm}^{t} - x_{im}^{t})(sf - \omega_i)p \tag{3.5}$$

$$x_{il}^{t+1} = x_{il}^{t} + (x_{jl}^{t} - x_{il}^{t})(sf - \omega_i)\sin\beta \tag{3.6}$$

$$x_{ik}^{t+1} = x_{ik}^{t} + (x_{jk}^{t} - x_{ik}^{t})(sf - \omega_i)\cos\alpha \tag{3.7}$$

Burada *m,k* ve *l,* 1 ve *D* arasında oluşturulmuş rastgele tam sayılardır. x_{im}, x_{il} ve x_{ik} *i*. alg kolonisinin x,y ve z koordinatlarını temsil eder. *J* komsu alg kolonisinin indisini gösterir ve turnuva seçimi ile elde edilir. *P*, 0 ile 1 arasında rastgele bir sayıdır. α ve β 0 ile 2π arasında rastgele derecelerdir. *sf* kesme kuvvetidir. ω_i ise *i.* alg kolonisinin sürtünme yüzey alanıdır ve alg kolonisin büyüklüğü ile orantılıdır.

Alg kolonisinin küresel şekline bağlı olarak yarım kürenin yüzey alanı, Denklem 3.8 ve 3.9 ile hesaplanabilir. Küçük alg kolonilerinin sürtünme yüzey alanı da küçük olduğu için çözüm uzayında daha hızlı hareket ederler. Bu durum keşif (exploration) yeteneğinin fazla olduğunu gösterir. Keşif yeteneği global çözümleri aramak için çözümleri çeşitlendirmektir. Alg kolonileri büyüyüp yüzey alanı arttıkça kolonilerinin sürtünme yüzey alanı da artar. Dolayısıyla koloniler daha yavaş hareket etmeye, çözüm uzayında daha kısa mesafe de konum değiştirmeye başlar. Bu durum sömürü (exploitation) yeteneğini artırır. Sömürü yeteneği yerel optimum çözümleri arama yeteneğidir. Alg kolonileri, sürtünme kuvveti sayesinde, büyüdükçe keşif yeteneğini azaltacak, sömürü yeteneğini artıracak şekilde modellenmiştir.

$$\omega_i = 2\pi r_i^2 \tag{3.8}$$

$$r_i = \left(\sqrt[3]{\frac{3S_i}{4\pi}} \right) \tag{3.9}$$

Bu denklemde r_i alg kolonisindeki yarım kürenin yarı çapını temsil eder S_i'de onun büyüklüğünü gösterir.

Her iterasyona başlarken alg kolonileri, büyüklüklerine göre enerji değerleri hesaplanır. Enerji değeri hesaplanırken algler büyüklerine göre normalize edilir. En küçük büyüklüğe sahip alg kolonisi en küçük enerji değerini alırken, en büyük alg kolonisi de en büyük enerji değerini alır. Alg kolonileri problem uzayında çözüm ararken alg hücrelerini (turnuva metoduyla seçtiği) başka bir alg kolonisinin (rastgele seçtiği)

farklı üç alg hücresiyle değiştirir. Alg kolonisinin her bir helisel hareket çevriminde, çözüm uzayında kaç kez yer değiştireceği (yeni çözüm arayacağı) bu enerji değeri ile belirlenir. Enerjiler besin konsantrasyonu ile doğru orantılıdır. Helisel hareket sürecinden sonra evrimsel süreç başlar.

3.1.2. Evrimsel Süreç

Gerçek alg kolonisi yaşadığı çevrede, yeterli besin kaynağına sahip ise hızla büyür ve gelişir. Yeterli besin kaynağının bulamadığı yaşama elverişsiz ortamlarda ise koloniler ölmeye başlar. Benzer şekilde AAA'da alg kolonileri mevcut konumlarından daha ideal bir konuma hareket ederse ve daha uygun bir çözüm elde ederse, koloniler daha da büyür. Alg kolonilerinin büyüklükleri Denklem 3.4'te hesaplanır. Her bir çevrimde, helisel hareket aşaması tamamlandığında, evrimsel sürece geçilir. Evrimsel süreçte rastgele seçilen bir boyutta, en büyük alg kolonisinin bir alg hücresi, en küçük alg kolonisinde karşılık bir alg hücresine kopyalanır. Bu süreç Denklem 3.10 – 3.12 ile gösterilmektedir.

$$\text{biggest}^t = \arg\ max\{size(x_i^t)\}, \qquad i = 1,2,\ldots,N \tag{3.10}$$

$$\text{smallest}^t = \arg\ min\{size(x_i^t)\}, \qquad i = 1,2,\ldots,N \tag{3.11}$$

$$\text{smallest}_m^{t+1} = \text{biggest}_m^t\ , \qquad m = 1,2,\ldots,D \tag{3.12}$$

t anında büyüklüklerine göre, en küçük alg hücresinin rastgele belirlenen bir boyuttaki (*m*) hücresinin yerine en büyük alg kolonisindeki alg hücresi kopyalanır. Denklemlerde *t* mevcut iterasyonu , *N* popülasyondaki koloni sayısını, *D* problem boyutunu, *biggest* en büyük alg kolonisini, *smallest* en küçük alg kolonisini ve *m* rastgele seçilmiş bir alg hücresinin indeksini temsil etmektedir.

3.1.3. Adaptasyon Süreci

Gerçek alg kolonileri büyüme sürecinde, yetersiz ışık ve besin yoğunluğu ortamında aç kalırlar. Adaptasyon, aç kalan alg kolonilerinin, kendilerini çevreye adapte etmeye ve büyük kolonilere doğru hareket etmeye çalıştıkları bir süreçtir. Benzer şekilde AAA'da uygunluk fonksiyonundan iyi değerler alamayan alg kolonilerinin açlık seviyeleri artar. AAA'da her bir alg kolonisi, bir açlık değerine algoritmadaki ismiyle

starvation parametresine sahiptir. Açlık parametresi değeri başlangıçta sıfıra ayarlanır ve helisel hareket çevriminde artar. Helisel hareket, alg kolonisinin daha iyi bir pozisyona ya da daha kötü bir pozisyona gitmesine neden olur. Eğer alg kolonisi daha iyi bir konuma giderse yani daha iyi bir uyguluk değeri elde ederse, ilgili alg kolonisinin açık değerinde bir değişiklik olmaz. Aksi halde çevrim sonunda daha iyi bir uygunluk değeri bulamayan alg kolonisinin açlık değeri bir artar.

Adaptasyon sürecinde, bir çevrim tamamlandıktan sonra, en aç (en yüksek açlık değerine sahip olan) alg kolonisi, kendisini en büyük alg kolonisine adapte etmeye çalışır. Bu süreç Denklem 3.13 ve 3.14 ile gösterilmektedir.

$$\text{starving}^t = \arg\max\{starvation(x_i)\}, \quad i = 1, 2, \ldots, N \tag{3.13}$$

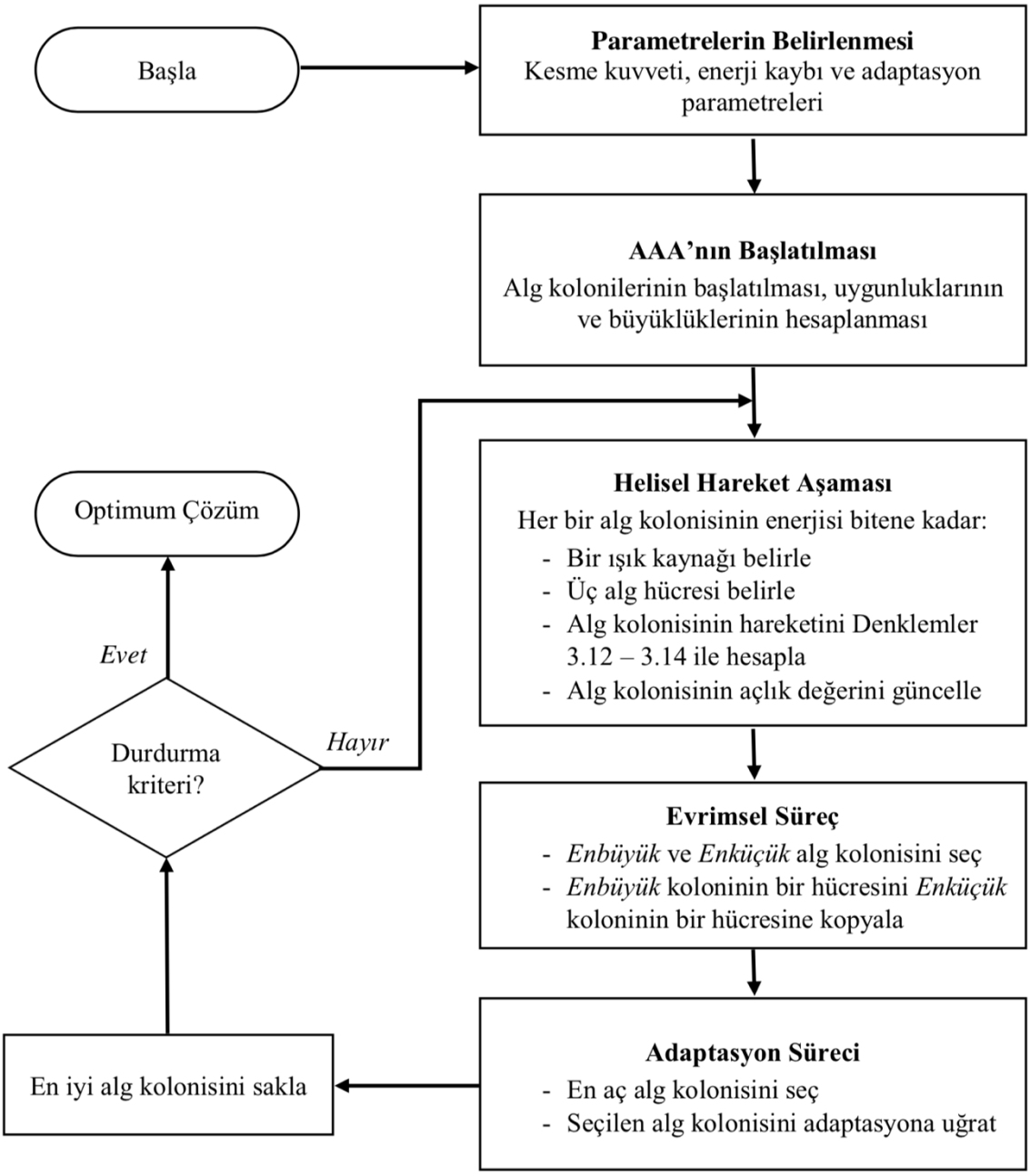
$$\text{starving}^{t+1} = \text{starving}^t + (\text{biggest}^t - \text{starving}^t) \times \text{rand} \tag{3.14}$$

Denklemde, popülasyondaki alg kolonilerinin açlık seviyeleri $starvation(xi)$ ile, *t* anındaki açlık değeri en yüksek alg kolonisi *starving*^{t} *ile,* en büyük alg kolonisi *biggest*^{t} ile gösterilmiştir. *Rand* 0 ile 1 arasında rastgele seçilen bir sayıdır.

Adaptasyon parametresi (Ap), başlangıçta belirlenen sıfır ile bir arasında sabit bir değerdir. Adaptasyon işlemi sıfır ile bir arasında rastgele seçilen bir sayının başlangıçta belirlenen adaptasyon parametresinden büyük olduğu durumda gerçekleşir. Özetle Adaptasyon, hayatta kalan fakat yeterince büyüyemeyen alg kolonisinin bulunduğu ortamdaki en büyük alg kolonisine adapte olmaya çalıştığı bir süreçtir.

3.1.4. Yapay Alg Algoritmasının Sözde Kodu ve Akış Şeması

AAA'nın akış şeması Şekil 3.2'de ve sözde kodu Şekil 3.3'te gösterilmektedir.



Şekil 3.2. Yapay Alg Algoritmasının akış şeması (Uymaz, 2015)

Yapay Alg Algoritması - Artificial Algae Algorithm (AAA)

Amaç fonksiyonu $f(x)$, $x = (x_1, x_2, \ldots, x_d)$
Parametreleri belirle (kesme kuvveti Δ, enerji kaybı e ve adaptasyon parametresi A_p)
n adet alg kolonisinden oluşan populasyonu rastgele çözümler ile başlat
Herbir alg kolonisinin büyüklüğünü 1 ve açlık değerini 0 olarak tanımla
Herbir alg kolonisinin amaç fonksiyon değerini ve büyüklüğünü (G) hesapla
While (t < MaksimumHesaplama)
  *A*lg kolonilerinin enerji (E) ve sürtünme yüzeylerini (τ) hesapla
  For i=1:n
    iStarve = 1
    While ($E(x_i) > 0$)
      Amaç fonksiyon değerlerine göre turnuva yöntemi ile j alg kolonisini (ışık kaynağını) seç
      Helisel hareket için rastgele üç boyut (alg hücresi) (k, l ve m) seç

$$x_{im}^{t+1} = x_{im}^{t} + \left(x_{jm}^{t} - x_{im}^{t}\right)(\Delta - \tau_i)p$$
$$x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)(\Delta - \tau_i)\cos\alpha$$
$$x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)(\Delta - \tau_i)\,sin\,\beta$$

