**T.C.**
**ERCİYES ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**
**MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALI**

# NESNELERİN İNTERNETİNDE VERİ ANALİZİ İÇİN TEKRARLAYICI SİNİR AĞLARI YÖNTEMİNİN YAPAY ARI KOLONİ ALGORİTMASI İLE EĞİTİLMESİ

**Hazırlayan**
**Ebru YÖNEM**

**Danışman**
**Dr. Öğr. Üyesi Rüştü AKAY**

**Yüksek Lisans Tezi**

**Haziran 2019**
**Kayseri**

**T.C.**
**ERCİYES ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**
**MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALI**

# NESNELERİN İNTERNETİNDE VERİ ANALİZİ İÇİN TEKRARLAYICI SİNİR AĞLARI YÖNTEMİNİN YAPAY ARI KOLONİ ALGORİTMASI İLE EĞİTİLMESİ

**(Yüksek Lisans Tezi)**

**Hazırlayan**
**Ebru YÖNEM**

**Danışman**
**Dr. Öğr. Üyesi Rüştü AKAY**

**Bu çalışma, Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FYL-2017-7662 kodlu proje ile desteklenmiştir.**

**Haziran 2019**
**Kayseri**

# BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.

Ebru YÖNEM



# YÖNERGEYE UYGUNLUK

**"Nesnelerin İnternetinde Veri Analizi İçin Tekrarlayıcı Sinir Ağları Yönteminin Yapay Arı Koloni Algoritması ile Eğitilmesi"** adlı Yüksek Lisans Tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi' ne uygun olarak hazırlanmıştır.

**Hazırlayan**
Ebru YÖNEM

**Danışman**
Dr. Öğr. Üyesi Rüştü AKAY

**Mekatronik Mühendisliği ABD Başkanı**

Prof. Dr. Şahin YILDIRIM

**Dr. Öğr. Üyesi Rüştü AKAY** danışmanlığında **Ebru YÖNEM** tarafından hazırlanan **"Nesnelerin İnternetinde Veri Analizi İçin Tekrarlayıcı Sinir Ağları Yönteminin Yapay Arı Koloni Algoritması ile Eğitilmesi"** adlı bu çalışma jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü **Mekatronik Mühendisliği** Anabilim Dalında **Yüksek Lisans** tezi olarak kabul edilmiştir.

28/06/2019

**JÜRİ:**

Danışman : Dr. Öğr. Üyesi Rüştü AKAY .....

Üye : Prof. Dr. Şahin YILDIRIM .....

Üye : Dr. Öğr. Üyesi Ebubekir KAYA .....

ONAY:

Bu tezin kabulü Enstitü Yönetim Kurulunun 23/07/2019 tarih ve 2019/42-11 sayılı kararı ile onaylanmıştır.

23/07/2019

Prof. Dr. Mehmet AKKURT

Enstitü Müdürü

# TEŞEKKÜR

Bana çalışmalarım süresince her türlü yardımı ve fedakârlığı sağlayan, tez konumun seçiminden, çalışmalarımın sonuçlandırılmasına kadar her aşamasında bilgi birikimleri ve bilimsel katkılarıyla beni aydınlatan, yakın ilgi ve yardımlarını esirgemeyen, en büyük katkı sahibi değerli hocam Dr. Öğr. Üyesi Rüştü AKAY'a, bu tez çalışmasına maddi destek veren Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi'ne (Proje No: FYL-2017-7662) teşekkürlerimi sunarım.

Ayrıca her zaman olduğu gibi, tezim süresince de bana hep destek olan babam Cemil Yönem'e, manevi desteğini her zaman hissettiğim annem Zekiye Yönem'e ve tezimin sonuçlandırılmasında yanımda olamasa da başından beri beni motive eden bundan sonra da desteğini hep hissedeceğim anneannem Saadet Palamutoğlu'na teşekkür ederim.

Ebru YÖNEM

Haziran 2019, KAYSERİ

# NESNELERİN İNTERNETİNDE VERİ ANALİZİ İÇİN TEKRARLAYICI SİNİR AĞLARI YÖNTEMİNİN YAPAY ARI KOLONİ ALGORİTMASI İLE EĞİTİLMESİ

**Ebru YÖNEM**

**Erciyes Üniversitesi, Fen Bilimleri Enstitüsü**
**Yüksek Lisans Tezi, Haziran 2019**
**Danışman: Dr. Öğr. Üyesi Rüştü AKAY**

## ÖZET

Nesnelerin interneti, haberleşme protokolleri aracılığıyla nesnelerin birbirleri ile iletişimi sonucu oluşturdukları ağlardır. Sanal dünya ile 7fiziksel dünya arasındaki bağlantıyı sağlayan bu yapı çok büyük boyutlarda veri üretmektedir. Bu verilerden anlamlı sonuçlar çıkarabilmek ancak etkili veri analiz yöntemleriyle mümkün olmaktadır. Veri analizi için kullanılan yöntemlerde temel yaklaşım geleceğin, geçmişin bir yansıması olduğu düşüncesidir. Bu nedenle özellikle zaman serisi biçiminde elde edilen nesnelerin interneti verileri için, örneklere bağlı olarak dinamik öğrenme yetenekleri sayesinde, doğrusal ve doğrusal olmayan ilişkileri çözümleyerek başarılı sonuçlar üreten yapay sinir ağları doğru bir veri analiz yöntemi olarak karşımıza çıkmaktadır. Yapay sinir ağlarında istenen düzeyde performansın sağlanması birçok parametreye bağlı olmakla birlikte ağın eğitiminde kullanılan algoritmalar üzerinde en çok çalışılan konulardandır. Gradyan düşüm, eşlenik gradyan düşüm, esnek geri yayılım, Quasi-Newton ve Levenberg-Marquart algoritmaları sıklıkla kullanılan klasik eğitim algoritmalarıdır. Bu klasik algoritmaların yanında bazı sezgisel yöntemler ağın performansını artırmak için kullanılabilmektedir. Bu amaçla tez kapsamında sezgisel yöntemlerden, arıların yiyecek arama davranışlarını modelleyerek geliştirilen yapay arı kolonisi algoritması zaman serileri için başarılı sonuçlar üreten tekrarlayıcı sinir ağlarının eğitiminde kullanılmıştır. Gerçekleştirilen model örnek bazı verilerinin analizinde kullanılmıştır. Bu çalışma ile yapay arı koloni algoritmasının nesnelerin interneti verilerinin analizinde etkin bir şekilde kullanılabileceği gösterilmiştir.

**Anahtar Kelimeler:** Nesnelerin interneti; veri analizi; tekrarlayıcı yapay sinir ağları; yapay arı koloni algoritması

**TRAINING RECURRENT NEURAL NETWORKS METHOD WITH ARTIFICIAL BEE COLONY ALGORITHM FOR DATA ANALYTICS IN INTERNET OF THINGS**

**Ebru YÖNEM**

**Erciyes University, Graduate School of Natural and Applied Sciences**
**Master Thesis, June 2019**
**Supervisor: Asst. Prof. Rüştü AKAY**

# ABSTRACT

Internet of Things is the network that objects can communicate with each other through communication protocols. This network, which provides the connection between the virtual worlds to the physical world, produces enormous amounts of data. To obtain meaningful results from the data is only possible through effective data analysis methods. In all methods used for data analysis, the basic approach is that the future is a reflection of the past. For this reason, artificial neural networks, which produce successful results by solving linear and nonlinear relations due to dynamic learning abilities depending on the examples, are appropriate methods of data analysis. Although the performance of artificial neural networks depends on many parameters, the learning algorithms are one of the most studying topics. Gradient descent, conjugate gradient descent, resillent backpropagation, Quasi-Newton and Levenberg-Marquart algorithms are frequently used classical learning algorithms. In addition to these classical algorithms, heuristic methods can be used to increase the performance of the network. The artificial bee colony algorithm, which is one of the heuristic methods, is developed by modeling the real food search behavior of bees. In the scope of this thesis, has been used to train the recurrent neural networks which produce successful results for time series and has been used in the analysis of the data of the Internet of things stored as time series.

**Keywords:** Internet of things; artificial neural networks; data analysis; artificial bee colony algorithm

# İÇİNDEKİLER

## NESNELERİN İNTERNETİNDE VERİ ANALİZİ İÇİN TEKRARLAYICI SİNİR AĞLARI YÖNTEMİNİN YAPAY ARI KOLONİ ALGORİTMASI İLE EĞİTİLMESİ

BİLİMSEL ETİĞE UYGUNLUK ..... i
YÖNERGEYE UYGUNLUK ..... ii
KABUL VE ONAY ..... iii
TEŞEKKÜR ..... iv
ÖZET ..... v
ABSTRACT ..... vi
İÇİNDEKİLER ..... vii
KISALTMALAR ..... x
SİMGELER ..... xii
TABLOLAR LİSTESİ ..... xiii
ŞEKİLLER LİSTESİ ..... xv

**GİRİŞ ..... 1**

**1. BÖLÜM**

**GENEL BİLGİLER ve LİTERATÜR ÇALIŞMASI**

**1.1. Nesnelerin İnterneti ..... 4**
**1.1.1. Nesnelerin İnterneti Uygulama Alanları ..... 6**
**1.1.2. Nesnelerin İnternetinde Veri Madenciliği ..... 7**
**1.1.3. Nesnelerin İnternetinde Veri Analizi ..... 8**
**1.1.4. Veri Analizinde Kullanılan Teknikler ..... 9**
**1.2. Literatür Taraması ..... 10**
**1.3. Araştırmanın Amacı ..... 15**
**1.4. Araştırmanın Önemi ..... 15**

## 2. BÖLÜM
## YÖNTEM VE MATERYAL

**2.1. Yapay Sinir Ağları ..... 17**
**2.1.1. Yapay Sinir Ağlarının Mimarisi ..... 19**
**2.1.2. Yapay Sinir Ağlarının Çalışması ..... 21**
**2.1.3. Yapay Sinir Ağlarında Kullanılan Aktivasyon Fonksiyonları ..... 21**
**2.1.4. Yapay Sinir Ağlarının Sınıflandırılması ..... 23**
**2.1.5. Sık Kullanılan Ağ Yapıları ..... 25**
**2.1.6. Yapay Sinir Ağlarında Kullanılan Eğitim Algoritmaları ..... 34**
**2.2. Yapay Arı Kolonisi (Artıfıcıal Bee Colony, ABC) Algoritması ..... 43**
**2.3. Performans Değerlendirme Kriterleri ..... 50**
**2.3.1. Hata Fonksiyonları ..... 50**
**2.3.2. Sınıflandırma Modellerinde Başarı Kriterleri ..... 52**
**2.3.3. Regresyon Eğrisi ..... 55**

## 3. BÖLÜM
## BULGULAR

**3.1. Ağ Yapılarının Karşılaştırılması ..... 59**
**3.2. Eğitim Algoritmalarının Karşılaştırılması ..... 63**
**3.3. RNN ağının ABC Algoritması ile eğitilmesi ve Parametre Analizi ..... 67**
**3.3.1. Parametre Aralığının Performansa Etkisi ..... 67**
**3.3.2. Limit Değerinin Performansa Etkisi ..... 68**
**3.3.3. Popülasyon Büyüklüğünün Performansa Etkisi ..... 68**
**3.4. RNN Ağ Eğitiminde ABC Ve LM Algoritmalarının Karşılaştırılması ..... 69**
**3.5. Gerçekleştirilen Modelin Temel Problemlere Uygulanması ..... 70**
**3.6. Gerçekleştirilen Modelin Farklı Mühendislik Problemlerine Uygulanması ..... 73**