      α, β ∈ [0,2π]; p ∈ [-1,1];
      Yeni çözümü hesapla
      $E(x_i) = E(x_i) - (\frac{e}{2})$ hareket kaynaklı enerji kaybı
      if yeni çözüm daha iyi, i alg kolonisini güncelle ve *iStarve* = 0
      else $E(x_i) = E(x_i) - (\frac{e}{2})$ metabolizma kaynaklı enerji kaybı **end if**
    end While
    if *iStarve*=1, $A(x_i)$ açlık seviyesini bir arttır **end if**
  end For
  Alg kolonilerinin büyüklüklerini (G) hesapla
  Rastgele belirlenen r boyutunda en büyük alg kolonisinin hücresini en küçük alg kolonisinin hücresine kopyala
  $smallest_r^t = biggest_r^t$
  En aç alg kolonisini seç
  if rand<A_p

$$starving^{t+1} = starving^{t} + (biggest^{t} - starving^{t}) \times rand$$

  end if
  En iyi çözümü sakla
end While

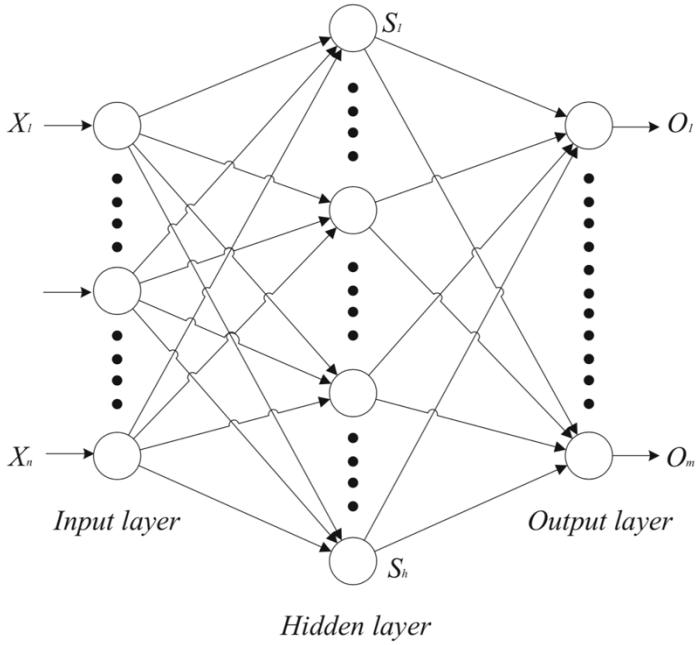
Şekil 3.3. Yapay Alg Algoritmasının sözde kodu (Uymaz, 2015)

3.2. İleri Beslemeli Yapay Sinir Ağları

İleri beslemeli ANN, denetimli (supervised) ANN'lerin özelleşmiş bir formudur. İleri beslemeli ANN'ler, nöron adı verilen bir dizi işlem elemanından oluşur. Nöronlar, her katmandan bir sonraki katmana çeşitli şekillerle bağlanır. En sık kullanılan bağlanma yöntemi, bir katmandaki tüm nöronların bir sonraki katmandaki tüm nöronlara tamamen bağlanmasıdır. İlk katmana giriş katmanı denir ve bu katman, problemin girişlerini sinir ağına eşler. Son katmana çıkış katmanı denir ve problemin çıkışlarını gösterir. Giriş katmanı ile çıkış katmanı arasındaki tüm katmanlara gizli katman adı verilir (Basheer ve Hajmeer, 2000; Panchal ve ark., 2011).

Bir sinir ağında kaç gizli katmanı olacağına ve gizli katmanda kaç nöron bulunacağına problemin karmaşıklığına göre karar verilir. Ara katman sayısı ve bu katmandaki nöron sayısı arttıkça daha karmaşık problemlerin çözümünü mümkün kılmakla birlikte hesaplama maliyetini de artırmaktadır. Problemin giriş sayısı, giriş katmanındaki nöron sayısına, problemin çıkış sayısı da çıkış katmanındaki nöron sayısına eşittir. Gizli katmandaki nöron sayısının belirlenmesinde kesin bir kural olmamakla birlikte literatürde giriş katmanındaki nöron sayısının bir fazlası veya iki katının bir fazlası kadar nöron seçilmesi önerilmiştir (Wdaa ve Sttar, 2008; Mirjalili ve ark., 2014b). Gizli katmandaki nöron sayısının artması eğitim süresini artırmaktadır.

İleri beslemeli ANN'lerde bilgi tek bir yönde ileriye doğru aktarılır. Nöronların çıkışlarının kendilerini beslediği geri besleme bağlantıları yoktur. Çok Katmanlı Algılayıcı (Multi Layer Perceptron, MLP) ileri beslemeli ANN'lerin en yaygın türüdür. MLP'de nöronlar tek yönde ve ileri doğru birbirine bağlanır. Bağlantılar yani ağırlıklar genelde −1 ile 1 aralığındaki gerçek sayılardan oluşur. Şekil 3.4'te, tek gizli katmana sahip olan MLP'nin bir örneği gösterilmiştir.



Şekil 3.4. Tek gizli katmana sahip bir MLP

MLP çalışma prensibi olarak, kendisine sunulan giriş verilerinin istenilen çıktısını üretmeyi amaçlar. Bunu yapabilmesi için sistemin öncelikle kendisine sunulan giriş ve çıkış verilerine bakarak parametrelerini güncellemesi (eğitmesi) gerekir. Eğitim işlemi hata miktarı belirli bir miktara düşünceye kadar devam eder. Eğitim sonunda bulunan parametrelerle MLP, kendisine sunulan girişlerin çıkışını üretir.

MLP'nin çıkışı aşağıdaki adımlarla hesaplanır:

1. Girdilerin ağırlıkları toplamı Denklem 3.15 ile hesaplanır.

$$s_j = \sum_{i=1}^{n} (W_{ij}.X_i) - \theta_j, \quad j = 1, 2, \ldots, h \tag{3.15}$$

Denklemde, n giriş nöronlarının sayısını, W_{ij} giriş katmanındaki *i.* nörondan gizli katmandaki *j.* nörona bağlantı ağırlığını, θ_j *j.* gizli nöronun eşik (bias) değerini, X_i *i.* girişi temsil eder.

2. Her bir gizli nöronun çıkışı, Denklem 3.16 ile hesaplanır.

$$S_j = sigmoid(s_j) = \frac{1}{(1 + exp(-s_j))}, \quad j = 1, 2, \ldots h \tag{3.16}$$

3. Ağın çıktıları, gizli nöronun çıktıları kullanılarak Denklem 3.17'de hesaplanır. Toplama fonksiyonunun değerine bağlı olarak nöronların çıkışını tetiklemek için bir aktivasyon fonksiyonu kullanılır. MLP'de tanjant hiperbolik, sinüs gibi farklı aktivasyon fonksiyonları tercih edilebilir. Literatürde kolay türevlenebilir olması ve doğrusal olmaması dolayısıyla en çok tercih edilen aktivasyon fonksiyonu sigmoid fonksiyonudur.

$$o_k = \sum_{j=1}^{h} (W_{jk}.S_j) - \theta'_k, \quad k = 1, 2, \ldots, m \tag{3.17}$$

$$O_k = sigmoid(o_k) = \frac{1}{(1 + exp(-o_k))}, \quad k = 1, 2, \ldots, m \tag{3.18}$$

Denklemde , W_{jk} *j.* gizli katmandan *k.* çıkış nöronuna bağlantı ağırlığını, θ_k *k.* çıkış nöronun eşik (bias) değerini gösterir.

MLP'nin en önemli parametreleri bağlantı ağırlıkları ve eşik değerleridir. Ağırlık ve eşik değerleri çıktıların nihai değerlerini belirler. MLP eğitimi, belirli girdilerden istenen çıktıları elde etmek için diğer bir ifadeyle giriş ve çıkışlar arasındaki istenen ilişkiyi kurmak için, optimum ağırlık ve eşik değerlerinin bulunması sürecidir.

3.3. Yapay Sinir Ağlarının Yapay Alg Algoritması ile Eğitimi

Literatürde MLP'lerin eğitimde sezgisel algoritmalar üç farklı şekilde kullanılabilir. Birinci yöntem, MLP'nin ağırlık ve eşik değerlerinin bulunmasında sezgisel algoritmaların kullanılmasıdır. İkinci yöntem, belirli bir problemde MLP'ye uygun bir mimari tasarlamak için sezgisel algoritmaların kullanılmasıdır. Üçüncü yöntem, öğrenme hızı ve momentum gibi parametrelerin ayarlanmasında sezgisel bir algoritmanın kullanmasıdır.

İlk yöntemde, öğrenme sürecinde ağın mimarisi değişmez. MLP'nin en yüksek başarısını sağlayan optimum parametrelerinin (ağırlıklar ve eşik değerleri) bulunması için sezgisel bir optimizasyon algoritması ile eğitim yapılır. Bu yöntem ile yapılan bir çok çalışma, kaynak araştırmasında verilmiştir.

İkinci yaklaşımda, MLP'nin mimari yapısı değişmektedir. Bu yöntemde, sezgisel bir eğitim algoritması belirli bir problemi çözmek için en uygun mimari yapıyı bulmaya

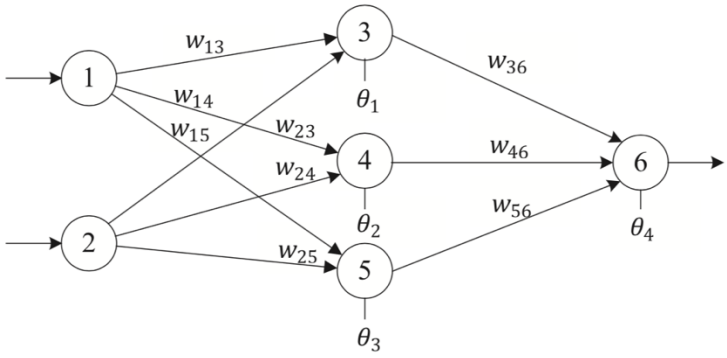
çalışır. Ağ yapısı, gizli katman sayısının ve her katmandaki nöron sayısının farklılaşmasıyla değişir. Bu yönteme örnek olarak, iki gerçek dünya problemin MLP ile çözümünde, uygun ağ yapısının tasarımda PSO algoritması kullanılmıştır (Yu ve ark., 2008).

Bu tez çalışmasında, birinci yöntem kullanılarak tek bir gizli katmana sahip MLP ağını eğitmek için AAA önerilmiştir. Önerilen yaklaşımda iki önemli nokta vardır.

1. Temsil Stratejisi : Ağırlık ve eşik değerlerinin AAA'daki arama ajanlarına uygun formda (alg kolonileri) temsil edilmesi gerekir.
2. Uygunluk Fonksiyonu : Alg kolonilerini değerlendirmek için MLP'de oluşan hatayı kullanan bir uygunluk fonksiyonu tanımlanmalıdır.

3.3.1. Yapay Alg Algoritması için Temsil Stratejisi

Literatürde MLP'nin ağırlıklarını ve eşik değerlerinin temsil edilmesi için vektör kodlama, matris kodlama ve ikili kodlama olmak üzere üç yöntem vardır. (Zhang ve ark., 2007). Vektör kodlamada her birey bir vektör olarak kodlanır. Her bir vektör, bir MLP'nin tüm ağırlıklarını ve eşik değerlerini temsil eder. Matris kodlamada, her birey bir matris olarak temsil edilir. İkili kodlama da, her birey ikili bit dizeleri olarak temsil edilir. Bu stratejilerin her birisinin seçilen probleme göre avantajları ve dezavantajları mevcuttur (Mirjalili ve ark., 2012). Bu çalışmada, literatürde en çok tercih edilen vektör kodlama stratejisi kullanılmıştır. Şekil 3.5'te gösterilen MLP'nin vektör olarak temsil edilmesi Denklem 3.19'da gösterilmiştir.



Şekil 3.5. Üç katmanlı MLP örneği

$$\text{alg kolonisi} = [w_{13}w_{23}w_{14}w_{24}w_{15}w_{25}w_{36}w_{46}w_{56}\theta_1\theta_2\theta_3\theta_4] \quad (3.19)$$

Alg kolonilerini vektör formunda temsil ettikten sonra, her birinin değerlendirilmesi için bir uygunluk (amaç) fonksiyonu gereklidir.

3.3.2. Uygunluk Fonksiyonunun Belirlenmesi

Öğrenme eğitim algoritmalarının amacı, doğrulama ve test verilerini başarılı bir şekilde sınıflandırabilen bir MLP eğitmektir. Öğrenme aşamasında eğitim kümesi oldukça önemlidir. Çünkü kümedeki her bir eğitim örneği, çözüm olarak temsil edilen vektörün uygunluk değerinin hesaplanmasında kullanılır. Tüm eğitim örneklerinin değerlendirilmesi için Denklem 3.20'de gösterilen ortalama karesel hata kullanılmıştır.

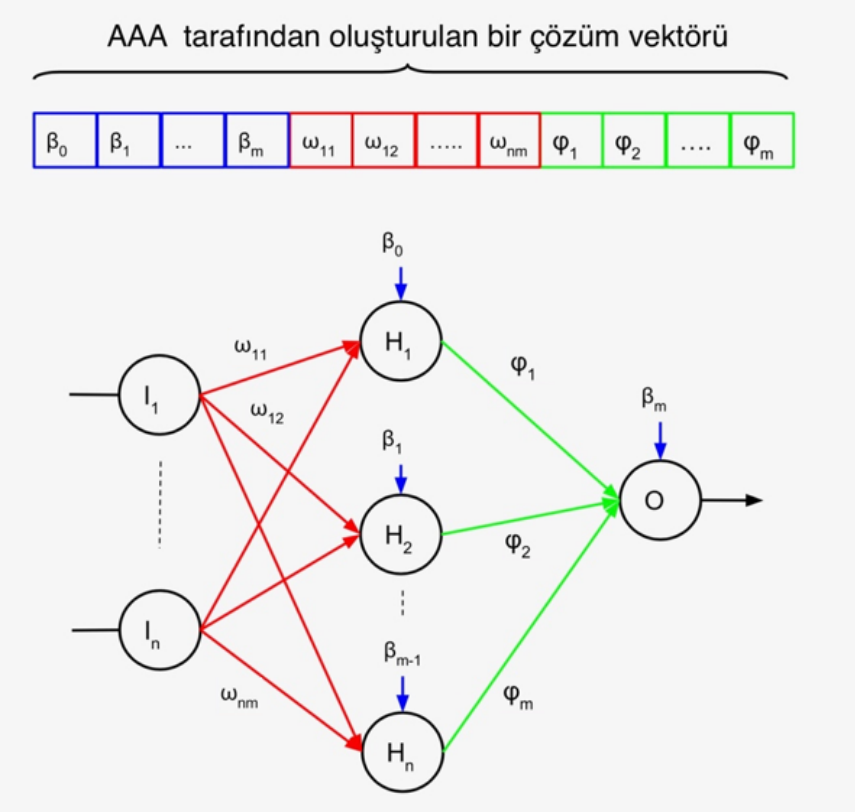
$$E = \sum_{k=1}^{q} \frac{\sum_{i=1}^{m} (o_i^k - d_i^k)^2}{q} \quad (3.20)$$

Denklemde *E (MSE)* hatayı, *q* eğitim örneklerinin sayısını, *m* çıkış sayısını, d_i^k *k.* eğitim örneğinde *i.* girişin istenen çıkışını, o_i^k *k.* eğitim örneğinde *i.* girişin gerçek çıkışını temsil etmektedir.