## 4. BÖLÜM
## TARTIŞMA – SONUÇ ve ÖNERİLER

**4.1. Tartışma – Sonuç ..... 84**

**4.2. Öneriler ..... 85**

**KAYNAKÇA ..... 87**

**EKLER ..... 100**

**EK 1. Algoritmanın Matlab Kodu ..... 100**

**ÖZGEÇMİŞ ..... 105**

# KISALTMALAR

| Kısaltma | İngilizce Karşılığı | Türkçe Karşılığı |
|---|---|---|
| **IoT** | İnternet Of Things | Nesnelerin İnterneti |
| **ANN** | Artificial Neural Network | Yapay Sinir Ağları |
| **FFNN** | Feed Forward Neural Network | İleri Beslemeli Sinir Ağı |
| **RNN** | Recurrent Neural Network | Yinelemeli (Tekrarlayıcı) Sinir Ağı |
| **GRNN** | Generalized Regression Neural Network | Genelleştirilmiş Regresyon Sinir Ağı |
| **TDNN** | Time Delay Neural Network | Zaman Gecikmeli Sinir Ağları |
| **NARX** | Nonlinear Autoregressive Exogenous | Doğrusal Olmayan Otoregresif Eksojen |
| **RBF** | Radial Basis Function | Radyal Temelli Fonksiyon |
| **ABC** | Artificial Bee Colony | Yapay Arı Kolonisi |
| **BP** | Back Propogation | Geri Yayılım |
| **GD** | Gradient Descent | Eğim Düşüm |
| **GDM** | Gradient Descent With Momentum | Momentum Terimli Gradyan Düşüm |
| **GDA** | Gradient Descent With Adaptive Learning Rate | Adaptif Öğrenme Oranı Terimli Gradyan Düşüm |
| **GDX** | Gradient Descent With Momentum And Adaptive Learning Rate | Momentum Ve Adaptif Öğrenme Oranı Terimli Gradyan Düşüm |
| **RP** | Resilient Backpropagation | Esnek Geri Yayılım |
| **CGD** | Conjugate Gradient Descent | Eşlenik Gradyent Düşüm |
| **CGB** | Conjugate Gradient With Powell-Beale Restarts | Powell/Beale Başlatmalı Eşlenik Gradyent |
| **CGF** | Conjugate Gradient With Fletcher-Reeves Restarts | Fletcher-Reeves Başlatmalı Eşlenik Gradyent |
| **CGP** | Conjugate Gradient With Polak/Ribiére Restarts | Polak/Ribiére Başlatmalı Eşlenik Gradyent |
| **SCG** | Scaled Conjugate Gradient | Ölçeklendirilmiş Eşlenik Gradyent |
| **QN** | Quasi-Newton | Quasi-Newton |

| | | |
|---|---|---|
| **BFGS** | Broyden–Fletcher–Goldfarb–Shanno | Broyden–Fletcher–Goldfarb–Shanno |
| **OSS** | One-Step Secant | Tek Adım Sekant |
| **LM** | Levenberg-Marquardt | Levenberg-Marquardt |
| **BR** | Bayesian Regulaziton | Bayesian Regulaziton |
| **MAE** | Mean Absolute Error | Ortalama Mutlak Hata |
| **MSE** | Mean Square Error | Ortalama Karesel Hata |
| **RMS** | Root Mean Square Error | Ortalama Karesel Hatanın Kökü |
| **MAPE** | Mean Absolute Percentage Error | Oransal Ortalama Mutlak Hata |
| **SMAPE** | Symmetric Mean Absolute Percentage Error | Simetrik Oransal Mutlak Hata |
| **ROC** | Receiver Operating Characteristic | Alıcı İşletim Karakteristiği |

# SİMGELER

| Simge | Karşılığı |
|---|---|
| $\mathbf{x_n}$ | Ağ Girdisi |
| $\mathbf{y_n}$ | Ağ Çıktısı |
| $\mathbf{w_{ij}}$ | Ağırlık Değeri |
| $\mathbf{b_j}$ | Bias Değeri |
| $\mathbf{S_k}$ | Çıkış Katmanı Girdisi |
| $\mathbf{S_j}$ | Ara Katman Girdisi |
| $\mathbf{h_j}$ | Gizli Katman Çıktısı |
| $\mathbf{Y_{dk}}$ | Hedef Çıktı |
| **e** | Hata |
| **E** | Hataya Bağlı Performans Fonksiyonu |
| $\mathbf{\nabla f}$ | Gradyent Vektörü |
| **H** | Hessian Matrisi |
| **J** | Jakobyen Matrisi |
| **g** | Gradyent |
| $\boldsymbol{\alpha}$ | Öğrenme Oranı (Adım Büyüklüğü) |
| **kwH** | Kilowatt Saat |

# TABLOLAR LİSTESİ

Tablo 2.1. Biyolojik Sinir Sistemi Ve Yapay Sinir Ağı Elemanları ..... 19
Tablo 2.2. Aktivasyon Fonksiyonları ..... 23
Tablo 2.3. Öğrenme Algoritmalarının Sınıflandırılması ..... 34
Tablo 2.4. Hata Fonksiyonları ..... 52
Tablo 3.1. İnternet Trafiği Veri Seti Girişleri Ve Çıkışı ..... 59
Tablo 3.2. Ağ Yapılarının Performans Metriklerine Göre Karşılaştırılması ..... 60
Tablo 3.3. CFNN, Elman, FFNN Ve RNN Ağlarının Eğitim Ve Test Verileri için Determinasyon Katsayıları ..... 61
Tablo 3.4. RNN Ağı İçin Eğitim Algoritmalarının Karşılaştırılması ..... 63
Tablo 3.5. Determinasyon Katsayısı İle RNN Ağı İçin Eğitim Algoritmalarının Karşılaştırılması ..... 65
Tablo 3.6. ABC Algoritmasına Aralık Parametresinin Etkisi ..... 67
Tablo 3.7. ABC Algoritmasına Limit Parametresinin Etkisi ..... 68
Tablo 3.8. ABC Algoritmasına Popülasyon Büyüklüğü Parametresinin Etkisi ..... 68
Tablo 3.9. İnternet Trafiği Veri Seti İçin ABC İle LM Algoritmasının Karşılaştırılması ..... 69
Tablo 3.10. Xor Doğruluk Tablosu ..... 71
Tablo 3.11. 3-Bit Parity Doğruluk Tablosu ..... 71
Tablo 3.12. Encoder Decoder Doğruluk Tablosu ..... 71
Tablo 3.13. Temel Problemler İçin ABC Algoritmasıyla Eğitilen RNN Ağının Performansı ..... 72
Tablo 3.14. Geliştirilen Modelin Uygulanacağı Problemler ..... 74
Tablo 3.15. Elektrik Üretim Veri Seti Girişleri Ve Çıkışı ..... 74
Tablo 3.16. Güneş Paneli Güç Üretim Veri Seti Girişleri Ve Çıkışı ..... 75
Tablo 3.17. Trafik Yavaşlığı Veri Seti Girişleri Ve Çıkışı ..... 75
Tablo 3.18. İki Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı ..... 76
Tablo 3.19. Dört Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı ..... 76

Tablo 3.20. Yirmidört Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı ..... 76
Tablo 3.21. Oda Doluluğu Veri Seti Girişleri Ve Çıkışı ..... 77
Tablo 3.22. Çalışılan Veri Setleri İçin RNN Ağının Eğitim Performans Metrikleri ..... 78
Tablo 3.23. Çalışılan Veri Setleri İçin RNN Ağının Test Performans Metrikleri ..... 78
Tablo 3.24. Robot Navigasyon Verisinde Farklı Çalışmaların Başarı Değerleri ..... 81
Tablo 3.25. Doluluk Algılama Problemi İçin Farklı Çalışmaların Doğruluk Oranları ..... 82
Tablo 3.26. Doluluk Algılama Problemi İçin ROC Eğrileri Alanları ..... 83

# ŞEKİLLER LİSTESİ

Şekil 1.1. Yıllara Göre Kişi Başına Düşen İnternete Bağlı Cihaz Sayısı ..... 5
Şekil 1.2. Veri Madenciliği Adımları ..... 8
Şekil 2.1. Biyolojik Sinir Hücresi ..... 18
Şekil 2.2. Yapay Sinir Hücresi ..... 18
Şekil 2.3. Yapay Sinir Ağı ..... 20
Şekil 2.4. Cascade Ağ Yapısı ..... 25
Şekil 2.5. Rekabetçi Ağ Yapısı ..... 26
Şekil 2.6. Elman Ağ Yapısı ..... 27
Şekil 2.7. İleri Beslemeli Ağ Yapısı ..... 27
Şekil 2.8. Tekrarlayıcı Ağ Yapısı ..... 28
Şekil 2.9. RNN Ağ Yapısının Açık Şekli ..... 29
Şekil 2.10. LVQ Ağ Yapısı ..... 30
Şekil 2.11. NARX Ağ Yapısı ..... 31
Şekil 2.12. Hopfield Ağ Yapısı ..... 31
Şekil 2.13. Perceptron Ağ Yapısı ..... 32
Şekil 2.14. Olasılıksal Ağ Yapısı ..... 32
Şekil 2.15. RBF Ağ Yapısı ..... 33
Şekil 2.16. Genelleştirlmiş Regresyon Ağ Yapısı ..... 33
Şekil 2.17. ABC Algoritması Akış Şeması ..... 49
Şekil 2.18. Karışıklık Matrisi ..... 53
Şekil 2.19. ROC Eğrisi ..... 55
Şekil 2.20. Regresyon Eğrisi ..... 56
Şekil 3.1. CFNN, Elman,FFNN ve RNN Ağı Çıktılarının Hedef Verilerle İlişkisi ..... 60
Şekil 3.2. RNN Eğitim Verisi için Regresyon Eğrisi ..... 61
Şekil 3.3. RNN Test Verisi İçin Regresyon Eğrisi ..... 62
Şekil 3.4. FFNN, CFNN, Elman Ve RNN Ağları Test Verileri İçin Yakınsama Sonuçları ..... 62

Şekil 3.5. GD, LM, RP ve BR ile Eğitilen RNN Ağı Çıktılarının Hedef Verilerle İlişkisi ..... 64
Şekil 3.6. Eğitim Verisi İçin BFG Algoritmasına Ait Regresyon Eğrisi ..... 65
Şekil 3.7. Test Verisi İçin BFG Algoritmasına Ait Regresyon Eğrisi ..... 66
Şekil 3.8. Eğitim Algoritmaları İçin Yakınsama Grafikleri ..... 66
Şekil 3.9. ABC Algoritması Yakınsama Grafiği ..... 69
Şekil 3.10. İnternet Trafiği Eğitim Veri Seti İçin Hedef Veri İle ABC, LM Algoritmalarıyla Eğitilmiş RNN Ağının Sonuçları ..... 70
Şekil 3.11. İnternet Trafiği Test Veri Seti İçin Hedef Veri İle ABC, LM Algoritmalarıyla Eğitilmiş RNN Ağının Sonuçları ..... 70
Şekil 3.12. Xor Verisi İçin Yakınsama Grafiği ..... 72
Şekil 3.13. 3-Bit Parity Verisi İçin Yakınsama Grafiği ..... 73
Şekil 3.14. Encoder Decoder Verisi İçin Yakınsama Grafiği ..... 73
Şekil 3.15. Sırasıyla İki, Dört Ve Yirmidört Sensörlü Robotun Şematik Gösterimi ..... 77
Şekil 3.16. Doluluk Algılama Problemi İçin Roc Eğrisi ..... 79
Şekil 3.17. Doluluk Algılama Problemi Eğitim Verisi İçin Karışıklık Matrisi ..... 80
Şekil 3.18. Doluluk Algılama Problemi Test Verisi İçin Karışıklık Matrisi ..... 80

# GİRİŞ

Nesnelerin interneti (Internet of things, IoT) farklı fiziksel nesnelerin birbirleri ile iletişim kurmasını sağlayarak; gerçek dünya ile sanal dünyayı birbirine bağlayan yapıdır. Günümüzün trend konularından olan bu yapı mühendislik uygulamalarının yanında sağlıktan ulaşıma, akıllı evlerden akıllı şehirlere kadar hayatımızın neredeyse her alanına yerleşmiş durumdadır [1]. Yakın gelecekte yirmi milyar nesnenin internete bağlı olacağı öngörüldüğü düşünülürse artan veri miktarının devasa boyutlara ulaşacağı çok açık görülmektedir [2].

Nesnelerin internetindebu kadar veriden anlamlı sonuçlar çıkarmak ancak verilerin etkili analiz yöntemleriyle değerlendirilmesiyle mümkündür. Veri analizi için geliştirilen yöntemlerde temel mantık geleceğin, geçmişin bir yansıması olduğu düşüncesidir. Bu fikre göre geçmişe ilişkin doğru verilerle çalışma ve uygun yöntemlerin kullanılması geleceğin de tahmin edilmesini sağlayacaktır [3]. Bu yaklaşımdan da hareketle özellikle zaman serisi analizinde, geçmişteki örneklerden de yararlanarak dinamik öğrenme yetenekleri sayesinde başarılı sonuçlar çıkarabilen yapay sinir ağları (ArtificialNeural Network, ANN) doğru bir yöntem olabilmektedir.

Günümüzde yapay zekânın gelişimine paralel olaraken hızlı ve en güçlü bilgi işleme aracı olduğu bilinen insan beyninin matematiksel olarak modellenmesiyle ortaya çıkan ANN hayatımızın birçok alanında yerini almıştır. ANN modelleri biyolojik ağların karmaşıklığını tam olarak yansıtmasa da temel yapısından esinlenerek yüksek performans sağlayacak şekilde yoğun bağlantılar içeren basit hesaplama elemanlarından meydana gelmiştir. Bu elemanlar düğüm, yapay sinir hücresi ya da işlemci olarak adlandırılır. Temel bir düğümün bir ya da birden fazla girdisi ve tek çıktısı bulunmaktadır. Çıktıya olan etkilerine göre bu girdiler farklı ağırlıklarla çarpılır. Bu şekilde ağın net girdisi hesaplanır. Bu net girdi doğrusal, sigmoid, lineer, tanjant

hiperbolik ya da eşik değer fonksiyonu olabilen bir aktivasyon fonksiyonundan geçirilerek hücrenin net çıktısı hesaplanır. Hesaplanan net çıktı başka bir hücrenin girdisi olabilir. Son katmandaki hücrelerin net çıktısı ağın tamamı için net çıktıdır. Hedef veriyle ağın çıktısı arasındaki farkı minimize etmek amacıyla ağırlıkların sürekli güncellenerek optimum ağırlık matrisinin oluşturulmasına ağın eğitilmesi, optimum ağırlık matrisleri ile farklı veriler üzerinde ağın performansının hesaplanmasına ağın test edilmesi ve ağın istenen performansa ulaşmasına ağın öğrenmesi denir [4].Öğrenme süreci doğrudan kullanılan eğitim algoritmalarıyla ilişkilidir

Bu algoritmalar öğrenim sürecinde kullandıkları yöntemlere göre birbirinden ayrılırlar. Klasik algoritmalarda sınıflandırma yapılırken toplam hata fonksiyonuna göre ilgili ağırlığın türevi kullanan gradyan vektörü, ilgili hücrenin hata fonksiyonuna göre ağırlığın türevinden yararlanan Hessian matrisi ve ilgili hücrenin hata fonksiyonuna göre ağırlığın ikinci türevinden yararlanan Jakobyen matrisi temel alınmaktadır. Bu algoritmalar, gradyan vektörünü kullanan gradyan düşüm (Gradient Descent, GD); bu vektöre dik ardışık gradyan yönünü kullanan eşlenik gradyan düşüm (Conjugate Gradient Descent, CGD); vektörün işaretinden yararlanan esnek geri yayılım (ResilientBackpropagation, RP); Hessian matrisini kullanan Quasi-Newton (QN) ve Jakobyen matrisini kullanan Levenberg-Marquardt (LM) algoritmalarıdır.

ANN eğitiminde bu klasik algoritmaların kullanılabilmesinin yanında ağın performansını arttırmak amacıyla farklı sezgisel yöntemlerin geliştirilmesi araştırmacıların hala ilgisini çekmektedir. Bu sezgisel yöntemlerden biri olan yapay arı koloni algoritması (Artificial Bee Colony, ABC) ANN ağlarını eğitmek için kullanılan sezgisel yöntemlerden bir tanesidir. 2005 yılında Karaboğa tarafından önerilen bu algoritma arıların doğada yiyecek arama davranışlarından esinlenilerek geliştirilmiştir [5]. Birçok mühendislik probleminin çözümünde kullanılmasının yanında [6] ANN eğitiminde de oldukça başarılı sonuçlar verdiği bilinmektedir [7; 8].

Bu tez çalışmasının amaçlarından biri farklı yapay sinir ağı modellerinin ve bilinen farklı eğitim algoritmalarının örnek bazı veri setleri üzerinde performans karşılaştırmalarının yapılmasıdır. Bu amaç doğrultusunda özellikle IoT verilerine benzemesi açısından anlık ve zaman serisi olarak saklanan veriler tercih edilmiştir. Burada tekrarlayıcı ya da yinelemeli ağ yapıları (RecurrentNeural Network, RNN)

olarak adlandırılan ve geri besleme bağlantıları içeren ağların bu tür veri setlerinde diğer ağ modellerine nazaran daha başarılı olduğu gözlemlenmiştir. Diğer bir amaç ise, RNN ağlarının eğitiminde ABC algoritmasının kullanımını gerçekleştirmek, algoritmanın parametrelerini detaylıca analiz etmek ve XOR,3-bit parity ve encoder decoder problemleri için elde edilen sonuçları klasik eğitim algoritmaları ile karşılaştırılmaktadır. Son olarak gerçekleştirilen modeli elektrik üretimi, solar güç, kentsel trafik ve robot yönlendirme problemlerinin IoT teknolojiler ile toplanan verilerinin analizinde kullanarak sonuçları değerlendirmektir.

Bu amaçlar doğrultusunda tez akışı şu şekilde planlanmıştır: Bölüm 1'de nesnelerin interneti, veri analizi ve literatürdeki benzer çalışmalar anlatılmıştır. Bölüm 2'deANN temel yapısı, ağların sınıflandırılması, RNN ağ yapısı, ABC algoritması ve performans karşılaştırmalarında kullanılan hata ölçüm metrikleri verilmiştir. Bölüm 4'de zaman serisi örneği olan internet trafiği için ağ yapıları ve eğitim algoritmaları karşılaştırılmış, buradan elde edilen sonuçlara göre ABC algoritması ile gerçekleştirilen RNN ağ yapısı temel problemlere uygulanmış ve farklı mühendislik problemlerinde veri analiz yöntemi olarak kullanılmış elde edilen sonuçlar yorumlanmıştır. Bölüm 5'de tez çalışmasının kısa bir değerlendirmesi yapılarak, tezin katkısı ve ileriye yönelik yapılması planlanan çalışmalara değinilmiştir

# 1. BÖLÜM

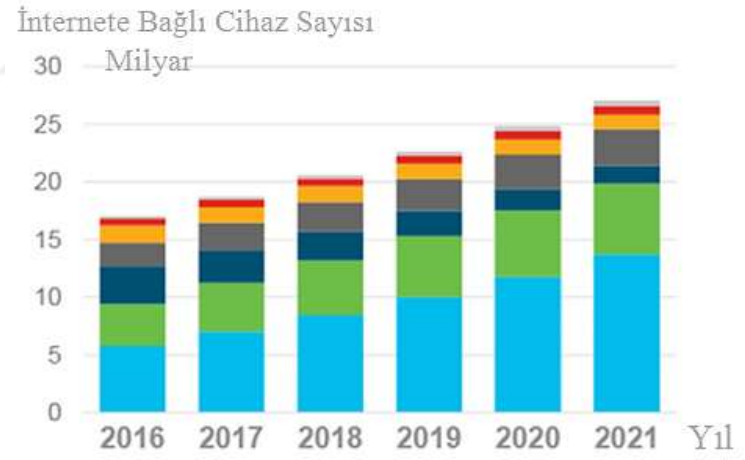
# GENEL BİLGİLER ve LİTERATÜR ÇALIŞMASI

## 1.1. Nesnelerin İnterneti

Yazılım, donanım ve iletişim teknolojisindeki hızlı gelişmeler beraberinde fiziksel dünyadan veri sağlayan internete bağlı sensör teknolojisini geliştirmektedir [9]. Giderek artan kullanımlarıyla beraber bu cihazlar sadece internete bağlı olmakla kalmayıp siber dünya ile fiziksel dünya arasındaki bağlantıyı da sağlamışlardır [10]. Bu bağlantılar bütün nesnelerin internete bağlı olduğu ve haberleşme protokolleri sayesinde kendi aralarında iletişim kurarak bilgi paylaşımı yaptıkları çok geniş bir ağ olan IoT kavramının da temelini oluşturmuştur [1]. Bu kavramı ilk olarak ortaya atan KevinAshton, bilgisayar ya da internet ortamına aktarılan gerçek dünyadaki verilerin, insanlar tarafından ifade edilebilecek bilgilerden çok daha fazlası olduğunu ve bu verilerin nesnelerden alınabileceğini öne sürmüştür. Bu bağlamda radyo frekanslı tanımlama (Radio Frequency Identification, RFID) ve sensör teknolojileri ile daha çok verinin insan müdahalesi olmaksızın işlenerek zaman ve maliyet açısından büyük oranda tasarruf edilebileceğini savunmuştur [11].IoT ile ilgili literatürde çok fazla tanımlama ile karşılaşmak mümkündür. Uluslararası Telekomünikasyon Birliği tarafından yapılan tanıma göre IoT herhangi bir zamanda, herhangi bir yerden her türlü nesnenin bağlantı kurabilmesidir. IoT bilgi toplumu için küresel bir altyapı olup var olan ve gelişen iletişim ve bilgi teknolojilerine dayanarak sanal ve fiziksel dünya arasında kurduğu bağlantı sayesinde gelişmiş hizmetler sağlayan yapıdır [12].

Nesnelerin akıllanmasını sağlayan bu oluşum henüz başlangıç evresinde olarak görülmesine rağmen günümüzde ulaşımdan, sağlığa, ev ve iş yeri ortamından, kişisel alana ve engelliler için olan uygulamalara kadar birçok alanda yerini almıştır [1].

İnternetin de hayatımızın her alanına girmesiyle giderek yaygınlaşan bu kavram otomatikleşen dünyamızda hayatımızı daha kolaylaştırmak ve her alanda yardımcı olabilmek için çağımızın en güncel araştırma konularından biri haline gelmiştir. Öyle ki Gartner'ın Şubat 2017'de yayımladığı rapora göre 2017'de 8,4 milyar bağlı nesnenin kullanılarak 2016 ya oranla %36 lık bir artışın yaşandığı görülmektedir. Yine bu rapora göre 2020'de 20 milyar nesnenin kullanılacağı öngörülmektedir. Başka bir çalışmaya göre ise IoT akıllı nesnelerinin 2020 sonunda dünya genelinde 212 milyar varlığa ulaşması 2022 yılında ise makineler arasındaki trafik akışının tüm internet trafiğinin %45 kadarını teşkil etmesi beklenmektedir [2]. Şekil 1.1'de Cisco'nun 2017 yılındaki rapora göre internete bağlı cihaz sayısı ve cihaz türleri görülmektedir. Renkler aşağıdan yukarı M2M,akıllı telefon, telefon, televizyon, bilgisayar, tablet ve diğer cihazları simgelemektedir. Rapora göre 2016 yılında tüm cihazların %34'ü olan makineden makineye (Machine to Machine, M2M) cihazlarının oranının 2021 yılında %51'e çıkması beklenmektedir [13].