3.3.3. Önerilen Model

AAA'ya dayalı olarak önerilen modelde her bir arama ajanı (alg kolonisi), bir aday çözümü (MLP) temsil eder. Bu aday çözüm vektörü, tek boyutlu bir dizi olarak temsil edilir. Aday çözüm vektörü üç bölümden oluşmaktadır. Birinci kısım nöronların sahip olduğu eşik değerlerini, ikinci kısım giriş katmanı ile gizli katman arasındaki ağırlık değerlerini, üçüncü kısım gizli katman ile çıkış katmanı arasındaki ağırlık değerlerini göstermektedir. Vektörlerin büyüklükleri birbirine eşittir ve ağdaki toplam ağırlık ve eşik değeri sayısı kadardır.

Şekil 3.6'da mavi renkli oklar eşik değerlerini, kırmızı renkli oklar giriş ile gizli katman arasındaki ağırlıkları ve yeşil renkli oklar gizli katman ile çıkış katmanı arasındaki ağırlıkları göstermektedir.



Şekil 3.6. Çözümü temsil eden vektörün MLP'ye yerleştirilmesi (Aljarah ve ark., 2018)

Denklem 3.21'de MLP'nin oluşturduğu vektörün, uzunluğunun hesaplandığı formül gösterilmektedir. Çalışmadaki tüm MLP'lerin tek bir gizli katmanı bulunmaktadır.

$$\text{Vektör uzunluğu} = (n \times m) + (2 \times m) + 1 \qquad (3.21)$$

Denklemde *n* ağın giriş sayısını , *m* ise gizli katmandaki nöron sayısını temsil etmektedir.

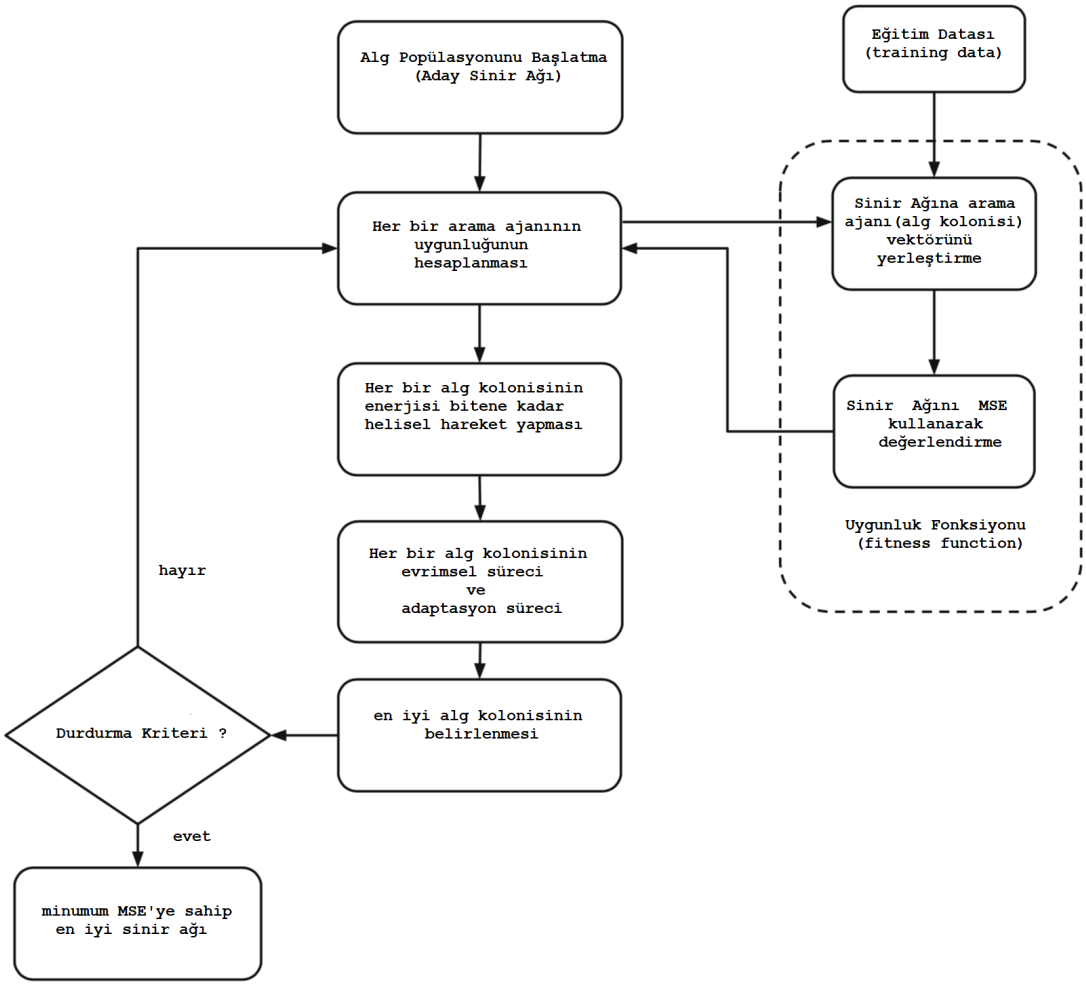
Oluşturulan vektörlerin uygunluk değerlerini belirlemek için ortalama karesel hata (Mean Square Error, MSE) fonksiyonu kullanılmıştır. MSE, gerçek ve tahmin edilen değerlerin arasındaki karesel farklı hesaplamaya dayanır. Denklem 3.20'de gösterilmektedir.

Önerilen modelin adımları aşağıdaki gibidir.

1. Başlatma: Başlangıçta tanımlanmış sayı (popülasyondaki birey sayısı) kadar ajan (alg kolonisi) rastgele oluşturulur. Her ajan bir MLP ağını temsil eder.

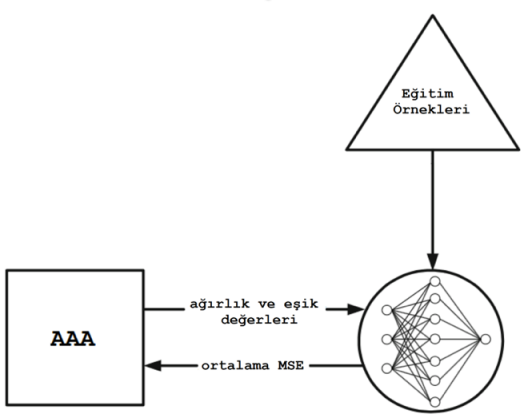
2. Uygunlukların Değerlendirilmesi: Üretilen MLP'lerin başarısı uygunluk fonksiyonu (MSE) kullanılarak değerlendirilir. Bu adım uygulanırken vektörleri oluşturan ağırlık ve eşik değerleri kümesi önce MLP'ye aktarılır, sonra her ağ uygunluk fonksiyonu ile değerlendirilir. Eğitim algoritmasının amacı, veri kümesindeki eğitim örneklerini modelleyen minimum MSE ye sahip MLP'yi bulmaktır.
3. Güncelleme: Arama ajanlarının konumları, AAA'daki denklemlere göre güncellenir.
4. Döngü: Maksimum iterasyon veya belirli hata miktarına ulaşıncaya kadar adımlar tekrar edilir. Son olarak, minimum MSE değerine sahip MLP ağı veri setinin test için ayrılan kısmı üzerinde test edilir.

Şekil 3.7'de önerilen yaklaşımının genel adımları gösterilmektedir.



Şekil 3.7. Önerilen yöntemin akış şeması

Sonuç olarak AAA, MLP'ye ağırlık ve eşik değerlerini sunar ve geri dönüş olarak ağdaki oluşan hatayı alır. AAA, tüm eğitim örneklerinin ortalama hatasını (MSE) en aza indirmek için ağırlıkları ve eşik değerlerini iterasyonlar boyunca günceller. Şekil 3.8'de önerilen yöntemin temsili şeması gösterilmiştir.



Şekil 3.8. Önerilen yöntemin temsili şeması

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu bölümde MLP'leri eğitmek için önerilen AAA yaklaşımı, farkı zorluk seviyelerine sahip on sınıflandırma problemi üzerinde değerlendirilmiştir. Seçilen veri setleri, eğitim algoritmalarının performansını farklı koşullarda test etmek için farklı sayıda özelliğe ve örneğe sahiptir. Bu durum eğitim sürecini daha zor hale getirmektedir. Sınıflandırma veri setleri, Kaliforniya Üniversitesi Irvine (UCI) Makine Öğrenmesi Veritabanı'dan (http://archive.ics.uci.edu/ml) alınan Australian, Blood, Breast cancer, Chess, Diabetes, Ionosphere, Liver, Parkinsons, Tic-tac-toe ve Vertebral'dır (Asuncion ve Newman, 2007). Çizelge 4.1, bu veri setlerini sınıf, özellik, eğitim örneği ve test örneği sayısı bakımından sunmaktadır. Sonuçların istatistiksel testleri, yerel minimumlara takılma eğilimi, en iyi ve ortalama sınıflandırma başarısı, standart sapma değerleri, yakınsama eğrisi ve çalıştırma zamanına göre kıyaslanarak yorumlanmıştır. Eğitim için önerdiğimiz AAA'nın başarısını doğrulamak için Parçacık Sürü Optimizasyonu (Particle Swarm Optimization, PSO), Balina Optimizasyon Algoritması (Whale Optimization Algorithm, WOA), Güve Optimizasyon Algoritması (Mont Flame Algorithm, MFO), Çoklu Evren Algoritması (Multi Verse Optimization, MVO), Yarasa Optimizasyon Algoritması (Bat Optimization Algorithm, BAT), Guguk Kuşu Optimizasyon Algoritması (Cuckoo Search Optimizasyon Algorithm, CS) ve Gri Kurt Optimizasyon Algoritması (Grey Wolf Optimizer, GWO) ile bahsi geçen veri setleri üzerinde ANN eğitimi yapılarak kıyaslanmıştır.

4.1. Çalışma Ortamının Hazırlanması

Tüm algoritmaların veri setleri üzerindeki performanslarının kıyaslanması için gerekli kodlamalar Python'da geliştirildi. Kodlamalardaki dizi manipülasyonu için NumPy kütüphanesi, matris desteği için de SciPy kütüphanesini kullanıldı. Ayrıca algoritmanın ANN kısmının kodlanması için Neurolab kullanıldı. Literatürdeki popüler metasezgisel algoritmaların kodları için EvoloPy'dan yararlanıldı. EvoloPy, doğadan ilhamlı metasezgisel optimizasyon algoritmalarının kodlanıldığı Python'da geliştirilmiş açık kaynak kodlu ve cross platform bir Framework'dür. Matlab ile kıyaslandığında çok boyutlu problemlerde çalışması süresi açısından daha makuldur. EvoloPy ile ilgili tüm detaylara ve karşılaştırma testlerine Github üzerinden ulaşılabilir (Faris ve ark., 2016).

Çizelge 4.1. Sınıflandırma Veri setleri

Veri Seti	Sınıf	Özellik	Eğitim Örneği	Test Örneği
Australian	2	14	455	235
Blood	2	4	493	255
Breast cancer	2	8	461	238
Chess	2	36	2109	1087
Diabetes	2	8	506	262
Ionosphere	2	33	231	120
Liver	2	6	79	41
Parkinsons	2	22	128	67
Tic-tac-toe	2	9	632	326
Vertebral	2	6	204	106

Tüm veri setlerinin, sınıflandırma problemlerinde eğitim ve test için ayrılması gerekir. Literatürde eğitim ve test verisini ayırmak için iki yöntem mevcuttur. Birincisi, veri setini eğitim ve test olarak sabit bir yüzde ile ayırmaktır. Bu yöntemin en popüler olanı eğitim için % 66 ve test için % 34 olarak ayırmaktır. İkinci yöntem ise k parçalı çapraz doğrulamadır. Bu yöntemde veri seti k eşit parçaya ayrılır ve k kez çalıştırılır. Her seferinde k-1 parça ile eğitilir ve eğitilmeyen kısmı üzerinde testi yapılır. Bu yöntem daha güvenilir olmakla birlikte doğrulama süresini k katına çıkartmaktadır (Kohavi, 1995).

Literatürdeki ANN'nin sezgisel yöntemlerle eğitilmesinde birinci yöntem kullanılmaktadır. Çünkü ANN'nin eğitimi uzun süren bir süreç olmakla birlikte popülasyon tabanlı algoritmalarla eğitimi, süreyi daha da uzatmaktadır. Dahası yapılan doğrulamanın akademik düzeyde olması için otuz kez birbirinden bağımsız olarak çalıştırılması gerekmektedir. Bu sebeplerden ötürü bu tez çalışmasında birinci yöntem kullanılarak veri setinin %66'sı eğitim, %34ü test için ayrılmıştır. İlk olarak eğitim için ayrılan kısmında ANN'nin eğitimi yapılmış, test için ayrılan kısım üzerinde de başarısı test edilmiştir.

Tüm veri setlerinin farklı ölçeklere sahip olan özelliklerinin etkisini ortadan kaldırmak için Denklem 4.1'deki minimum-maksimum normalizasyonu kullanılarak veri setleri ölçeklenmiştir. Tüm algoritmalar, tüm veri setleri üzerinde birbirinden bağımsız başlangıç parametreleri ile 30 kez çalıştırılmıştır. Popülasyondaki birey sayısı 40 olarak belirlenmiştir. Her bir çalıştırma 250 iterasyon, 10000 uygunluk hesaplaması (fitness calculation) yapılmıştır. Çalışmadaki parametreler belirlenirken (iterasyon sayısı, birey sayısı gibi) literatürde kabul gören çalışmalar örnek alınmıştır (Mirjalili ve ark., 2014b; Mirjalili, 2015; Aljarah ve ark., 2018). ANN'nin ağırlık ve eşik değerleri [-1, 1] arasındaki reel sayılardır.

$$v' = \frac{v_i - \min_A}{\max_A - \min_A} \tag{4.1}$$

Burada v ′, v'nin minA ile maxA arasında normalleştirilmiş değeridir.