*Şekil 1.1.* Yıllara Göre Kişi Başına Düşen İnternete Bağlı Cihaz Sayısı

Nesnelerin interneti sistemleri temelde nesne, insan, veri ve süreç olmak üzere dört bileşenden oluşur. Nesne bileşeni, ağda bulunan internete ve birbirine bağlı olan fiziksel cihazlardır. Sensörler, kontrolörler ve aktivatörler bu bileşenlere örnek olarak verilebilir. İnsan bileşeni sisteme gerekli bilgiyi sağlamakla görevli bir bileşendir. Veri bileşeni insan ve nesnelerin ürettiği bilgilerdir. Süreç bileşeni ise diğer üç bileşen arasındaki etkileşimi gösterir [14].

### 1.1.1. Nesnelerin İnterneti Uygulama Alanları

Akıllı evler, akıllı şebekeler, izleme ve kontrol uygulamaları, güvenli madencilik, yiyecek kontrol, doğal afetlerin önceden tahmini, hasta hizmetleri, akıllı tarım, akıllı ulaşım, akıllı şehirler, akıllı güvenlik, lojistik,tedarik zinciri ve endüstriyel internet bu uygulamalardan bazılarıdır [15].

Ev aletleri arasındaki etkileşim, enerji tüketiminin optimum şekilde ayarlanması, duman ve gaz algılama sistemleri ile güvenliğin sağlanması, web ve mobil uygulamalarla uzaktan denetim sayesinde evin uzaktan izlenebilmesi ve gerektiği anda güvenlik önlemlerinin sağlanabilmesi IoT cihazların akıllı evlerdeki büyük oranda etkisini göstermektedir [16].

Akıllı trafik uygulamalarında akıllı trafik ışıkları, mobil ve statik sensörler trafiğin rahat yönetilmesini sağlanmaktadır. Toplu ya da özel taşıma araçlarıyla ulaşımı etkili bir şekilde sağlanırken, trafiği izleyerek anlık sıkışıklık durumları sürücülere bildirilerek alternatif rotalar sunulmaktadır [17; 18]. Ayrıca belediyelere yollardaki çukur ya da kayganlık durumları gibi aksaklıklarla ilgili bilgi verilerek güvenlik seviyesini artırılmaktadır. Trafikte kullanılan IoTsensörleri, trafik durumu ve hava durumu verileri aracılığıyla kaza oranlarının düşmesine, trafik yoğunluğu azaltılmasına, park yeri sorunlarının önüne geçilmesine katkı sağlarkençevre kirliliğinin hafifletilmesinde de etkili olmaktadır [19; 20].

Akıllı şebeke uygulamalarında, yenilenebilir enerji kaynaklarının kullanımını artırmak ve tüketicilere daha iyi hizmet verebilmekiçin akıllı sayaçlardan alınan anlık enerji tüketim verileri ile enerji yönetimi sağlanmaktadır. Akıllı fişler, akıllı şarj cihazları bu amaçla kullanılan bazı IoT cihazlarıdır [21]. Akıllı evler, akıllı trafik, akıllı şebeke uygulamalarının bütünleşik olarak kullanılmasıyla oluştuğu söylenebilen akıllı şehirler bilgi toplamak için sensör ağları, ekranlar, kameralar, akıllı metreler ve termostatlar gibi cihazlarla donatılmıştır [22; 23; 24].

Sağlık hizmetlerinde de çeşitli IoT uygulamalarıyla karşılaşmamız mümkündür .IoT tabanlı bir sağlık uygulaması temelde uzaktan izleme, tıbbi cihazları belli saatlerde çalıştırma ve basit tedavi yöntemleri önerme gibi içerikleri barındırır, Alzheimer

hastaları için konum tespiti aracılığıyla güvenliği sağlamak da yaygınlaşan uygulama alanlarından biridir [15; 25; 26].

Doğal afet uygulamaları kapsamında çevresel izleme ve bölgesel koruma için hava kirliliği, su kirliliği ve daha ileri seviyede havadaki radyasyon ve alerjen tespiti bulunmaktadır [27]. Sıcaklık, nem, duman, su seviyesi vb. sensörleri kullanarak olası bir felaketi önceden algılayabilen erken uyarı sistemleri ile internet üzerinden kolayca yayılarak gerekli önlemlerin alınmasını sağlamaktadır ayrıca daha sonrasında da IoT cihazları sayesinde yardım ekiplerini öncelikli bölgelere yönlendirmekte ve risk alanının tahliyesi için güvenli yollar belirlemektedir.Geçmişte ve günümüzde çok fazla can kaybına sebep olan madencilik uygulamalarında da madencilerin yerini izleyerek, yer altı sensörlerinden gelen zararlı gaz,toz gibi verilerin istenmeyen düzeylere yaklaşması durumunda IoT cihazları sayesinde erken önlemle güvenlik artırılıp kayıpların önüne geçilebilir [28].

Tedarik zinciri uygulamaları ürünün hammadde halinden son kullanıcıya ulaşana kadar bütün üretim aşamalarını kapsar, bu aşamalarda IoT ile ürünün son durumu, konumu gibi bilgiler alınarak üretim ve kullanıcıya ulaşım hattındaki gecikmeler, hangi aşamada olduğu gibi bilgiler alınabilir. [29].

### 1.1.2. Nesnelerin İnternetinde Veri Madenciliği

Nesnelerin internetinde ortak bir amaca yönelik olarak cihaz, ağ, teknoloji ve insan kaynaklarından alınan yüksek miktarda verinin toplanmasından daha önemli olan analiz edilerek kullanılabilir bilgiler elde edilebilmesidir. Bu şekilde veriler anlamlı olacak ve etkin kararlar alınabilecektir.Veri analizinin anlamlı bir şekilde ilerleyebilmesi için şu aşamaları içermesi gerekmektedir [30] [31] :

**Uygulama alanının incelenmesi:** Uygulama alanının ve amacının belirlenmesidir.

**Amaca uygun veri kümesi yaratma:** Verinin uygulamanın amacına yönelik olarak seçilmesidir.

**Veri ayıklama ve önişlem:** Gereksiz görülen verilerin yok edilmesidir.

**Veri azaltma ve veri dönüşümü:** Amaca uygun özelliklerin belirlenmesi, özellikler arasındaki ilişkilerin belirlenmesi ve verinin kullanılabilecek şekilde biçimlendirilmesidir.

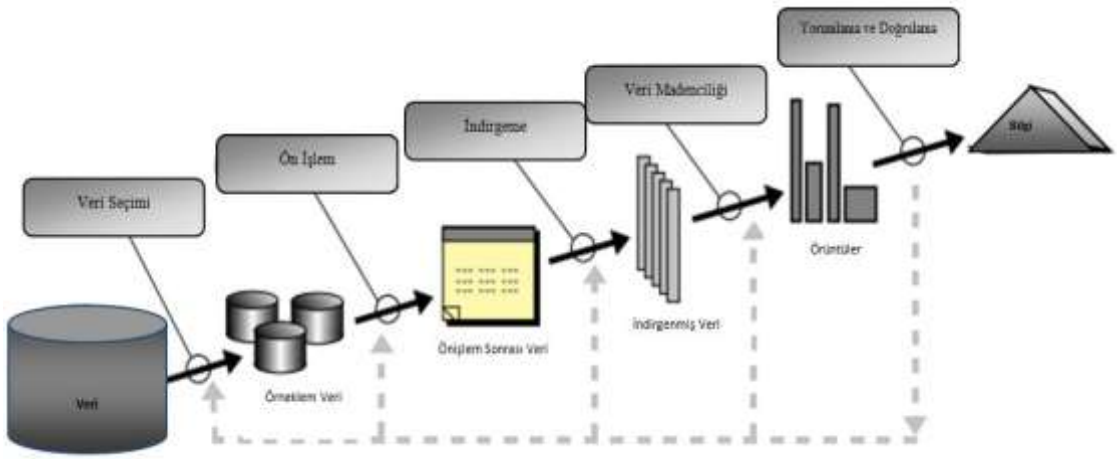
**Veri madenciliği tekniği seçme:** Veriler arasındaki ilişkileri ortaya çıkarabilecek matematiksel ya da sezgisel yöntemlerin belirlenmesidir. Sınıflandırma, birliktelik kuralları, kümeleme tekniklerini içerir.

**Veri madenciliği algoritması seçme:** Bir önceki aşamada belirlenen tekniğe uygun algoritma seçiminin yapılmasıdır.

**Model değerlendirme ve bilgi sunma:** Sonuçların değerlendirilerek grafikler, tablolar vb. etkili sunum araçlarıyla aktarılmasıdır.

**Bulunan bilginin yorumlanması:** Elde edilen sonuçların matematiksel yöntemler veya istatistiksel testlerle anlamlı bir şekilde ortaya koyulmasıdır.

Yukarıda anlatılan aşamaların şematik gösterimi Şekil 1.2'de verilmiştir




*Şekil 1.2.* Veri Madenciliği Adımları [23]

### 1.1.3. Nesnelerin İnternetinde Veri Analizi

Nesnelerin internetinde ele aldığımız problemleri analizlerin konusu açısından dört ana başlık halinde incelememiz mümkündür:

**Akış (Streaming) Analizi**: Bu veri analiz biçimi olay akışının işlenmesi olarak da adlandırılır. Büyük veri (bigdata) olarak adlandırılan devasa miktardaki veri setlerinin analizi bu kapsamdadır. Acil önlem gerektiren acil durum verilerinin analizinde de bu yöntemden faydalanılır. Finansal işlemler, hava filosu takibi ve trafik verilerinin analizi gibi uygulamalar örnek verilebilir.

**Mekansal (Spatial) Analiz:** Fiziksel nesneler arasındaki coğrafi ilişkilerin belirlenmesinde kullanılan konuma dayalı uygulamalar bu kapsamda incelenir. Akıllı park etme, coğrafi uygulamalar örnek verilebilir.

**Zaman Serisi (Time Series) Analizi:** Bir serinin zaman serisi olarak isimlendirilebilmesi için geçmişteki değerlerin geleceği etkilediği zamana bağlı durumların meydana gelmesi gerekmektedir. Bu verileri tanımlamak, aralarındaki ilişkileri ortaya çıkarmak, geleceğe yönelik tahmin yapmak ve verileri açıklayabilecek uygun matematiksel modeller geliştirebilmek amacıyla sıkça kullanılan analiz türüdür. Hava tahmini, borsa, ekonomi, kalite kontrol, kestirimci bakım ve sağlık izleme uygulamaları örnek verilebilir.

**Önlemsel (Prescriptive) Analiz:** Tanımlama ve tahminin bir arada kullanılmasıyla oluşan analiz türüdür. Erken fark edilen risk için önceden önlem alınabilir. Ticari uygulamalar ve mühendislik problemlerinde sıkça yararlanılan bu analiz türü işletmeler için büyük oranda fayda sağlar.

### 1.1.4. Veri Analizinde Kullanılan Teknikler

Veri analizinde yaygın olarak kullanılan yöntemler şunlardır [32];

**İstatistiksel Yöntemler**: Temel bir istatistik problemi olan veri analizi çok boyutlu analiz problemleri kapsamında incelenir ve bu amaçla sınıflandırma, regresyon, kümeleme, boyut azaltma, hipotez testi, varyans analizi, ilişki kurma gibi teknikler uzun yıllardır kullanılmaktadır [26].

**Bellek Tabanlı Yöntemler:** Sık kullanılan yöntemlerden en yakın komşu algoritmaları, bellek tabanlı yöntemler kapsamına girmektedir. 1950'li yıllarda önerilen bu yöntemler o yıllarda bilgisayarların tam anlamıyla gelişememesi, işlemci güçlerinin yetersiz

olması gibi nedenlerden dolayı o yıllarda yeterince ilgi görmemiş ilerleyen zamanlarda kullanılmaya başlanmıştır [33].