Literatürde MLP'nin gizli katmanındaki nöronların sayısını seçmek için farklı yaklaşımlar önerilmiştir. Ancak önerilen yaklaşımların hangisinin üstün olduğu konusunda kesinlik yoktur. Yapılan çalışmalarda *N+1* ve *2* x *N* +1 seçim yaklaşımları mevcuttur. *N,* giriş katmanındaki nöron sayısını temsil etmektedir. Bu çalışmada gizli katmandaki nöron sayısı *2* x *N* +1 kadar seçilmiştir (Wdaa ve Sttar, 2008; Mirjalili ve ark., 2014b; Aljarah ve ark., 2018). Bu yöntem seçildiğinde, her bir veri seti için oluşan MLP yapısı Çizelge 4.2'de gösterilmiştir. Aynı zamanda çizelgede her bir MLP'nin vektör uzunluğu da Denklem 3.21'deki formüle göre hesaplanarak gösterilmiştir.

Çizelge 4.2. Veri setleri için oluşan MLP yapısı

Veri Seti	Özellik	Ağ Yapısı (giriş-gizli-çıkış)	Vektör Boyutu
Australian	14	14-29-1	465
Blood	4	4-9-1	55
Breast cancer	8	8-17-1	171
Chess	36	36-73-1	2775
Diabetes	8	8-17-1	171
Ionosphere	33	33-67-1	2346
Liver	6	6-13-1	105
Parkinsons	22	22-45-1	1081
Tic-tac-toe	9	9-19-1	210
Vertebral	6	6-13-1	105

4.2. Deneysel Sonuçlar

Sonuçlar, bağımsız otuz çalışmanın ortalama sınıflandırma başarı, en iyi sınıflandırma başarısı, standart sapma değeri, yakınsama eğrisi, eğitim süresi ve Wilcoxon testi üzerinden yorumlanmıştır.

Çizelge 4.3'te algoritmaların ortalama sınıflandırma başarısı, en iyi sınıflandırma başarısı ve standart sapma değerlerini gösterilmiştir. Çizelgedeki değerler veri setlerinin eğitim için ayrılan kısmında otuz bağımsız çalışmanın sonuçlarıdır.

Çizelge 4.3. Veri setlerinin eğitim için ayrılan kısmında algoritmaların 30 bağımsız çalışmada ki ortalama başarı, en iyi başarı ve standart sapma değerleri

Algoritma / Veri Seti		AAA	PSO	WOA	MFO	MVO	BAT	CS	GWO
Australian	Ort.	0.870	0.850	0.561	0.876	**0.891**	0.608	0.858	0.557
	Std.	0.011	0.014	0.098	0.009	**0.007**	0.192	**0.007**	0.056
	Eniyi	**0.894**	0.874	0.804	0.892	0.907	**0.894**	0.874	0.854
Blood	Ort	**0.788**	0.787	0.715	0.786	0.786	0.665	**0.788**	0.778
	Std.	0.003	0.003	0.162	0.001	0.001	0.211	0.003	**0.000**
	Eniyi	**0.791**	**0.791**	0.778	**0.791**	**0.791**	0.782	0.791	0.778
Breast cancer	Ort.	**0.960**	0.958	0.677	0.959	0.958	0.538	0.956	0.652
	Std.	0.003	0.003	0.083	0.003	0.003	0.247	0.004	**0.000**
	Eniyi	**0.967**	0.965	0.965	0.965	0.963	0.965	0.965	0.652
Chess	Ort.	0.716	0.715	0.484	0.760	**0.859**	0.625	0.731	0.472
	Std.	0.023	0.035	0.030	0.016	**0.010**	0.195	0.020	0.048
	Eniyi	0.761	0.767	0.531	0.796	0.884	**0.945**	0.761	0.676
Diabetes	Ort.	0.780	0.782	0.413	0.785	**0.788**	0.517	0.776	**0.788**
	Std.	0.007	0.007	0.135	0.005	**0.005**	0.181	0.008	**0.005**
	Eniyi	0.796	0.794	0.660	0.798	**0.800**	0.794	0.788	0**.800**
Ionosphere	Ort.	0.875	0.840	0.451	0.928	**0.964**	0.492	0.846	0.670
	Std.	0.022	0.053	0.169	0.020	0.009	0.211	0.025	**0.005**
	Eniyi	0.913	0.913	0.670	0.965	**0.978**	0.961	0.904	0.679
Liver	Ort.	0.705	**0.727**	0.456	0.725	0.725	0.556	0.688	0.418
	Std.	0.014	0.010	0.069	0.011	0.008	0.131	0.013	**0.000**
	Eniyi	0.731	**0.748**	0.581	0.740	0.740	0.731	0.718	0.418
Parkinsons	Ort.	0.859	0.847	0.565	0.881	**0.905**	0.594	0.850	0.750
	Std.	0.023	0.026	0.239	0.020	0.017	0.221	0.023	**0.000**
	Eniyi	0.914	0.890	0.750	0.914	**0.937**	0.890	0.890	0.750
Tic-tac-toe	Ort.	0.740	0.743	0.387	0.770	**0.810**	0.636	0.710	0.346
	Std.	0.013	0.017	0.095	0.013	0.014	0.151	0.010	**0.000**
	Eniyi	0.764	0.784	0.653	0.799	**0.833**	0.765	0.740	0.346
Vertebral	Ort.	0.834	0.838	0.607	**0.836**	**0.835**	0.636	0.822	0.650
	Std.	0.010	0.009	0.122	0.005	**0.005**	0.165	0.019	0.059
	Eniyi	0.857	**0.867**	0.661	**0.848**	0.852	0.838	0.862	0.661

Çizelge 4.4'te Veri setlerinin test için ayrılan kısmında algoritmaların ortalama sınıflandırma başarısı, en iyi sınıflandırma başarısı ve standart sapma değerleri gösterilmiştir.

Çizelge 4.4. Veri Setlerinin test için ayrılan kısmında algoritmaların 30 bağımsız çalışmada ki ortalama başarı, en iyi başarı ve standart sapma değerleri

Algoritma / Veri Seti		AAA	PSO	WOA	MFO	MVO	BAT	CS	GWO
Australian	Ort.	0.831	0.817	0.562	0.832	**0.843**	0.599	0.839	0.579
	Std.	0.013	0.021	0.095	0.016	**0.007**	0.177	0.012	0.052
	Eniyi	0.855	0.851	0.791	0.851	0.855	0.859	**0.863**	0.859
Blood	Ort	0.742	0.743	0.675	**0.745**	0.741	0.636	**0.745**	0.729
	Std.	0.003	0.004	0.135	0.009	0.004	0.175	0.004	**0.000**
	Eniyi	0.749	0.749	0.729	**0.764**	0.752	0.737	0.752	0.729
Breast cancer	Ort.	0.976	0.968	0.681	0.976	**0.977**	0.537	0.970	0.659
	Std.	0.006	0.007	0.083	0.007	**0.005**	0.255	0.007	0.000
	Eniyi	**0.991**	0.987	0.966	0.987	0.987	0.974	0.987	0.659
Chess	Ort.	0.691	0.678	0.492	0.719	**0.829**	0.619	0.715	0.495
	Std.	0.029	0.037	0.022	0.018	**0.016**	0.191	0.023	0.046
	Eniyi	0.749	0.744	0.507	0.749	0.856	**0.938**	0.751	0.666
Diabetes	Ort.	0.749	0.746	0.427	0.753	**0.754**	0.516	0.739	0.366
	Std.	0.010	0.015	0.112	0.010	0.005	0.152	0.013	**0.001**
	Eniyi	0.767	**0.770**	0.633	**0.770**	0.763	0.759	0.767.	0.374
Ionosphere	Ort.	0.839	0.790	0.469	0.861	**0.897**	0.511	0.808	0.582
	Std.	0.038	0.046	0.094	0.032	**0.026**	0.151	0.039	0.009
	Eniyi	0.925	0.875	0.583	0.933	**0.950**	0.925	0.891	0.608
Liver	Ort.	0.758	0.750	0.459	0.760	**0.765**	0.580	0.713	0.423
	Std.	0.022	0.015	0.065	0.017	0.014	0.153	0.025	**0.000**
	Eniyi	**0.805**	0.779	0.576	0.788	0.788	0.771	0.762	0.423
Parkinsons	Ort.	0.835	0.789	0.562	0.838	**0.867**	0.592	0.812	0.761
	Std.	0.042	0.044	0.258	0.036	0.026	0.231	0.035	**0.000**
	Eniyi	**0.910**	0.880	0.761	0.895	**0.910**	0.895	0.865	0.761
Tic-tac-toe	Ort.	0.688	0.687	0.389	0.706	**0.747**	0.613	0.674	0.346
	Std.	0.026	0.027	0.097	0.024	0.019	0.129	0.022	**0.000**
	Eniyi	0.736	0.754	0.653	0.754	**0.794**	0.379	0.717	0.346
Vertebral	Ort.	0.873	0.869	0.638	**0.881**	**0.881**	0.661	0.835	0.693
	Std.	0.017	0.016	0.157	0.013	**0.011**	0.194	0.035	0.075
	Eniyi	**0.905**	**0.905**	0.707	**0.905**	**0.905**	0.886	**0.905**	0.707

Çizelgelerdeki sonuçlarda görüldüğü gibi algoritmaların sınıflandırma başarıları birbirine oldukça yakındır. Sınıflandırma başarısındaki farklılıkların anlamlı olup olmadığına, parametrik olmayan istatiksel testlerle karar verilebilir. Çizelge 4.3'deki sonuçlar Wilcoxon işaretli sıralar testi ile analiz edilmiştir. Bu testin amacı, aynı veri kümeleri üzerinde farklı sınıflandırma algoritmalarından elde edilen sonuçlar arasında anlamlı bir fark olup olmadığına karar vermektir. Test, iki algoritma arasında gerçekleşir. Kıyaslanacak iki algoritma, aynı veri seti üzerindeki otuz farklı çalışmanın sınıflandırma

başarı değerleri üzerinden test edilir. *P-Value* değerinin 0.05'ten küçük olması iki algoritma arasında anlamlı bir farkın olduğunu söylemektedir. Wilcoxon testinin sonuç tablolarındaki önemli nokta *W* değeridir. Çizelgelerdeki *W* değeri 0 ise kıyaslanan iki algoritmanın sonuçları arasında anlamlı bir fark yoktur. Eğer W değeri 1 ise AAA algoritması daha başarılı, eğer 2 ise kıyaslanan ikinci algoritma daha başarılı olduğu sonucu çıkmaktadır.

Örneğin Australian veri setinde AAA'nın ortalama başarısı 0.831, MFO'nun başarısı 0.832'dir. İki algoritma arasında yapılan Wilcoxon testinde W değeri 0 çıkmıştır. Bunun anlamı iki algoritma arasındaki 0.001'lik değerde anlamlı bir fark yoktur.

Breast cancer veri setinde AAA'nın başarısı 0.976, CS'nin başarısı 0.970'dir. İki algoritma arasında yapılan Wilcoxon testinde W değeri 1 çıkmıştır. Bu sonuç AAA'nın üstünlüğünün anlamlı bir farkı olduğunu göstermektedir.

Parkinsons veri setinde MVO'nun başarısı 0.876, AAA'nın başarısı 0.835'tir. İki algoritma arasında yapılan Wilcoxon testinde W değeri 2 çıkmıştır. Bu sonuç MVO'nun 0.041'lik üstünlüğünün anlamlı olduğu göstermektedir.

Çizelge 4.5'te AAA ile PSO ve AAA ile WOA arasında yapılan Wilcoxon işaretli sıralar testinin sonucu gösterilmektedir.

Çizelge 4.5. AAA-PSO ve AAA-WOA Wilcoxon işaretli sıralar testi

P	*AAA - PSO*			*AAA-WOA*		
	p-Value	*T*	*W*	*p-Value*	*T*	*W*
Australian	8,0999	395	1	1,71E-02	465	1
Blood	0,9860	116	0	1,40E-02	465	1
Breast cancer	5,22E-01	270	1	1,59E-02	465	1
Chess	0,1846	297	0	1,73E-02	465	1
Diabetes	0,6356	255	0	1,71E-02	465	1
Ionosphere	7,0993E-04	374	1	1,7041E-06	465	1
Liver	0,1437	249	0	1,64E-02	465	1
Parkinsons	5,65E+00	290	1	2,79E-02	434	1
Tic-tac-toe	0,9546	205	0	1,7181E-06	465	1
Vertebral	0,1415	250	0	1,62E-02	465	1

Çizelge 4.6'da AAA ile MFO ve AAA ile MVO arasında yapılan Wilcoxon işaretli sıralar testinin sonucu gösterilmektedir.

Çizelge 4.6. AAA-MFO ve AAA-MVO Wilcoxon işaretli sıralar testi

P	*AAA - MFO*			*AAA-MVO*		
	p-Value	*T*	*W*	*p-Value*	*T*	*W*
Australian	0.8006	178	0	3,56E+00	46	2
Blood	0,5252	139	0	0,4983	423	0
Breast cancer	0,8923	157	0	0,7847	140	0
Chess	3,19E+00	57	2	1,73E-02	0	2
Diabetes	0,1547	130	0	0,0327	100	2
Ionosphere	0,0244	113	2	1,0109E-05	18	2
Liver	0,7598	163	0	0,2582	153	0
Parkinsons	0,8786	169	0	0,0037	83	2
Tic-tac-toe	0,0358	130	2	1,7246E-06	0	2
Vertebral	0,2365	129	0	0,0842	98	0

Çizelge 4.7'de AAA ile MFO ve AAA ile MVO arasında yapılan Wilcoxon işaretli sıralar testinin sonucu gösterilmektedir.

Çizelge 4.7. AAA-PSO ve AAA-WOA Wilcoxon işaretli sıralar testi

P	*AAA - BAT*			*AAA-CS*		
	p-Value	*T*	*W*	*p-Value*	*T*	*W*
Australian	3,79E-01	408	1	0,0206	101	2
Blood	1,58E-02	465	1	0,0124	44	2
Breast cancer	1,73E-02	465	1	0,026	314	1
Chess	0,2059	294	0	0,017	80	2
Diabetes	3,85E-02	457	1	0,087	298	1
Ionosphere	2,5986E-06	461	1	0,0046	348	1
Liver	1,16E-01	395	1	8,76E-02	398	1
Parkinsons	1,51E-01	417	1	0,0531	251	0
Tic-tac-toe	0,0544	326	0	0,0327	259	1
Vertebral	7,53E-02	424	1	5,57E-01	334	1

Çizelge 4.8'de AAA ile GWO arasında yapılan Wilcoxon işaretli sıralar testinin sonucu gösterilmektedir.

Çizelge 4.8. AAA-GWO Wilcoxon işaretli sıralar testi

P	*AAA - GWO*		
	p-Value	*T*	*W*
Australian	1,85E-02	464	1
Blood	1,23E-02	465	1
Breast cancer	1,53E-02	465	1
Chess	1,73E-02	465	1
Diabetes	1,68E-02	465	1
Ionosphere	1,6774E-06	465	1
Liver	1,59E-02	465	1
Parkinsons	4,94E-02	403	1
Tic-tac-toe	1,6998E-06	465	1
Vertebral	1,56E-02	465	1

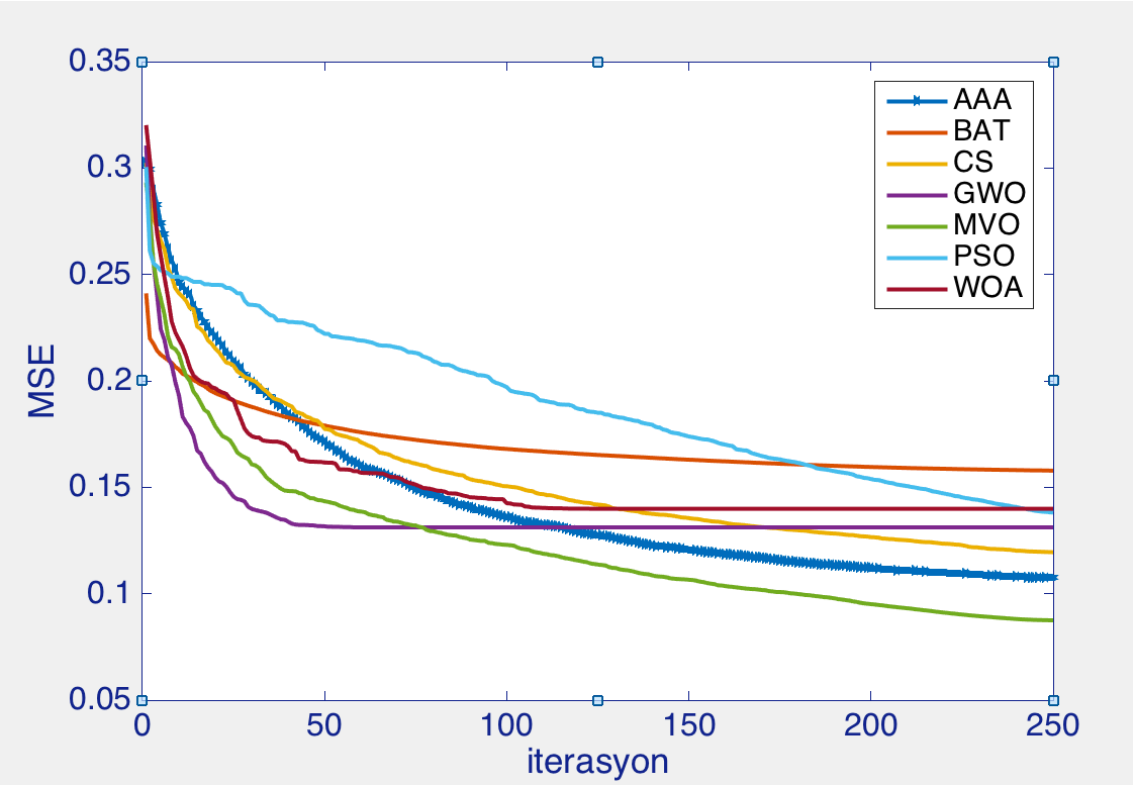
Çizelge 4.9'da algoritmaların veri setlerini eğitmek için harcadıkları sürelerin otuz çalışmadaki ortalaması gösterilmiştir. Bu süreler saniye cinsinden olup i7 3770 CPU, 3.4 Ghz ve 8Gb RAM özelliklerine sahip bir bilgisayarda hesaplanmıştır. Bahsi geçen bilgisayardaki toplam eğitim süresi 6315 dakikadır.

Çizelge 4.9. Algoritmaların eğitim için harcadıkları süreler (saniye)

Veri Seti Algoritma	AAA	PSO	WOA	MFO	MVO	BAT	CS	GWO
Australian	**170**	180	199	171	195	185	374	215
Blood	**185**	186	**185**	**185**	**185**	186	373	186
Breast cancer	**180**	185	183	181	181	188	373	187
Chess	114	125	115	120	113	**101**	200	113
Diabetes	**171**	179	178	176	176	174	350	180
Ionosphere	108	197	157	162	151	**103**	207	244
Liver	77	81	79	**71**	78	68	140	82
Parkinsons	51	89	73	72	95	**46**	93	107
Tic-tac-toe	208	210	210	199	205	**195**	389	208
Vertebral	65	64	67	63	63	**62**	122	67

Australian veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.1'de gösterilmiştir. Australian veri setinde ortalama başarı değeri en yüksek algoritma MVO'dur. Aynı zamanda MVO standart sapma değeri en düşük olan algoritmadır. Otuz çalışmada en yüksek sınıflandırma başarısı CS algoritmasına aittir. Yapılan Wilcoxon testinde MFO'nun AAA'dan 0,001 üstünlüğünde anlamlı bir fark olmadığı sonucu çıkmıştır. AAA'nın ortalama sınıflandırma başarısı PSO, WOA, BAT ve GWO algoritmalarını geride bırakmıştır. Aynı zamanda AAA otuz çalışma içinde en

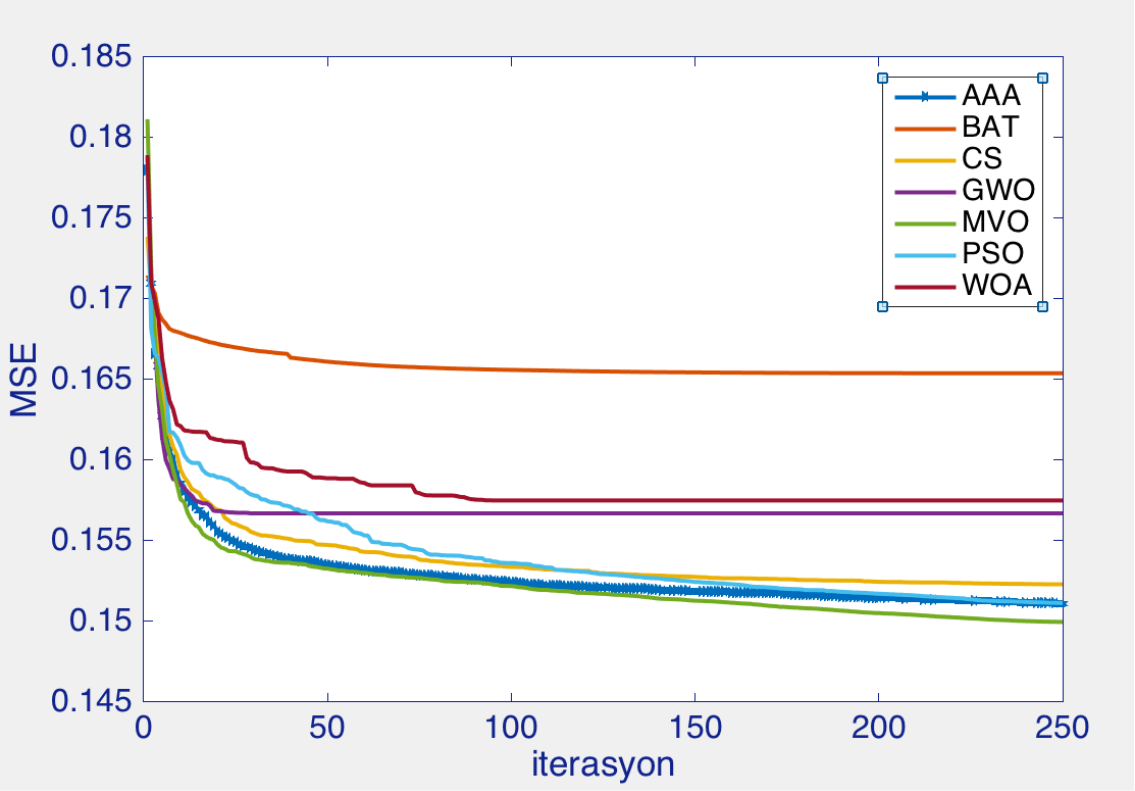
iyi sınıflandırma başarısında PSO, WOA ve MFO algoritmalarından daha iyi performans göstermektedir. Ayrıca eğitim süresi en kısa olan algoritma AAA'dır. AAA, ortalamada en başarılı olan MVO algoritmasından başarı noktasında 0.012 daha geride olduğu halde, 25 saniye daha verimli performans göstermiştir. En iyi yakınsama eğrileri sırasıyla MVO ve AAA'ya aittir. GWO'nun erken yakınsama eğrisi, yerel minimuma takıldığının bir göstergesidir.



Şekil 4.1. Australian veri setine ait yakınsama eğrisi

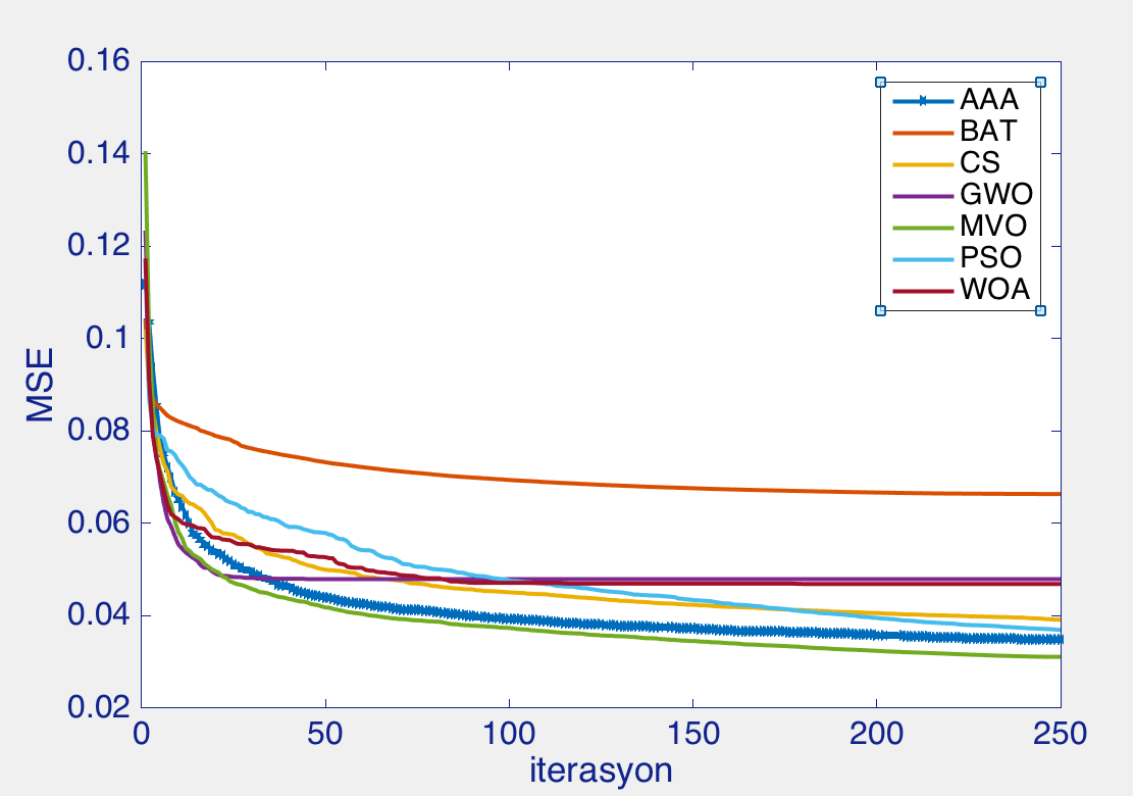
Blood veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.2'de gösterilmiştir. Blood veri setinde ortalamada en başarılı yöntem MFO ve CS gibi gözükse de MFO ve AAA arasında yapılan Wilcoxon testi, aradaki 0.03'lük farkın anlamsız olduğunu göstermektedir. Fakat CS algoritmasının AAA'dan 0.03'lük farklı anlamlıdır. Bununla birlikte AAA'nın standart sapma değeri daha düşüktür. ortalama başarı açısından AAA; WOA, MVO, BAT ve GWO algoritmalarından daha üstün performans göstermektedir. Otuz çalışmadaki en yüksek başarı MFO algoritmasına aittir. Çalıştırma süresi açısından tüm algoritmalar hemen hemen aynıdır. Ancak CS algoritması yaklaşık iki katı kadar fazla sürede çalışmaktadır. MVO, AAA, CS ve PSO'nun yakınsama eğrileri birbirlerine yakın olmakla birlikte MVO algoritmasının yakınsaması

daha iyidir. Bu veri setinin yapısı dolayısıyla yakınsama eğrileri hızlı yakınsama göstermişlerdir.