**Yapay Sinir Ağları:** Lineer olmayan ilişkileri kolaylıkla çözebilen ANN'lar istatiksel yöntemler gibi veri hakkında parametrik bir model varsaymaz yani uygulama alanı daha geniştir ve bellek tabanlı yöntemler kadar yüksek işlem ve bellek gerektirmez. Problemin türüne ve amacına göre farklı yapıda ANN'ler ile çalışmak mümkündür.

**Karar Ağaçları:** Diğer bütün yöntemlerden farklı olarak çıkan sonucu yorumlamak için elverişlidir. Kural çıkarma ile veri madenciliği çalışmasının sonucunun geçerlenmesini sağlar.

### 1.2. Literatür Taraması

Günümüzün trend konularından olan IoT, veri analizi ve yapay sinir ağları ile ayrı ayrı literatürde çok fazla çalışmayla karşılaşmamız mümkündür. Çok yeni bir araştırma konusu olmasına karşın IoT'da veri analizi, IoT'a giderek artan ilginin ve çok fazla uygulama alanında kısa zamanda başarıyla uygulanması sonucu literatürde kendine yer edinmiştir. Literatür taraması IoT'da veri analizi, veri analizinde ANN kullanımı, farklı eğitim algoritmalarının ANN performansına etkisi, RNN uygulamaları, ANN eğitiminde ABC algoritması ve hibrit ABC algoritması kullanımını içermektedir.

Chen ve arkadaşları IoT'da veri analiz yöntemlerini bilgi, teknik ve uygulama kapsamında sınıflandırma, kümeleme, ilişki analizi, zaman serisi analizi ve aykırılık analizi açısından daha önce yapılmış çalışmalar kapsamında incelemişlerdir [34] . Bhatia ve Patel IoT'da veri analizini aykırılık analizi; kümeleme, sınıflandırma, özellik seçimi ve zaman serisi başlıkları altında incelemişlerdir. Aykırılık analizi için en yakın komşu algoritmaları; sınıflandırma için karar ağaçları, K en yakın komşu algoritması, destek vektör makineleri, Apriori ve Adaboost algoritmaları; kümeleme için yoğunluğa dayalı hibrit veri akış kümeleme algoritmaları; özellik seçimi için filtresel, gömülü ve sarıcı özellikteki yöntemleri içeren algotirmalar; son olarak zaman serisi analizi için ise yinelemeli Bayesian modelleri önermişlerdir [35]. Bin ve arkadaşları IoT'da veri madenciliği için çok katmanlı model, dağıtılmış model, ızgara tabanlı model ve çoklu teknoloji entegrasyon modeli olmak üzere dört farklı model önermişlerdir. Çok katmanlı

model veri toplama, veri yönetimi, bilgi işleme ve servis tabakası katmanlarından oluşmaktadır, dağıtılmış model farklı kaynaklardan gelen depolanmış veriye dair problemleri çözmektedir, ızgara tabanlı model ızgara çerçevesinde fonksiyonların uygulanmasını sağlarken entegrasyonmodeli gelecek internetinin çerçevesini açıklamaktadır [36].

Literatürde veri analizinde ANN kullanımı ile ilgili olarak çok fazla çalışmayla karşılaşmamız mümkündür. Li, IoT'da veri analizi için alternatif bir yöntem olarak geri yayılım algoritması (BackPropagation, BP) kullanan ağları önermiştir. Algoritmanın momentum ve adaptif öğrenme katsayısı ile birlikte kullanıldığında daha başarılı sonuçlar verdiği görülmüştür [37]. Kamruzzaman ve arkadaşları veri madenciliği problemlerinden olan sınıflandırma için ANN önermişlerdir. Bu amaçla üç katmanlı aktivasyon fonksiyonu olarak hiperbolik tanjant kullanan bir ileri beslemeli sinir ağı (FeedForwardNeural Network, FFNN) kullanmışlardır. Analiz verinin hazırlanması ve kural çıkarımı olmak üzere iki aşamadan oluşmuştur [38]. Nirkhi self-organizing map, neuro-fuzzy, adaptif rezonans teorisi ve BP sinir ağlarıyla yapmış olduğu sınıflandırma problemleri için veri analizinde yapay sinir ağlarının iyi bir seçenek olabileceğini ve parçacık sürü algoritması (Particle Swarm Optimization, PSO) ve karınca koloni algoritması gibi algoritmalarla geliştirildiğinde daha yüksek performans sağlayarak, daha iyi çözümler üretebileceğini öne sürmüştür [39]. Alam ve arkadaşları içerisinde yapay sinir ağları da bulunan sekiz farklı madencilik algoritmasını karşılaştırmışlardır ve henüz başlangıç aşamasında olan derin öğrenme algoritmasına sahip yapay sinir ağlarının çok iyi öğrenme oranına sahip olduğunu ve ileride çok yüksek boyutlu verileri yüksek başarı oranıyla analiz edebileceğini öngörmüşlerdir [40]. Singh, ANN'ların kara kutu ismiyle anılan yapısı tam çözülemese de veri analizi için gelecek vaat eden bir araç olduğunu ve medikal, borsa finans ve endüstri uygulamaları için araştırılması gerektiğini belirtmiştir [41].

Kısa sürede kendisine çok fazla uygulamada yer bulan tekrarlayıcı sinir ağları günümüze kadar çok çeşitli alanlarda, çok sayıda uygulamalarda kullanılmışlardır. Liu ve arkadaşları yapmış oldukları çalışmalarda hava istasyonlarından elde ettikleri geçmiş hava ve radyasyon dedektöründen aldıkları geçmiş radyasyon verilerinden yararlanarak arka plandaki radyasyon seviyesini belirlemeye çalışmışlardır. Bu verilerden hem yüksek hem düşük gürültü seviyeleri ile kullanılmak üzere öncelikle sadece radyasyon

verisinden, daha sonra hem radyasyon hem hava durumu verisinden yararlanmışlardır. RNN ve hareketli ortalama yöntemini kıyaslamış, yüksek gürültü oranında RNN daha iyi sonuca üretirken düşük gürültü oranında her iki modelin de benzer sonuçlar ürettiğini gözlemlemişlerdir [42]. Catalbas hareketli bir robotun bacak açılarının kontrolü için RNN sinir ağından yararlanmıştır [43]. Shimodaria çok katmanlı algılayıcıların, sonlu darbe cevabına sahip çok katmanlı algılayıcıların ve RNN'lerin zaman serisi tahmininde performanslarını incelemiş ve RNN'lerin daha yüksek performans gösterdiğini belirlemişlerdir. Ayrıca başarılı sonuçlar için çok büyük veri setlerine ihtiyacı olmadığını gözlemlemişlerdir [44]. Samarawickrama ve Fernando, FFNN, RNN, RNN in özelleştirilmiş türleri olan uzun kısa süreli hafıza (Long Short Term Memory, LSTM) ve tek geçitli yinelemeli (Gated Recurrent Unit, GRU) ağı ekonomi alanında, hisse senedi tahminine dayalı uygulamalar için kıyaslamıştır [45]. Hanifa ve arkadaşları ön işlemden geçen veriyi RNN ile modelleyerek uçuş güvenliği seviyesini artırmıştır. Elde ettiği sonuçları MultipleKernel metodu ile kıyaslamıştır. RNN doğruluk değeri açısından %90.17 değerinde performansla daha iyi bir sonuç verirken recalldeğeriiçin tersi bir durum söz konusu olmuştur. %59 olan bu değer daha kötü sonuç vermiştir [46]. Zozo ve arkadaşları çok kısa süre içinde konuşmadan konuşmacının yaşını algılamak için modelledikleri sistemde RNN'in özelleştirilmiş türü olan LSTM den yararlanmışlardır. Deneyler erkek ve kadın bireyler için ayrı ayrı yapılmış ve yaklaşık olarak aynı sonuçlar alınmıştır. Bu da sistemin cinsiyetten bağımsız olduğunu kanıtlamıştır. [47]. Zhang ve arkadaşları RNN'i görüntü tabanlı robot kontrol problemi için kullanmışlardır. Kamerasında lens olan Puma 560 robotunu kullanan araştırmacılar, piksel hataları sıfıra yaklaştıkça, eklem hızları ve açılarının da tatmin edici olduklarını gözlemlemişlerdir [48]. Zhang ve arkadaşları kısa vadede hava aracı güvenliği için RNN'i ve uzun vadede tahminler için çok yönlü bağlantılar içeren özelleştirilmiş bir RNN kullanmışlardır. Havalandırma sisteminin sıcaklık sensöründen aldığı verileri kullanan ağ ortalama karesel hatanın karekökü (Root Mean Square, RMS),değerleri açısından bakıldığında kısa vade ve uzun vadede önerilen yapıların var olan yapılardan iyi sonuç verdiği gözlemlenmiştir [49]. Abhigna ve arkadaşları RNN ve FFNN'iokyanustaki dalga boyundan alınan verilere göre hortum, deprem ve tsunami yaratabilme tahmini için kullanmış ve kıyaslamışlardır [50]. Nemirovskiy ve arkadaşları LM, CGD ve BR (BayesianRegulazition, BR) algoritmalarıyla eğittikleri sinir ağlarını tüm renklere ayrıştırma bakımından incelemiş ve en iyi sonuçları BR algoritmasıyla

eğitilen RNN ağından almışlardır [51]. Yu ve arkadaşları dizel motoru modellemek için RNN ağ yapısını önermiştir. Bu modelde dişlinin yer değişimi, dönme hızı ve yükü RNN'in özelleştirilmiş yapısı olan doğrusal olmayan otoregresif(NonlinearAutoregressiveExogenous, NARX) ağ yapısını eğitmek için kullanmışlardır [52]. Shen ve arkadaşları konveyör enerji verimliliğini artırmak için RNN yapısını önermişlerdir. RNN ile modelledikleri sistemde veri setinde malzeme tüketimi, minimum ve maksimum stok verilerini kullanmışlardır. Sonuçlar beş saniye içinde hata değerinin neredeyse sıfırlandığını göstermiştir [53]. Li ve arkadaşları cerrah robotların kontrol hassaslığını artırmak ve robot hareketine uyarlanabilirliğini geliştirmek için RNN ağ yapısını önermişlerdir. Çalışma için robotun kinematik analizi ile sınır koşulları elde edilmiştir. Ağ bu verileri değerlendirerek sistemin anlık durumuna bağlı olarak dinamik bir şekilde robot hareketlerini değiştirebilmektedir. Sonuçta RMS, değeri hassaslık için 0.016 m çıkmıştır ve diğer yöntemlere göre daha başarılı bir performans gösterdiğini kanıtlamaktadır. Güvenilirlik için farklı gürültü seviyeleri ile test edilen ağ yine iyi sonuçlar üretmiştir [54]. Wafa ve arkadaşları fotovoltaik hücreyi RNN ile modellemişlerdir. Giriş verileri sıcaklık, güneş ışığı ve gerilimdir çıkış ise akımdır. Ağın çıktısı, gerçek akım değerini yüksek duyarlılıkla izlemiştir. Normalize edilmiş RMS değeri 0,0559 olarak bulunarak ağın performansının yüksek olduğunu kanıtlamıştır [55]. Yunpeng ve arkadaşları LSTMağ yapısı ile zaman serileri problemleri için tahminler yapmışlardır. Bu amaçla elektrik akımı verisini güçlü periyodik veri seti, havayolu yolcu verisini artış, azalışlar içeren periyodik veri seti, iletişim cihazlarının çıkış gücü verisini ise çok uzun veri seti olarak kullanmışlardır. Sonuçları bütünleşik otoregresif hareketli ortalama (Auto Regressive İntegrated Moving Average, ARIMA) modeli ve genelleştirilmiş regresyon sinir ağı(Generalized Regression Neural Network, GRNN)modeli için değerlendirmişlerdir. Güçlü periyodik veri için en iyi sonuç ortalama karesel hata (Mean Square Error, MSE) değeri 0.02 olarak ARIMA'dan, periyodik veri için (MeanSquareRoot Of PercentageError, MSPE) değeri 0.059 olarak LSTM RNN'den alınmıştır. LSTM RNN çok uzun veri setine uygulandığında MSE değeri 6.856 olarak elde edilmiştir [56]. Amirkhan, kolorektral kanserini önceden tahmin edebilmek için RNN ağ yapısını önermiştir ve ağın performansı çok başarılı olduğu için diğer hastalıkların erken teşhisinde de kullanılabileceğinióne sürmüştür [57]. Ciu ve arkadaşları olası sistem hatalarını önceden belirleyip önlem alabilmek amacıyla, rulman yatağı arızalarının kestirimci bakım ile

önceden tespiti için RNN modelini önermişlerdir. Giriş verileri olarak frekans uzayında dönüştürülmüş titreşim sinyallerini kullanmıştır; sonuçlar ağın büyük ölçüde başarılı olduğunu göstermiştir [58]. Lukito ve arkadaşları Wi-Fi üzerinden topladıkları verileri iç konumlandırma sistemi için RNN üzerinde modellemişlerdir. Sonuçlar incelendiğinde % 82.47 doğruluk oranına sahip RNN; çok katmanlı algılayıcı (Multi Layer Perceptron, MLP), NaïveBayes, J48 ve destek vektör makinesinden iyi olmasına karşın %83.58 doğruluk K en yakın komşu algoritmasının gerisinde kalmıştır [59]. Tornai ve arkadaşları akıllı şebekelerde verimlilik için RNN ve özelleştirilmiş yapılarıyla kullanıcı sınıflandırılmasını önermiştir ve özelleştirilmiş yapı olan NAR-RNN 0.9 ile 1 arasında olan sınıflandırma oranıyla en başarılı model olmuştur. [60].

Literatürde ANN eğitiminde optimizasyon algoritması olan ABC algoritmasının kullanıldığı çalışmalar giderek artmaktadır. Karaboga ve arkadaşları ileri beslemeli sinir ağı eğitiminde ABC algoritmasından yararlanmıştır [61]. Karaboga ve arkadaşları makine öğrenmesi alanında çok geniş alanda uygulamaları bulunan örüntü sınıflandırma için yararlandıkları ileri beslemeli sinir ağını ABC algoritmasıyla eğitmişlerdir [62]. Hsieh ve arkadaşları wavelet dönüşümlerinden yararlanarak oluşturdukları veri setlerini RNN sinir ağında ABC algoritmasıyla eğiterek hisse seneditahmini için kullanmışlardır [63]. Karaboga ve Baştürk ABC algoritmasının parametre analizini yaparak, performansını çok boyutlu sayısal problemler için PSO, evrimsel algoritma ve diferansiyel gelişim algoritmalarıyla kıyaslamışlardır [64]. Hajimirzaeive arkadaşları bulut bilişimde saldırı algılama için önerdikleri sistemde ağ trafiğini ABC algoritmasıyla eğitilmiş çok katmanlı algılayıcı ağı ile modellemişlerdir [65]. Kisi ve arkadaşları nehir tarafından taşınan askıda sediment konsantrasyonunun tahmini için ABC algoritmasıyla eğitilmiş ağdan yararlanmışlardır [66]. Alpaslan ve arkadaşları ABC ve BP eğitim algoritmalarını tek çoğaltıcılı sinir ağı modeli için kıyaslamışlardır ve sonuçlar ABC algoritmasının BP algoritmasından daha başarılı olduğunu ortaya koymuştur [67].

Literatürde ağları eğitmek için yalnız ABC algoritmasını kullanan kaynaklara sıkça rastlayabileceğimiz gibi ABC algoritmasından yararlanılarak geliştirilen hibrit algoritmalarla da karşılaşmamız mümkündür. Irani ve arkadaşları yerel minimum ya da maksimum noktalarını bulma gücünden yararlandıkları BP algoritmasını ABC algoritması ile birlikte kullanmışlardırve ABC'nin küresel araştırma yeteneğiyle,

dengesiz sondajda alt delik basınç tahmini amacıyla kullanılan sinir ağının başarısını arttırmıştır [68]. Ozturk ve arkadaşları yapay sinir ağlarını eğitmede sıkça kullanılan bir algoritma olan LM algoritmasını ABC algoritması ile birleştirerek ABC-LM algoritmasını ağ eğitmek için kullanmışlardır [69]. Ghanem ve arkadaşları genetik algoritmalardan olan PSO algoritmasından yararlanarak hibrit ABC-PSO algoritmasını ileri beslemeli ağların eğitimi için kullanmışlardır [70]. Shah ve arkadaşları zaman serileri kullanımında çok katmanlı perceptron ağını eğitmek için ABC algoritmasının iyileştirilmiş farklı modellerini önermişlerdir [71].

### 1.3. Araştırmanın Amacı

Bu tez çalışmasındaIoT üzerinden zaman serisi formatında alınan verilerin, ANN modellerinden olan zaman serisinde başarılı performans sergileyen RNN ile analiz edilmesi, bu analizin başarısının artırılması amacıyla ağın optimizasyon problemleri için geliştirilen ABC algoritmasıyla eğitilmesi ve bu şekilde veri analizinde yüksek performans sağlanması hedeflenmektedir.

### 1.4. Araştırmanın Önemi

IoT veri analizi gün geçtikçe daha önemli bir araştırma konusu haline gelmiştir. Yüksek miktarda veri üreten IoT, IBM tarafından önerilen akıllı gezegenin çekirdeğinde bulunmaktadır [36]. IoT'un ürettiği büyük boyutlardaki veri düşünüldüğünde IoT için veri madenciliği çok yeni ve önemli bir araştırma konusudur.

Yapay sinir ağları karmaşık yapılarına, uzun öğrenme zamanlarına ve kolay anlaşılamayan sonuçlarına karşın sağlam yapısı, adaptif öğrenme yetenekleri, yüksek hata duyarlılığı ve paralel işleme yetenekleri sayesinde yüksek gürültü oranına sahip verilerde bile yüksek başarı vaat etmektedir. Bu da veri analizi için uygun bir araç olarak görülmelerini sağlamıştır. Özellikle ekonomi, mühendislik gibi sayısal verilerle işlemler yaptığımız alanlarda istatiksel yöntemler bu matematiksel fonksiyonlar için yetersiz kalmaktadır. Güçlü ve dağıtılmış işlemci mimarisi ile lineer olmayan çok değişkenli ilişkiler içeren veri setlerinde ANN'ler umut vericidir [72].

ANN'lere baktığımızda yapısal olarak giderek çeşitlendirilse de temel de ileri beslemeli ve geri beslemeli bağlantılar içeren ağlar görülmektedir. İnsan beyninin geri beslemeli bağlantılar içeren tekrarlayıcı bir sinir ağı olduğu düşünüldüğünde; RNN'ler birçok davranışı, dizi modellerini geleneksel makine öğrenmesi metotları aksine kolaylıkla öğrenebilir. Bu nedenle mühendislik uygulamalarında RNN'lere giderek artan bir ilgi mevcuttur. RNN'ler diğer ağlarla kıyaslandıklarında biyolojik olarak daha mantıklı ve hesaplama kabiliyetleri açısından da daha güçlüdürler.

Veri analizi için yapay sinir ağları yöntemi literatürde sıklıkla karşılaşabileceğimiz bir yöntemdir. Bu çalışmada zaman serilerinde yüksek performans gösterdiği birçok çalışmayla kanıtlanmış RNN, geleneksel eğitim algoritmalarından farklı olan, optimizasyon problemlerinde başarısı kanıtlanmış olan ABC algoritmasıyla ilk kez eğitilmiş ve mevcut çalışmalarla kıyaslanabilmek adına zaman serisi analizi için RMS, ortalama mutlak hata (Mean Absolute Error, MAE), simetrik oransal ortalama mutlak hata(Symmetric Mean Absolute Percentage Error, SMAPE) kriterleri; regresyon eğrileri ile ve zaman serisi analizi yanında sınıflandırma problemleri için karışıklık matrisleri ve ROC eğrileri kullanılarak performans değerlendirilmesi yapılmıştır. Sonuçlar göstermiştir ki ABC algoritmasıyla eğitilen RNN, nesnelerin internetinde zaman serisi analizleri için kullanılabilecek başarılı bir yöntemdir.

# 2. BÖLÜM

# YÖNTEM VE MATERYAL

Bu bölümde yapay sinir ağlarının temel yapısı, sınıflandırılması, tekrarlayıcı ağ yapıları, ağın eğitiminde kullanılacak olan yapay arı koloni algoritması ve performans karşılaştırmalarında kullanılan hata ölçüm metrikleri verilmiştir.

## 2.1. Yapay Sinir Ağları

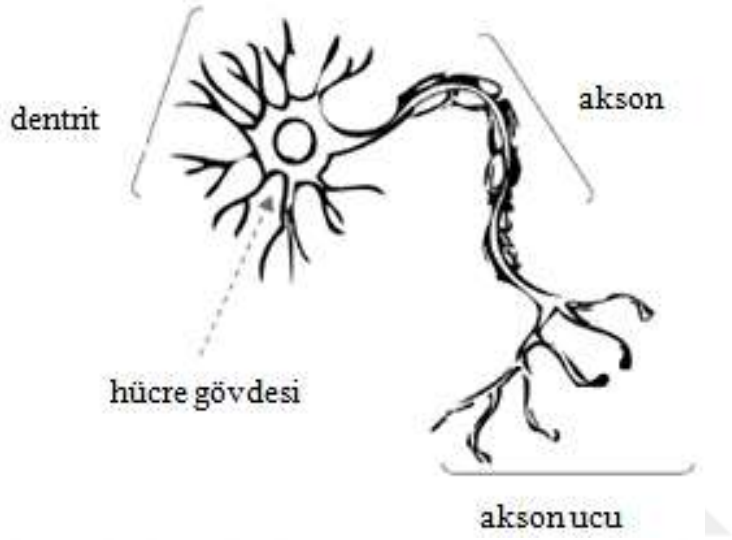
İnsan beyni günümüze kadar hala tam olarak keşfedilememiş en ilginç yapılardan biridir; ancak örneklerden öğrenme ve paralel işleme yeteneklerine sahip en hızlı ve en güçlü bilgi işleme aracı olduğu kesin olarak bilinmektedir [73]. İnsan beyninin bu özellikleri yapay sinir ağları kavramını ortaya çıkarmış ve ağlara örneklerden öğrenerek başka örneklere uygulanabilme yeteneğini kazandırmıştır.

İnsan beyninde yaklaşık olarak $10^{12}$ sinir hücresi (nöron) olduğu tahmin edilmektedir [74]. Bir sinir hücresi dentrit, akson, hücre gövdesi ve akson ucundan meydana gelmektedir. Bir sinir hücresinin akson ucuyla diğer hücrenin dentritinin teması sonucunda sinapslar oluşmaktadır. Dentritler bağlantılı nöronlardan sinyalleri almakta; hücre gövdesi dentritlerden gelen sinyalleri toplamakta, bu toplam belli bir aktivasyon eşiğini (threshold) aştığında bir sinyal oluşturmakta ve bu sinyal akson boyunca taşınarak sinapslar aracılığıyla diğer nöronlara aktarılmaktadır [75].