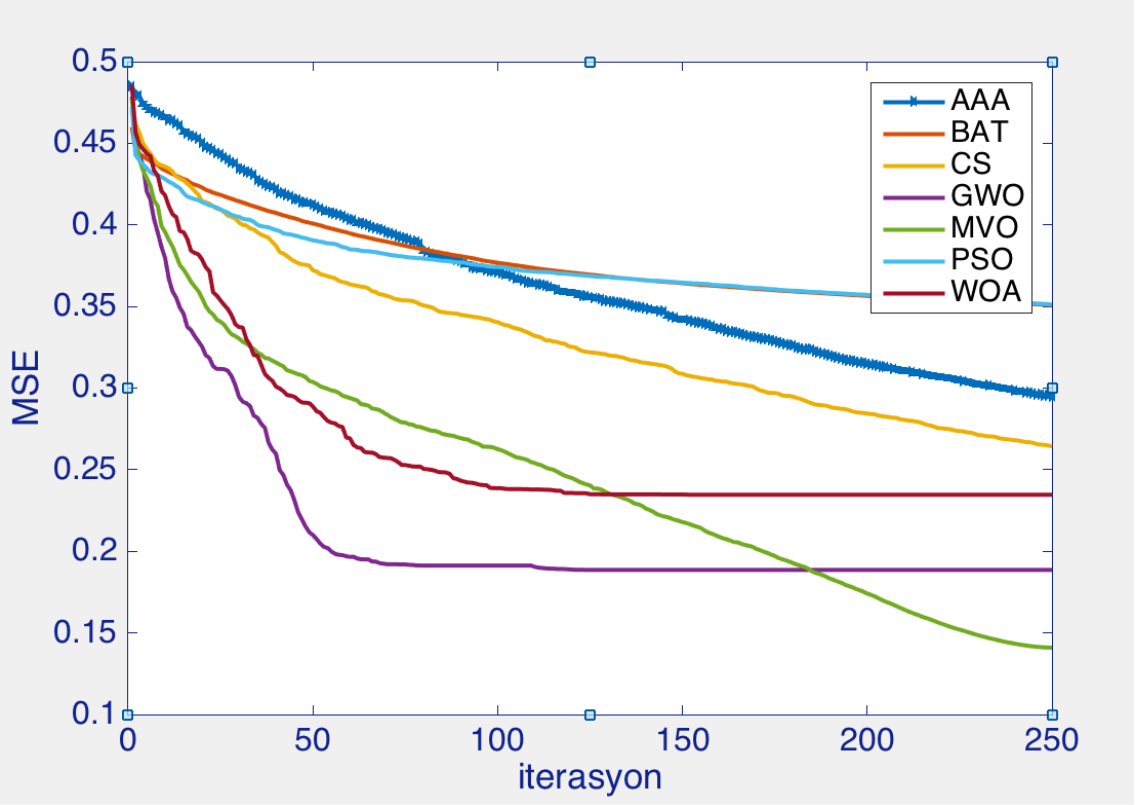
Şekil 4.2. Blood veri setine ait yakınsama eğrisi

Breast cancer veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.3'te gösterilmiştir. Breast cancer veri setinde AAA otuz bağımsız çalıştırmadaki en yüksek başarı değerine sahiptir (0.991). Çalıştırma süresi açısından da AAA algoritmalar içerisinde en kısasıdır. AAA ortalama sınıflandırma başarısında da PSO, WOA, BAT, CS, GWO algoritmalarını geride bırakmıştır. Bu veri setinin yakınsama eğrileri Blood veri setine benzemektedir. En iyi yakınsama eğrileri MVO ve AAA'ya aittir.



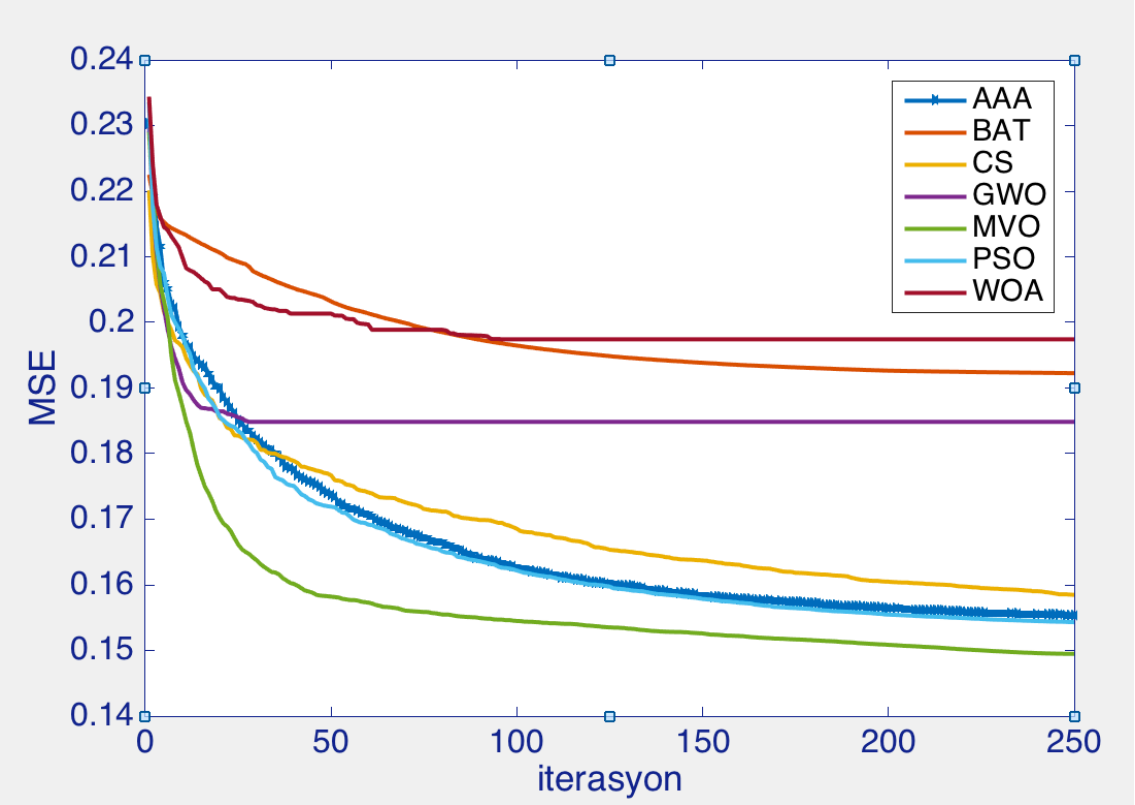
Şekil 4.3. Breast cancer veri setine ait yakınsama eğrisi

Chess veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.4'te gösterilmiştir. Chess veri setinde MVO algoritması ortalama sınıflandırma başarısı, standart sapma ve eğitim süresi açısından en iyi algoritmadır. En yüksek sınıflandırma başarısı baskın şekilde BAT algoritmasına aittir. Bunun yanında AAA ortalamada ve en iyi sınıflandırma başarısında PSO,WOA ve GWO algoritmalarından daha üstün başarı göstermektedir. Başarısı yüksek algoritmaların yakınsama eğrileri birbirine yakındır. Bu veri setinin özellik sayısı fazla olduğu için ağ yapısı daha karmaşıktır. Dolayısıyla algoritmaların veri setini öğrenmesi diğer veri setlerine göre daha uzun sürmektedir. Yakınsama eğrilerinden algoritmaların özellikle AAA'nın daha fazla iterasyonda daha fazla başarı gösterebileceği anlaşılmaktadır.



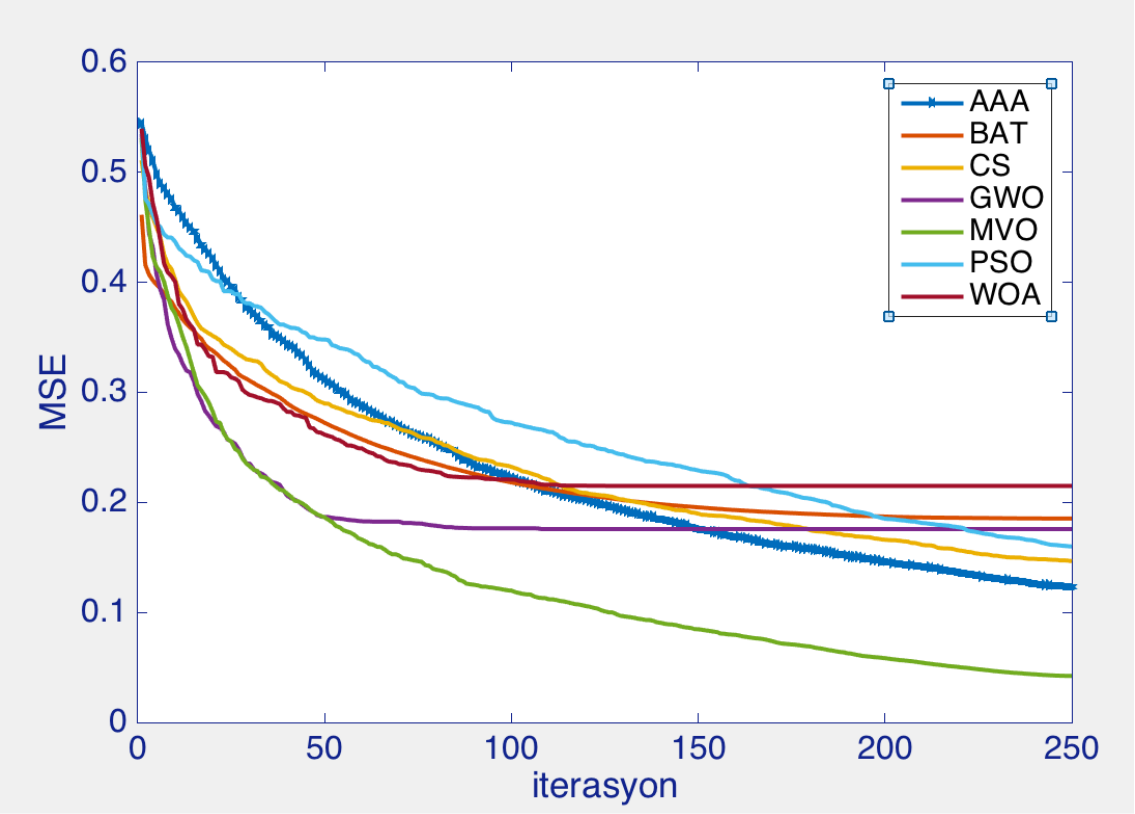
Şekil 4.4. Chess veri setine ait yakınsama eğrisi

Diabetes veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.5'te gösterilmiştir. Diabetes veri setinde ortalama başarısı en yüksek MVO algoritmasıdır. MVO algoritması başarı noktasında AAA'dan 0.05 daha iyi iken, AAA'nın eğitim süresi daha kısadır. AAA ortalamada PSO, WOA, BAT, CS ve GWO algoritmalardan daha üstün başarı göstermektedir. Algoritmaların standart sapma değerleri birbirine oldukça yakındır. En iyi sınıflandırma başarısı PSO ve MFO algoritmalarına ait olmakla birlikte bu algoritmaların başarısı AAA'dan sadece 0.03 daha üstündür. En iyi yakınsama eğrisi MVO'ya aittir. AAA ve PSO'nun eğrileri MVO'ya oldukça yakındır.



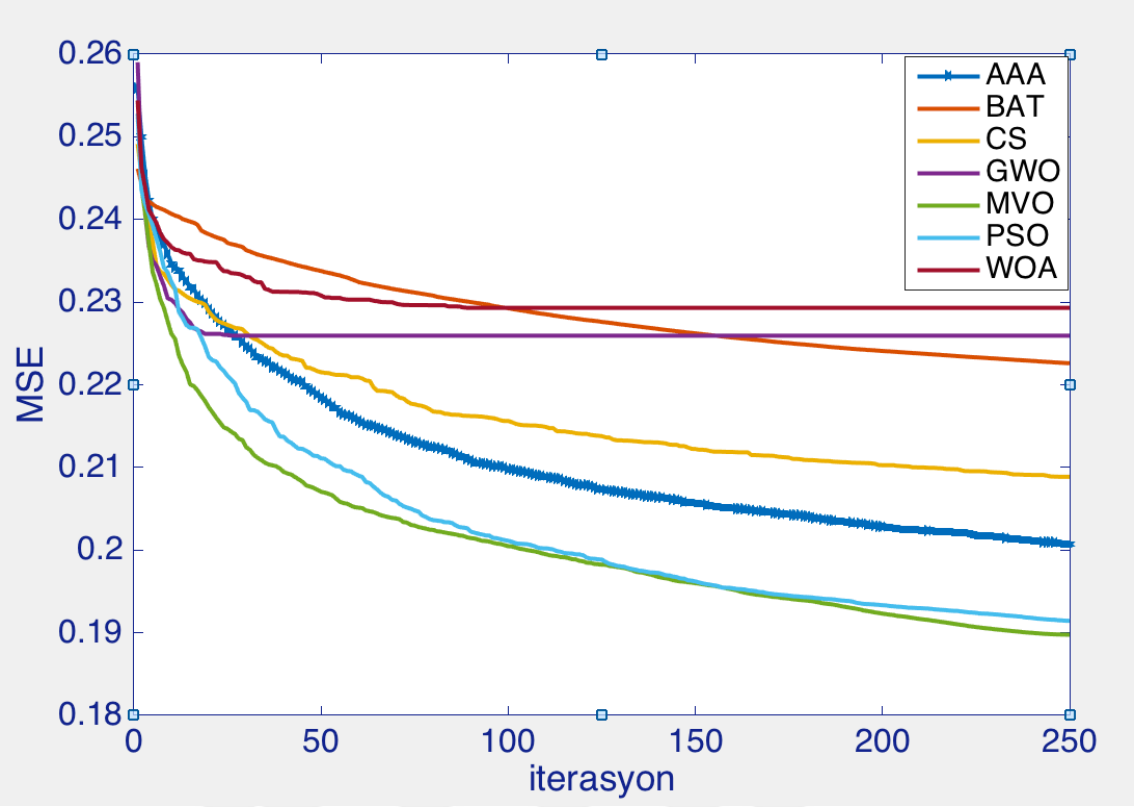
Şekil 4.5. Diabetes veri setine ait yakınsama eğrisi

Ionosphere veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.6'da gösterilmiştir. Ionosphere veri setinde ortalama başarı değeri, en yüksek başarı değeri ve en düşük standart sapma değerine sahip algoritma MVO'dur. AAA, MVO algoritmasından ortalama başarıda 0.05, en iyi başarıda da 0.03 geride kalmaktadır. Yapılan Wilcoxon testi 0.05'lik farkın anlamlı olduğunu göstermektedir. Fakat MVO'nun eğitim süresi 151 saniye, AAA'nın 108 saniyedir. AAA ortalama başarıda PSO, WOA, BAT, CS, GWO algoritmalarından daha iyi başarı göstermektedir. İlk iterasyonlarda GWO'nun yakınsama eğrisi başarılı görünse de 50.iterasyondan sonra yerel bir minimuma takılarak hatasını azaltamamıştır.