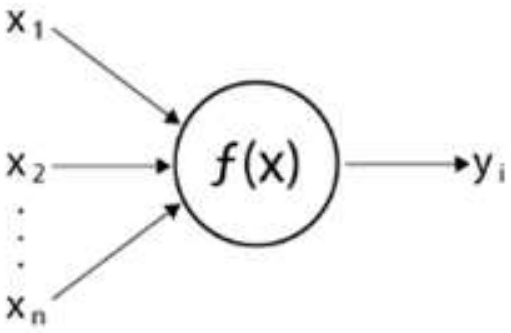
Yapay sinir ağları ise insan beynindeki bu nöronların bilgisayarlar üzerinde matematiksel olarak modellenmesiyle geliştirilmiştir [76]. Ağda insan beynindeki nöron hücresi yerine işlemci eleman ya da düğüm denilen elemanlar bulunmaktadır. Bu elemanlar girişler, ağırlıklar, toplam fonksiyonu, aktivasyon fonksiyonu ve çıkışlardan oluşmaktadır. İnsan beynindeki nöronlara benzer şekilde, ağdaki nöronlar belirli

ağırlıklarla sahip bağlantılar aracılığıyla birbirlerine bağlanır. Girişler belirli ağırlıklarla çarpılarak genellikle toplam fonksiyonu olan bir önişlem fonksiyonuna tabi tutulur daha sonra lineer olmayan bir aktivasyon fonksiyonundan geçirilerek çıkışa buradan da sinapslarla diğer elemanlara aktarılır [77].

Şekil 2.1'de biyolojik sinir hücresi [78], Şekil 2.2'de yapay sinir hücresi görülmektedir.




*Şekil 2.1.* Biyolojik Sinir Hücresi




*Şekil 2.2.* Yapay Sinir Hücresi

Tablo 2.1 biyolojik sinir sisteminde bulunan yapıların, yapay sinir ağında eleman olarak karşılığını göstermektedir [75].

*Tablo 2.1.* Biyolojik Sinir Sistemi Ve Yapay Sinir Ağı Elemanları

| **Biyolojik Sinir Sistemi** | **Yapay Sinir Ağı** |
|---|---|
| Nöron | İşlemci Eleman |
| Dentrit | Toplama Fonksiyonu |
| Hücre Gövdesi | Aktivasyon Fonksiyonu |
| Akson | Nöron Çıktısı |
| Sinaps | Ağırlık |

**Girdiler:** Girdiler dış dünyadan ya da diğer nöronlardan gelen verilerdir. Bu girdilerden gelen veriler biyolojik sinir hücrelerinde olduğu gibi nöron çekirdeğine gönderilerek burada toplanır.

**Ağırlıklar:** Yapay sinir hücresine gelen bütün bilgiler çıkış üzerinde aynı etkiye sahip değildir. Bu yüzden girdiler üzerinden çekirdeğe ulaşmadan önce etkileri oranında ağırlıklar içeren bağlantılardan geçirilirler. Bu ağırlıkların değerleri pozitif ya da negatif olabileceği gibi çıktı üzerinde etkisi olmayan girdiler için sıfır da olabilmektedir.

**Toplam Fonksiyonu (Birleştirme Fonksiyonu):**Ağırlıklarla çarpılan girdiler daha sonra hücrenin net girdisini hesaplamak için toplam fonksiyonundan geçirilirler.

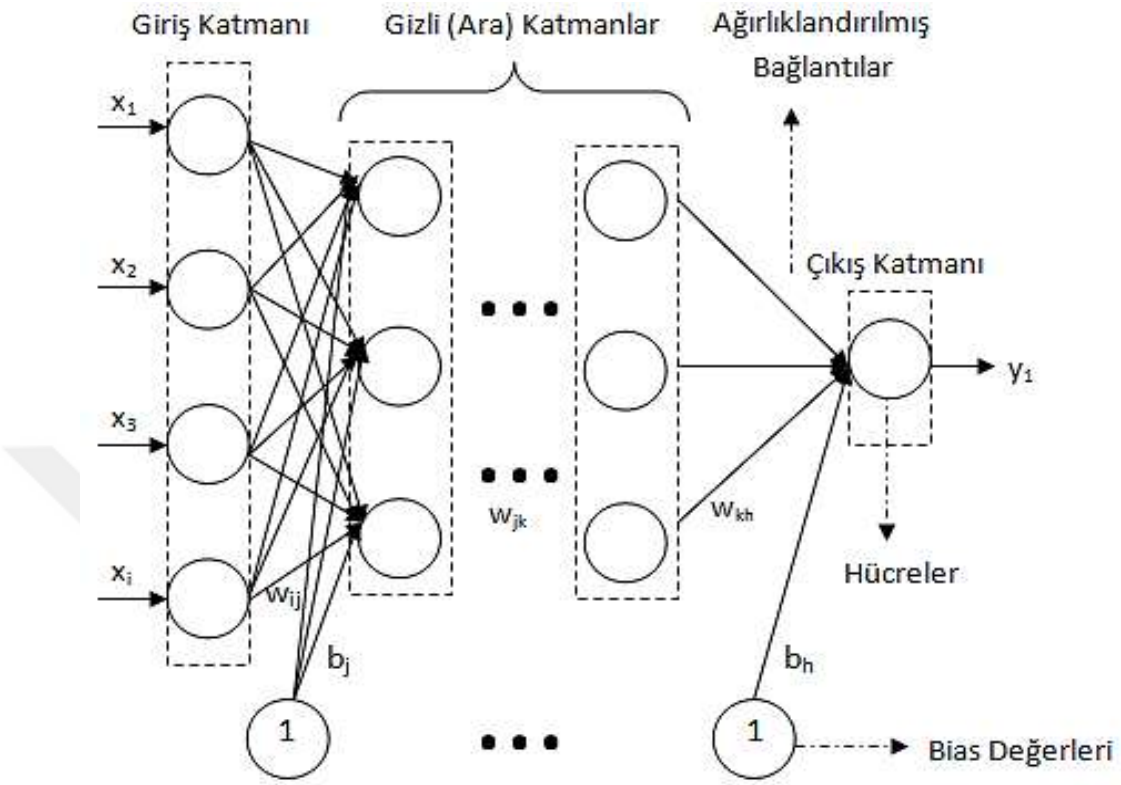
**Aktivasyon Fonksiyonu:**Bu fonksiyon hücrenin net girdisine karşılık üretilecek çıktıyı belirler. Ağın lineer olmayan ilişkileri çözümleyebilmesi bu fonksiyonların lineer olmama özelliğinden kaynaklanmaktadır. Çok katmanlı ağ yapılarında aktivasyon fonksiyonları arasında en çok tercih edilen sigmoid aktivasyon fonksiyonudur.

**Hücrenin Çıktısı:** Aktivasyon fonksiyonu sonucunda üretilen çıktı aynı zamanda hücrenin de çıktısıdır. Bu çıktı dış dünyaya gönderilebileceği gibi ağın için de tekrar da kullanılabilir.

### 2.1.1. Yapay Sinir Ağlarının Mimarisi

Yapay sinir ağları ise yapay sinir hücrelerinin birbirine bağlanmasıyla oluşan yapılardır. Bu ağlar giriş katmanı, ara katmanlar ve çıkış katmanı olarak üç ana katmanda incelenmektedir.

Bir ANN'nın temel yapısı ve bulundurduğu birimler Şekil 2.3'te görülmektedir.




*Şekil 2.3.* Yapay Sinir Ağı

**Giriş Katmanı:** Yapay sinir ağına dış dünyadan girdilerin geldiği katmandır. Bu katmanda dış dünyadan gelecek giriş sayısı kadar hücre bulunmaktadır ve genellikle girdiler herhangi bir işleme uğramadan alt katmanlara iletilmektedir.

**Gizli(Ara) Katman:** Giriş katmanından çıkan bilgiler bu katmana gelir. Ara katman sayısı ağdan ağa değişebilir. Bazı yapay sinir ağlarında ara katman bulunmadığı gibi bazı yapay sinir ağlarında ise çok sayıda ara katman bulunabilir. Ara katmanlardaki nöron sayıları giriş ve çıkış sayısından bağımsızdır. Birden fazla ara katman olan ağlarda ara katmanların kendi aralarındaki hücre sayıları da farklı olabilir. Ara katmanların ve bu katmanlardaki nöronların sayısının artması hesaplama karmaşıklığını ve süresini arttırmasına rağmen yapay sinir ağının daha karmaşık problemlerin çözümünde de kullanılabilmesini sağlar. Bu katman sayısının belirlenmesinde bazı varsayımlar olmakla birlikte henüz kesin bir bilgi bulunmamaktadır. Problemin amacına ve uygulama alanına göre en uygun katman ve nöron sayısı seçimi deneyerek belirlenmektedir.

**Çıkış Katmanı:** Ara katmanlardan gelen bilgileri işleyerek ağın çıktılarını üreten katmandır. Bu katmanda üretilen çıktılar dış dünyaya gönderilir. Geri beslemeli ağlarda bu katmanda üretilen çıktı tekrar kullanılarak ağın yeni ağırlık değerleri hesaplanır.

### 2.1.2. Yapay Sinir Ağlarının Çalışması

Bir ANN tasarımında öncelikle ağ mimarisinin seçilmesi, katman sayısı ve nöron sayısı gibi yapısal özelliklerinin belirlenmesi gerekir; sonraki aşamalarda ise işlemci elemanların kullandığı fonksiyonların belirlenmesi; öğrenme algoritması ve gerekli parametrelerin belirlenmesi gerekmektedir. Son aşamada ise eğitim, test ve istenirse doğrulama seti olarak veri seti ayrılmalıdır.

Bir ANN öğrenme aşamasında iken öncelikle çıktıları bilinen veri seti ağa verilir. Bu veri setine göre istenen çıktıyı sağlayabilecek şekilde girdiler için en uygun bağlantı ağırlıkları(w) bulunmaya çalışılır. Bu işlem önceden belirlenen performans kriterleri ya da maksimum iterasyon sayısı tamamlanıncaya kadar devam eder. İşlem tamamlandığında ağın eğitimi de tamamlanmış olur.

Bir yapay sinir ağı eğitilirken bulunan veri seti parçalara ayrılır bu ayırma işleminde kesin bir kriter olmamakla birlikte genellikle veri setinin %70'i eğitim , %15'i her zaman kullanılması zorunlu olmayan doğrulama ve kalan %15 ise test için ayrılır. Doğrulama seti kullanılmayacaksa veri %70 eğitim, %30 test ya da %85 eğitim,%15 test şeklinde ayrılabilir [79]. Doğrulama setinin kullanılmasının amacı ağın eğitimde çok yüksek performans gösterirken testte çok düşük performans göstermesini önlemektir. Yani eğitim setindeki veri öğrenmek yerine ezberlenirse, test setinde ağ çok başarısız olmaktadır. Bu probleme aşırı öğrenme(overfitting) denilmektedir. Ağ bu aşamaları tamamladıktan sonra eğitim tamamlanmış, belirlenen performans kriterleri içinde kaldığında ise ANN öğrenmiş olarak adlandırılır.

### 2.1.3. Yapay Sinir Ağlarında Kullanılan Aktivasyon Fonksiyonları

Doğrusal olmayan fonksiyonlar olan aktivasyon fonksiyonları,ANN'lere doğrusal olmama özelliği kazandırır. Hücrenin toplam net girdisi, giriş olarak fonksiyona verilir ve buna göre fonksiyona bağlı olarak ağın çıktısı ya da tekrar ağda kullanabilen hücre

çıktısı hesaplanır. Yapay sinir ağlarında genellikle hatanın türevi alınarak sıfırlanması yoluna gidildiğinden türevinin kolay hesaplanabileceği fonksiyonlar aktivasyon fonksiyonu olarak tercih edilir.