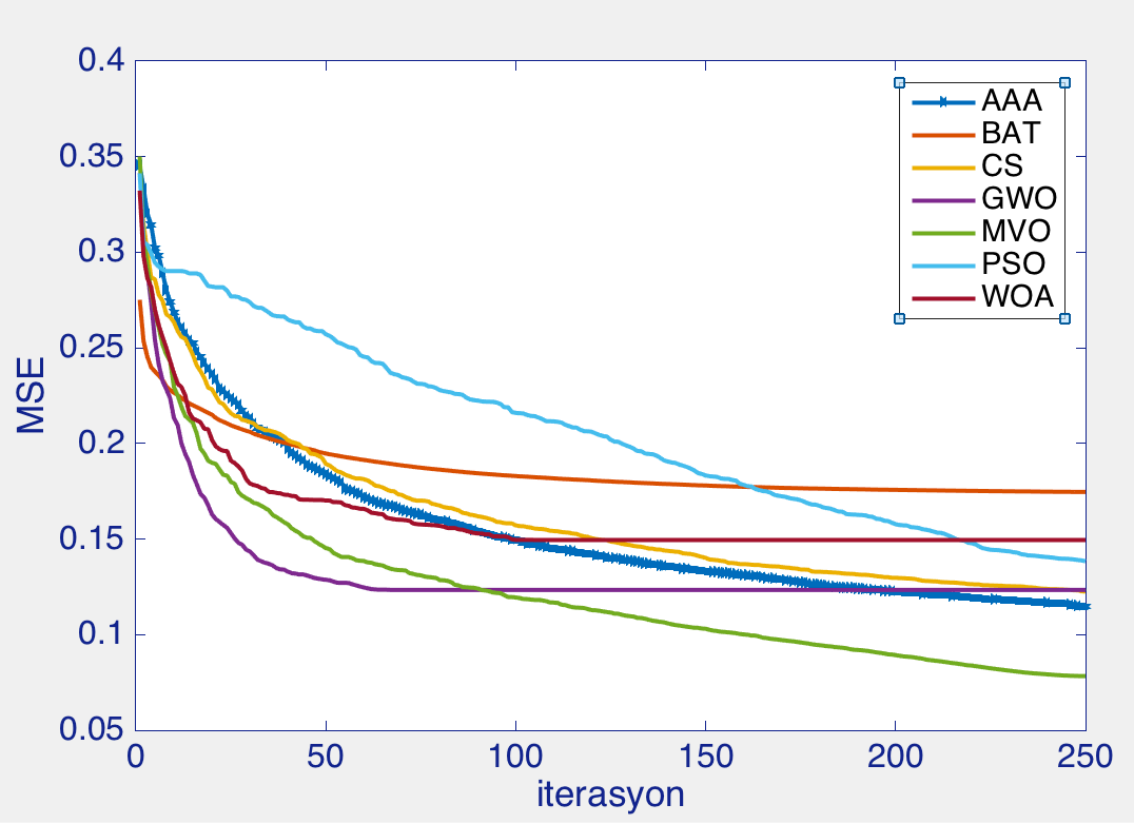
Şekil 4.6. Ionosphere veri setine ait yakınsama eğrisi

Liver veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.7'de gösterilmiştir. Liver veri setinde bağımsız otuz çalışmada tüm algoritmalar içerisinde en yüksek başarı AAA'ya aittir. AAA ortalama başarıda da PSO, WOA, BAT, CS ve GWO algoritmalarından daha üstün performans göstermektedir. En iyi yakınsama eğrisi PSO ve MVO algoritmalarına aittir. Bununla birlikte AAA'nın daha fazla iterasyonda daha fazla hatayı azaltabileceği yakınsama eğrisinden anlaşılmaktadır.



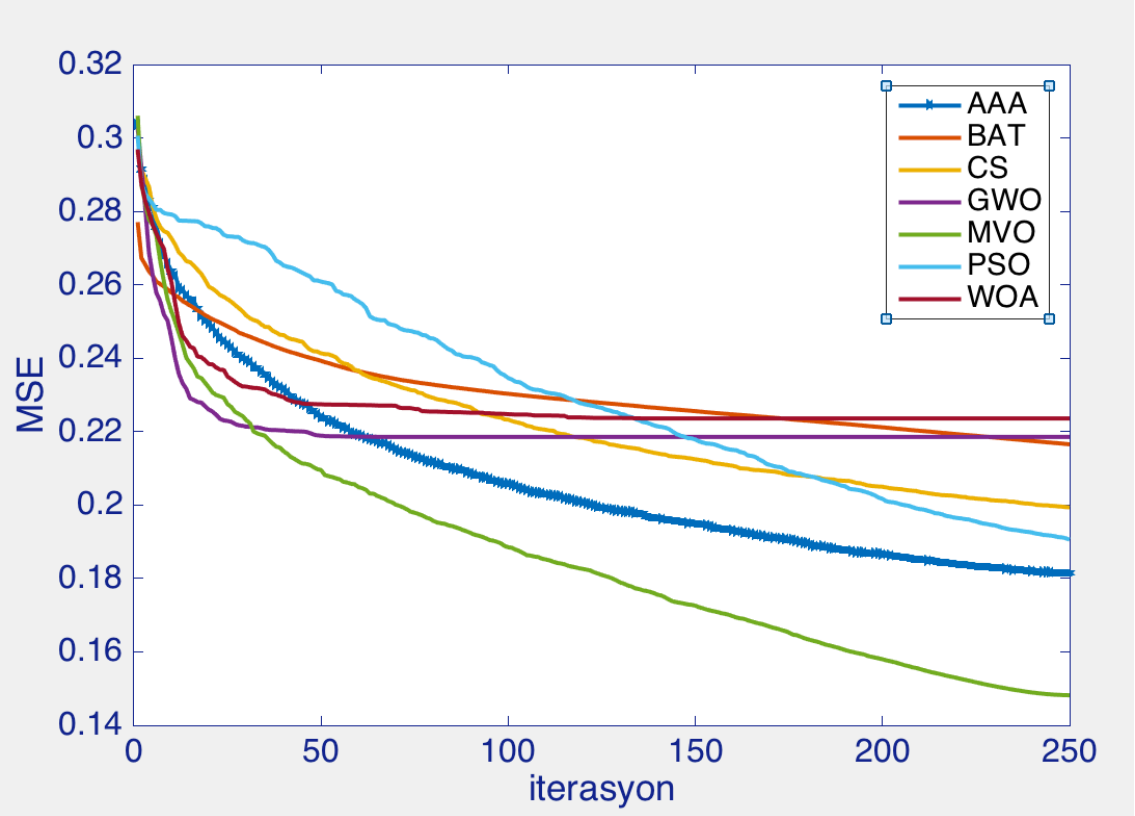
Şekil 4.7. Liver veri setine ait yakınsama eğrisi

Parkinsons veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.8'de gösterilmiştir. Parkinsons veri setinde bağımsız otuz çalışmadaki en yüksek başarıyı AAA ve MVO algoritması göstermiştir. Ortalama sınıflandırma başarısında, AAA 0.03 farkla MVO'nun gerisinde kalmaktadır. Bu fark Wilcoxon testine göre anlamlıdır. Ancak eğitim için AAA'nın harcadığı süre 51 saniye iken MVO'da 95 saniyedir. GWO algoritmasının erken ve hızlı yakınsaması yerel minimumlara takılmasına neden olmuştur.



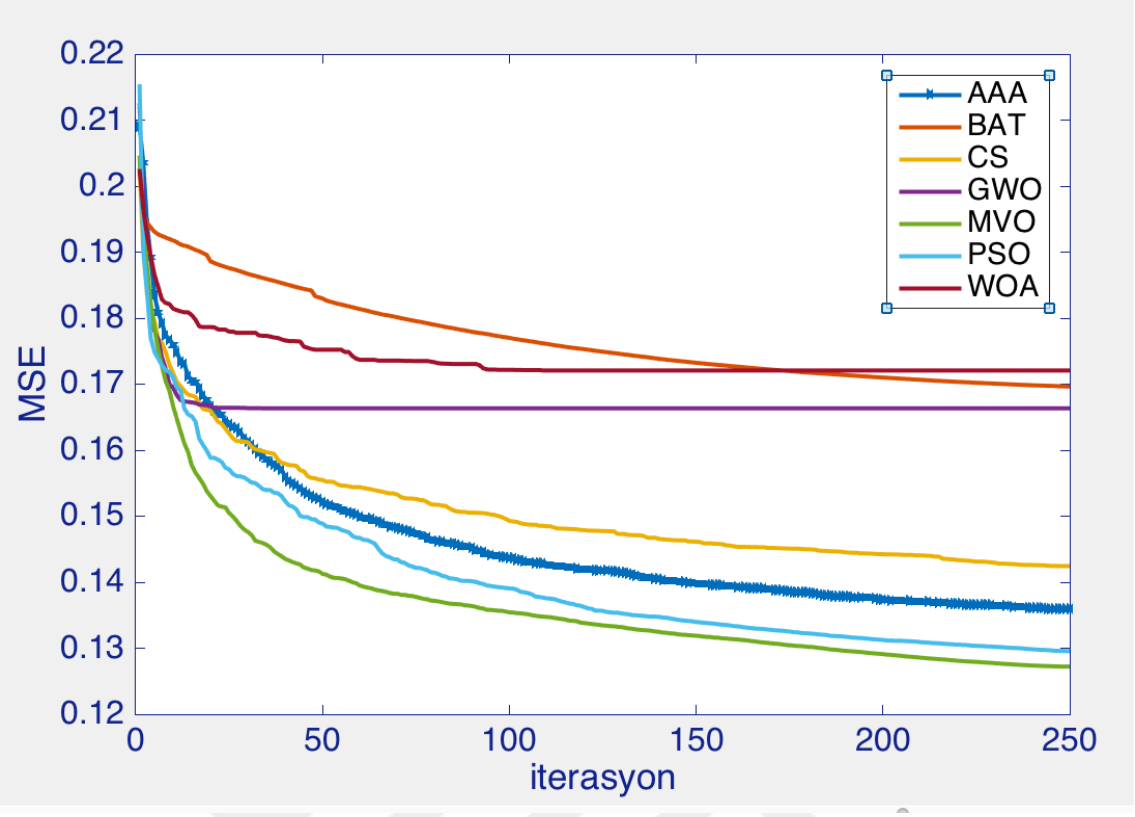
Şekil 4.8. Parkinsons veri setine ait yakınsama eğrisi

Tic-tac-toe veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.9'da gösterilmiştir. Tic-Tac-Toe veri setinde otuz bağımsız çalışmadaki ortalama ve en yüksek başarıyı MVO algoritması göstermiştir. AAA, ortalamada başarı sınıflandırmasında PSO, WOA, BAT ve GWO algoritmalarında daha üstündür. En kısa çalışma süresi BAT algoritmasına aittir. En başarılı yakınsama eğrisi MVO algoritmasına aittir. AAA'nın yakınsama eğrisi WOA, BAT, CS, GWO ve PSO'dan daha iyidir.



Şekil 4.9. Tic-tac-toe veri setine ait yakınsama eğrisi

Vertebral veri setine ait bağımsız otuz çalışmanın ortalama yakınsama eğrisi Şekil 4.10'da gösterilmiştir. Vertebral veri setinde bağımsız otuz çalışmadaki en yüksek başarıyı AAA, PSO, MVO, MFO ve CS algoritmaları göstermiştir. MFO ve MVO algoritmasının ortalama sınıflandırma başarısı diğer algoritmalardan daha üstündür. En düşük standart sapma değerine MVO algoritması sahiptir. Algoritmaların çalışma süreleri birbirine oldukça yakındır. En başarılı yakınsama eğrisine sırasıyla MVO,PSO,AAA,CS algoritmaları sahiptir.



Şekil 4.10. Vertebral veri setine ait yakınsama eğrisi

Yakınsama eğrileri algoritmaların veri setlerindeki MSE değerini iterasyonlar boyunca nasıl azalttığını göstermektedir. Algoritmanın MSE değerini, az sayıda iterasyonda azaltması daha iyidir. Sömürü yeteneği fazla olan algoritmaların yakınsaması daha erkendir. Fakat yerel minimumlara takılma olasılığı daha yüksek, yerel arama yeteneği daha fazladır. Dolayısıyla erken yakınsaması fazla olan algoritmalar, hatayı istenen değere kadar azaltamayabilir. Keşif yeteneği fazla olan algoritmalarda ise yakınsama kısmen daha yavaştır. Bununla birlikte yerel minimumlara takılma olasılığı daha düşük, global arama yeteneği daha fazladır.

AAA'nın bağımsız otuz çalışmadaki standart sapma değerinin düşük olması, başlangıç parametrelerinden bağımsız çalıştığının ve yerel minimumlardan kaçındığının göstergesidir. AAA; Breast cancer, Liver, Parkinsons ve Vertebral veri setlerinde otuz çalışmada en iyi sınıflandırma başarısını göstermiştir. Australian, Blood, Diabetes ve Ionosphere veri setlerinde de en iyi algoritmalara oldukça yakın sonuçlar vermektedir. Ayrıca Australian, Blood, Breast cancer ve Diabetes veri setlerinde en kısa eğitim süresine AAA sahiptir.

Parkinsons ve Ionosphere veri setlerinde özellik sayısı fazla olmasından ötürü MLP'deki parametre sayısı daha fazladır. Bu veri setlerinde MLP yapısı büyük olduğu için eğitim süresi uzun olması gerekir. Fakat eğitim setindeki örnek sayısı az olduğundan eğitim süresi kısadır. Aynı şekilde Vertebral ve Liver veri setlerinde, algoritmaların eğitim süresi diğer veri setlerine göre daha kısa olmasının nedeni, eğitim setindeki örnek sayısının az olmasıdır.

BAT algoritması, veri setlerinin genelinde en kısa eğitim süresine sahiptir. Fakat sınıflandırma başarısı bakımından üstün sıradaki algoritmaların içerisinde en verimli eğitim süresi AAA'ya aittir. CS, tüm veri setlerinde en fazla çalıştırma süresine sahip olan algoritmadır.

GWO algoritmasının, otuz bağımsız çalışmada birbirine çok yakın sonuçlar bulması dolayısıyla standart sapma değeri sıfıra çok yakındır. Ancak bu değerlerin global optimum olmayışı, yerel optimumlara takıldığının bir göstergesidir.

AAA'nın yüksek sınıflandırma başarısı ve düşük standart sapması, yerel optimumlara takılmayı güvenilir şekilde önleyebildiğini göstermektedir. Aynı zamanda MLP'nin ağırlık ve eşik değerleri için optimum çözümleri bulabildiğini kanıtlamaktadır. Yapılan Wilcoxon testleri, AAA'nın küçük farkla geride kaldığı algoritmalara kıyasla, aradaki farkın anlamsız olduğunu göstermektedir. Bu durumda AAA'nın eğitim sürenin kısa olması büyük bir avantajıdır.

GWO, BAT ve WOA algoritmalarının yakınsama eğrileri ilk iterasyonlarda başarılı görünse de iterasyon sayısı ilerledikçe bu algoritmalar hatayı yeterince azaltamamaktadır. Bu çalışmada 250 iterasyon yapılmıştır. AAA'nın iterasyonlar boyunca hatasını devamlı olarak düşürdüğü göz önünde bulundurulduğunda iterasyon sayısının artması AAA'nın başarısını artıracaktır.

5. SONUÇLAR VE ÖNERİLER

5.1 Sonuçlar

ANN'nin eğitim süreci uygun ağ parametrelerinin bulunması işlemi olması dolayısıyla bir optimizasyon problemidir. Bu problemi çözmek için literatürde çeşitli stratejiler bulunmaktadır. Bunlardan birisi matematiksel çözüm stratejileridir. Matematiksel çözüm stratejileri, problemin zorluğu, karmaşıklığı ve kaynak kısıtlılığının artması durumunda çözümün bulunma süresini artırarak çözümü zorlaştırmaktadır. Hatta türevlenemeyen problemleri, matematiksel stratejiler çözememektedir. Bu tür problemlerin üstesinden gelmek için literatürde sezgisel stratejiler önerilmiştir. Sezgisel stratejiler kesin çözüm garantisi vermemekle birlikte makul sürede iyi sonuçlar göstermektedir.

Literatürde ANN eğitimi için bir çok matematiksel ve sezgisel strateji önerilmiştir. Ancak mevcut algoritmalar halen yerel minimumlara takılma eğilimini ortadan kaldırabilmiş değillerdir. Bu tez çalışmasında yakın zamanda önerilen yapay alg algoritması, ANN eğitimi için önerilmiştir. Yapay alg algoritması alglerin karakteristik özelliklerinden ve yaşam davranışlarından ilham alan bir optimizasyon algoritmasıdır. AAA ile literatürdeki yapılan farklı çalışmalar, AAA'nın globalde ve yereldeki güçlü arama yeteneğini göstermiştir. Bu çalışmalar AAA'nın ANN eğitiminde de başarı gösterebileceğine bir işarettir.

Önerilen yöntemin başarısı farklı zorluk seviyelerine sahip on veri seti üzerinde doğrulanmıştır. Sonuçlar yedi farklı popülasyon tabanlı optimizasyon algoritması ile kıyaslanmıştır.

AAA'nın otuz bağımsız çalışmadaki standart sapma değeri oldukça düşük olması ve sınıflandırma başarısının yüksek olması, başlangıç parametrelerinden bağımsız çalıştığının ve yerel minimumlardan kaçındığının bir göstergesidir. AAA'nın yakınsama eğrisinin veri setlerinde tutarlı olması önerilen yönteme olan güveni artırmaktadır. Ayrıca AAA'nın diğer algoritmalara kıyasla eğitim süresi oldukça kısadır. AAA bir çok veri setinde en iyi sınıflandırma başarısını göstermiş, diğer veri setlerinde de en iyi sonuçlara oldukça yakındır.

Bu tez çalışmasında AAA'nın, ANN eğitiminde üstün performans gösteren güvenilir bir alternatif olduğu kanıtlanmıştır.

5.2 Öneriler

Bu tez çalışması temel AAA üzerinden gerçekleştirilmiştir. Çalışmaların devamında AAA'da geliştirmeler yapılarak başarısı artırılabilir. ANN'nin mimari yapısının tasarımında ve ANFIS parametrelerin optimizasyonu için AAA kullanılabilir.

Gittikçe popülaritesi artan Derin Öğrenme'de özellikle Evrişimsel Sinir Ağları'nın mimari yapısının tasarımında ve uygun ağ parametrelerinin optimizasyonda AAA kullanılabilir.

KAYNAKLAR

Aljarah, I., Faris, H. ve Mirjalili, S., 2018, Optimizing connection weights in neural networks using the whale optimization algorithm, *Soft Computing*, 22 (1), 1-15.

Asuncion, A. ve Newman, D., 2007, UCI machine learning repository.

Basheer, I. A. ve Hajmeer, M., 2000, Artificial neural networks: fundamentals, computing, design, and application, *Journal of microbiological methods*, 43 (1), 3-31.

Blum, C. ve Socha, K., 2005, Training feed-forward neural networks with ant colony optimization: An application to pattern classification, *Hybrid Intelligent Systems, 2005. HIS'05. Fifth International Conference on*, 6 pp.

Caruana, R. ve Niculescu-Mizil, A., 2006, An empirical comparison of supervised learning algorithms, *Proceedings of the 23rd international conference on Machine learning*, 161-168.

Črepinšek, M., Liu, S.-H. ve Mernik, M., 2013, Exploration and exploitation in evolutionary algorithms: A survey, *ACM Computing Surveys (CSUR)*, 45 (3), 35.

Ding, S., Su, C. ve Yu, J., 2011, An optimizing BP neural network algorithm based on genetic algorithm, *Artificial Intelligence Review*, 36 (2), 153-162.

Dorffner, G., 1996, Neural networks for time series processing, *Neural network world*.

Ebrahimi, A. ve Khamehchi, E., 2016, Sperm whale algorithm: An effective metaheuristic algorithm for production optimization problems, *Journal of Natural Gas Science and Engineering*, 29, 211-222.

Efe, M. O. ve Kaynak, O., 1999, A comparative study of neural network structures in identification of nonlinear systems, *Mechatronics*, 9 (3), 287-300.

Faris, H., Aljarah, I., Mirjalili, S., Castillo, P. A. ve Merelo, J. J., 2016, EvoloPy: An Open-source Nature-inspired Optimization Framework in Python, *IJCCI (ECTA)*, 171-177.

Gudise, V. G. ve Venayagamoorthy, G. K., 2003, Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks, *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, 110-117.

Gupta, J. N. ve Sexton, R. S., 1999, Comparing backpropagation with a genetic algorithm for neural network training, *Omega*, 27 (6), 679-684.

Hertz, J. A., 2018, Introduction to the theory of neural computation, CRC Press, p.

Hinton, G. E. ve Sejnowski, T. J., 1999, Unsupervised learning: foundations of neural computation, MIT press, p.

Ho, Y.-C. ve Pepyne, D. L., 2002, Simple explanation of the no-free-lunch theorem and its implications, *Journal of optimization theory and applications*, 115 (3), 549-570.

Holland, J. ve Goldberg, D., 1989, Genetic algorithms in search, optimization and machine learning, *Massachusetts: Addison-Wesley*.

Holland, J. H., 1992, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, MIT press, p.

Ilonen, J., Kamarainen, J.-K. ve Lampinen, J., 2003, Differential evolution training algorithm for feed-forward neural networks, *Neural Processing Letters*, 17 (1), 93-105.

Karaboga, D., 2005, An idea based on honey bee swarm for numerical optimization, *Technical report-tr06, Erciyes university, engineering faculty, computer engineering department*.

Karaboga, D., Akay, B. ve Ozturk, C., 2007, Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks, *International conference on modeling decisions for artificial intelligence*, 318-329.

Karaboğa, D., 2014, Yapay Zeka Optimizasyon Algoritmalari.

Kennedy, J., 2011, Particle swarm optimization, In: Encyclopedia of machine learning, Eds: Springer, p. 760-766.

Kim, J. ve Jung, S., 2015, Implementation of the RBF neural chip with the back-propagation algorithm for on-line learning, *Applied Soft Computing*, 29, 233-244.

Kohavi, R., 1995, A study of cross-validation and bootstrap for accuracy estimation and model selection, *Ijcai*, 1137-1145.

Kohonen, T., 1990, The self-organizing map, *Proceedings of the IEEE*, 78 (9), 1464-1480.

Krizhevsky, A., Sutskever, I. ve Hinton, G. E., 2012, Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems*, 1097-1105.

Linggard, R., Myers, D. ve Nightingale, C., 2012, Neural networks for vision, speech and natural language, Springer Science & Business Media, p.

Mendes, R., Cortez, P., Rocha, M. ve Neves, J., 2002, Particle swarms for feedforward neural network training, *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, 1895-1899.

Mirjalili, S., Hashim, S. Z. M. ve Sardroudi, H. M., 2012, Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm, *Applied Mathematics and Computation*, 218 (22), 11125-11137.

Mirjalili, S., Mirjalili, S. M. ve Lewis, A., 2014a, Grey wolf optimizer, *Advances in engineering software*, 69, 46-61.

Mirjalili, S., Mirjalili, S. M. ve Lewis, A., 2014b, Let a biogeography-based optimizer train your multi-layer perceptron, *Information Sciences*, 269, 188-209.

Mirjalili, S., 2015, How effective is the Grey Wolf optimizer in training multi-layer perceptrons, *Applied Intelligence*, 43 (1), 150-161.

Mirjalili, S. ve Lewis, A., 2016, The whale optimization algorithm, *Advances in engineering software*, 95, 51-67.

Mitchell, M., Holland, J. H. ve Forrest, S., 1994, When will a genetic algorithm outperform hill climbing, *Advances in neural information processing systems*, 51-58.

Mitchell, M., 1998, An introduction to genetic algorithms, MIT press, p.

Ozturk, C. ve Karaboga, D., 2011, Hybrid artificial bee colony algorithm for neural network training, *Evolutionary Computation (CEC), 2011 IEEE Congress on*, 84-88.

Panchal, G., Ganatra, A., Kosta, Y. ve Panchal, D., 2011, Behaviour analysis of multilayer perceptronswith multiple hidden neurons and hidden layers, *International Journal of Computer Theory and Engineering*, 3 (2), 332.

Park, J. ve Sandberg, I. W., 1993, Approximation and radial-basis-function networks, *Neural computation*, 5 (2), 305-316.

Rakitianskaia, A. S. ve Engelbrecht, A. P., 2012, Training feedforward neural networks with dynamic particle swarm optimisation, *Swarm Intelligence*, 6 (3), 233-270.

Reed, R. ve Marks, R. J., 1999, Neural smithing: supervised learning in feedforward artificial neural networks, Mit Press, p.

Rezaeianzadeh, M., Tabari, H., Yazdi, A. A., Isik, S. ve Kalin, L., 2014, Flood flow forecasting using ANN, ANFIS and regression models, *Neural Computing and Applications*, 25 (1), 25-37.

Schmidhuber, J., 2015, Deep learning in neural networks: An overview, *Neural networks*, 61, 85-117.

Seiffert, U., 2001, Multiple layer perceptron training using genetic algorithms, *ESANN*, 159-164.

Sexton, R. S. ve Gupta, J. N., 2000, Comparative evaluation of genetic algorithm and backpropagation for training neural networks, *Information Sciences*, 129 (1-4), 45-59.

Slowik, A. ve Bialko, M., 2008, Training of artificial neural networks using differential evolution algorithm, *Human System Interactions, 2008 Conference on*, 60-65.

Socha, K. ve Blum, C., 2007, An ant colony optimization algorithm for continuous optimization: application to feed-forward neural network training, *Neural Computing and Applications*, 16 (3), 235-247.

Storn, R. ve Price, K., 1997, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization*, 11 (4), 341-359.

Sutton, R. S. ve Barto, A. G., 1998, Reinforcement learning: An introduction, 1, MIT press Cambridge, p.

Uymaz, S. A., 2015, Yeni bir biyolojik ilhamlı metasezgisel optimizasyon metodu: Yapay alg algoritması, *Selçuk Üniversitesi Fen Bilimleri Enstitüsü*.

Uymaz, S. A., Tezel, G. ve Yel, E., 2015, Artificial algae algorithm (AAA) for nonlinear global optimization, *Applied Soft Computing*, 31, 153-171.

Van Laarhoven, P. ve Aarts, E., Simulated annealing. 1987, Springer.

Wang, D., 2001, Unsupervised learning: foundations of neural computation, *AI Magazine*, 22 (2), 101.

Wang, G.-G., Gandomi, A. H., Alavi, A. H. ve Hao, G.-S., 2014, Hybrid krill herd algorithm with differential evolution for global numerical optimization, *Neural Computing and Applications*, 25 (2), 297-308.

Wang, L., Zeng, Y. ve Chen, T., 2015, Back propagation neural network with adaptive differential evolution algorithm for time series forecasting, *Expert Systems with Applications*, 42 (2), 855-863.

Wdaa, A. S. I. ve Sttar, A., 2008, Differential evolution for neural networks learning enhancement, *Universiti Teknologi Malaysia*.

Whitley, D., Starkweather, T. ve Bogart, C., 1990, Genetic algorithms and neural networks: Optimizing connections and connectivity, *Parallel computing*, 14 (3), 347-361.

Wolpert, D. H. ve Macready, W. G., 1997, No free lunch theorems for optimization, *IEEE transactions on evolutionary computation*, 1 (1), 67-82.

Xu, G., 2013, An adaptive parameter tuning of particle swarm optimization algorithm, *Applied Mathematics and Computation*, 219 (9), 4560-4569.

Yang, X.-S., 2010, Nature-inspired metaheuristic algorithms, Luniver press, p.

Yegnanarayana, B., 2009, Artificial neural networks, PHI Learning Pvt. Ltd., p.

Yu, J., Wang, S. ve Xi, L., 2008, Evolving artificial neural networks using an improved PSO and DPSO, *Neurocomputing*, 71 (4-6), 1054-1060.

Zhang, J.-R., Zhang, J., Lok, T.-M. ve Lyu, M. R., 2007, A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, *Applied Mathematics and Computation*, 185 (2), 1026-1037.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Bahaeddin TÜRKOĞLU
Uyruğu	**:**	T.C
Doğum Yeri ve Tarihi	**:**	ANKARA 14.12.1994
Telefon	**:**	537 387 48 88
e-mail	**:**	bahaeddinturkoglu@gmail.com

EĞİTİM

Derece		**Adı, İl**	**Bitirme Yılı**
Lise	:	Mustafa Azmi Doğan Anadolu Lisesi, ANKARA	2012
Üniversite	:	Selçuk Üniversitesi, KONYA	2016

UZMANLIK ALANI

Makine Öğrenmesi, Optimizasyon

YABANCI DİLLER

İnglizce