Yaygın olarak kullanılan aktivasyon fonksiyonları şunlardır [80]:

**Doğrusal (Lineer) Aktivasyon Fonksiyonu:** Doğrusal problemlerin çözümünde yararlanılan bu fonksiyonda net girdi sabit bir katsayıyla çarpılarak çıkış elde edilir. Eğer dinamik değişim aralığı [-1 1] aralığında ise bu fonksiyon simetrik parçalı doğrusal fonksiyon olarak adlandırılır.

**Adım (Step) Aktivasyon Fonksiyonu:** Net girdinin eşik değerine göre çıktı alınan bu fonksiyonda eğer net girdi eşik değerinden küçükse çıkış 0, aksi takdirde 1 olmaktadır.

**Sigmoid Aktivasyon Fonksiyonu:** Ağa doğrusal olmama özelliğini kazandırması, sürekli ve türevlenebilir olması sebebiyle yapay sinir ağı problemlerinde en çok tercih edilen fonksiyonlardır. Fonksiyonun dinamik değişim aralığı [0 1] dir.

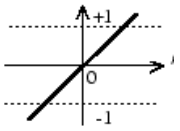
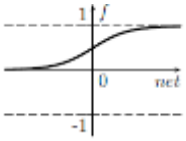
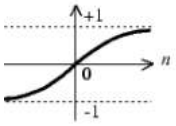
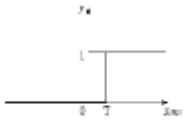
**Tanjant Hiperbolik Aktivasyon Fonksiyonu:** Sigmoid aktivasyon fonksiyonuna benzer özellikler gösteren fonksiyonun tek farkı dinamik değişim aralığının [-1 1] arasında olmasıdır.

**Eşik Değer Fonksiyonu:** Girdi eğer 0 dan küçük eşitse çıktı olarak 0 , 1 den büyük eşitse çıktı olarak 1, (0 1) aralığında olduğu durumlarda ise girdinin kendisini çıktı olarak veren fonksiyonlardır.

**Sinüs Fonksiyonu:** Eğer problem sinüs fonksiyon dağılımına uygun ise girişin sinüs değeri alınarak çıktının elde edildiği fonksiyonlardır.

Aktivasyon fonksiyonlarının grafikleri ve matematiksel ifadeleri Tablo 2.2’ de verilmiştir.

*Tablo 2.2.* Aktivasyon Fonksiyonları

| Doğrusal | Adım | Sigmoid | Hiperbolik Tanjant | Eşik Değer |
|---|---|---|---|---|
|  |  |  |  |  |
| f(net)=net | $fnet = \begin{cases} 0 \; net \leq 0 \\ 1 \; net > 0 \end{cases}$ | $f(\text{net}) = \frac{1}{1 + e^{-\text{net}}}$ | $f(\text{net}) = \frac{e^{net} + e^{-net}}{e^{net} - e^{-net}}$ | $f(\text{net}) = \begin{cases} 0 \; net \leq T \\ 1 \; net > T \end{cases}$ |

### 2.1.4. Yapay Sinir Ağlarının Sınıflandırılması

Yapay sinir ağları uzun süredir yoğun ilgi gören araştırma konularının merkezinde olmuştur [81]. Ortaya çıkmasından beri farklı ağ yapıları geliştirilmiştir. Yapay sinir ağlarındaki yapısal farklılıklar nöronların bağlantı şekilleri, katman sayıları ve bağlantı türlerinden; matematiksel farklılıklar ise kullanılan eğitim algoritmaları ve aktivasyon fonksiyonlarından kaynaklanmaktadır.

**Katman Sayısına Göre Sınıflandırma**

Ağların bulundurdukları katman sayısına göre sınıflandırılmasıdır. Tek katmanlı ve çok katmanlı olarak ikiye ayrılır [80]. Katman sayısı bulunurken giriş katmanı sayılmaz. Bu yüzden ara katman olmayan sadece giriş ve çıkış katmanından oluşan ağ yapısı tek katmanlı, en az bir ara katman içeren yapı ise çok katmanlı olarak kabul edilir.

**Tek katmanlı ağ:** Yapısında girdiler giriş katmanından geçerek çıkış katmanına gelir burada belirli fonksiyonlardan geçirilen girişler ağ çıktısını oluşturur.Ve (and), veya (or) gibi karmaşık olmayan problemlerin çözümünde yeterli olan bu ağlar xor problemi ve daha karmaşık problemler için yeterli olmamaktadır.

**Çok katmanlı ağ:** Tek katmanlı ağlar karmaşık işlemlerde yeterli olmadığı için ara katman ya da katmanlar içeren ağlar geliştirilmiştir. Bu yapıda girdiler gizli katmanlarda fonksiyonlardan geçirildikten sonra elde edilen çıkışlar bir sonraki gizli katmanın ya da son gizli katman için çıkış katmanının girdisini oluşturur. Son olarak yine çıkış katmanında yapılan işlemlerle ağın çıktısı oluşturulur.

**Katmanlar Arasındaki Bağlantı Şekline Göre Sınıflandırma**

Ağların bilgi akışının yönüne göre sınıflandırılmasıdır. İleri beslemeli ve geri beslemeli olmak üzere ikiye ayrılır [80].

**İleri beslemeli ağlar:** Her katmanın çıktısı kendisinden sonraki katmanlara gönderilir. Önceki katmanlara iletim yoktur.

**Geri beslemeli ağlar:** Ağ çıktısı ve katman çıktılarının kendisinden önceki katmanlara giriş olarak gönderilebilmesini sağlayan ağlardır. Yani çıkıştan girişe doğru da bağlantılar mevcuttur. Bu bağlantılar ağın çıkışında hem o andaki hem de geçmiş anlardaki bilgileri içinde bulundurmasını sağlar. Dinamik hafızaları olan bu ağlar çeşitli tiplerdeki zaman serileri tahminleri için oldukça başarılıdırlar [82].

İleri beslemeli ve geri beslemeli ağlarda katman sayılarındaki farklılıklar, bağlantıların farklı katmanlardan alınması ya da bağlantılara gecikme eklenmesi gibi faktörlerle bu ağlar çeşitlendirilmiştir. Bölüm 2.1.5'de sık olarak karşılaşabileceğimiz ve bu tez kapsamı çalışmasında incelenen ağ yapıları verilmiştir.

**Eğitim için Kullanılan Yönteme Göre Sınıflandırma**

Ağların eğitimde danışmanlı, danışmansız ve takviyeli öğrenme biçimi olarak üç yöntem vardır [80].

**Danışmanlı öğrenme:** Ağa hem girişler hem de bu girişlere karşı üretmesi gereken hedef çıkışlar gösterilir. Daha sonra girişlere karşı ağın ürettiği çıktı ile bu hedef değer arasındaki fark alınır. Bu farka göre ağın ağırlıkları güncellenir.

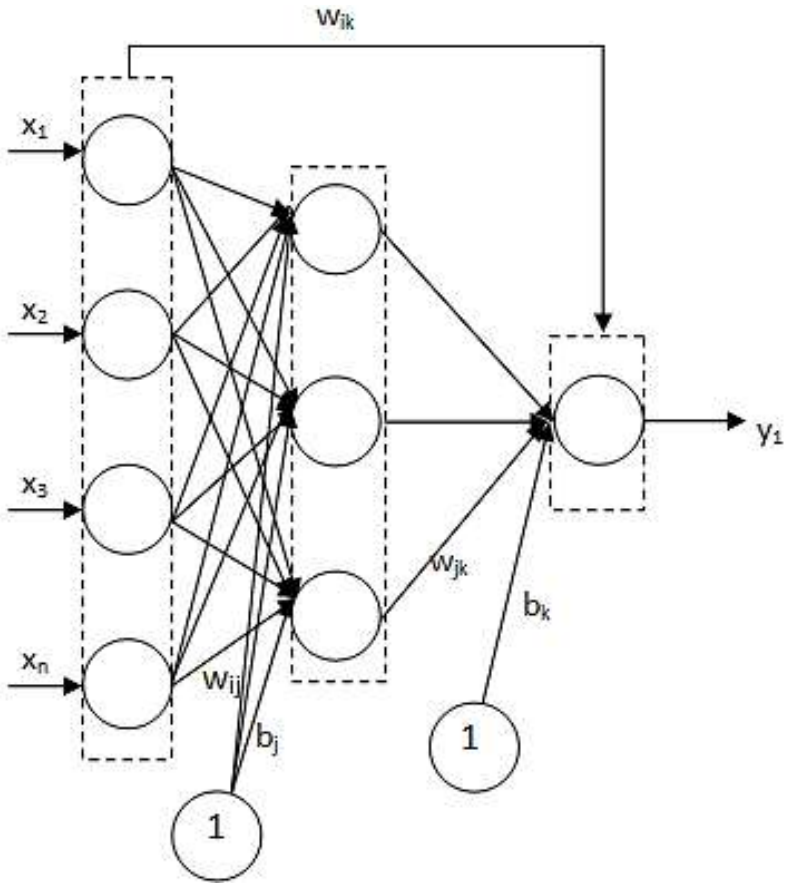
**Danışmansız Öğrenme:** Ağa sadece girişler gösterilir. Daha sonra ağ bu girişlerden üretilen çıkışlara göre girişleri kendi arasında sınıflandırabilecek modeli oluşturur. Ağ aynı niteliklere sahip girdileri sınıflandırabildiğinde öğrenme tamamlanır.

**Takviyeli (Destekleyici) Öğrenme:** Danışmansız öğrenmede olduğu gibi ağa çıkışlar gösterilmez; ancak ürettiği çıktığının iyi ya da kötü olduğuyla ilgili her iterasyon sonunda ağa bilgi verilir. Bu bilgiye göre ağırlıklar güncellenir.

Bu yöntemler uygulanırken kullanılan eğitim algoritmaları Bölüm 2.1.6'da anlatılmıştır.

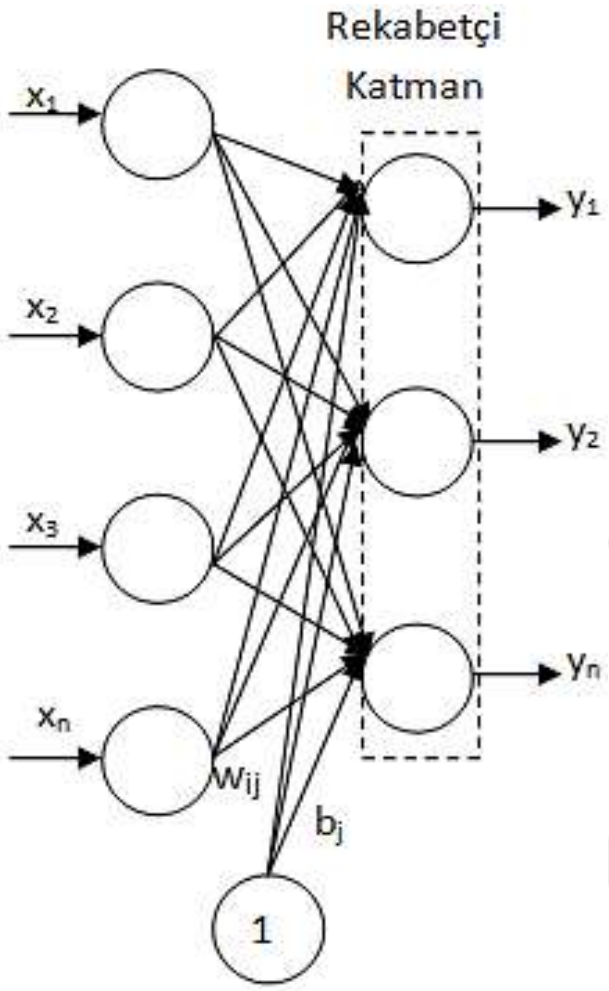
### 2.1.5. Sık Kullanılan Ağ Yapıları

**İleri Beslemeli Geri Yayılımlı Kademeli Sinir Ağı (Cascade Forward Neural Network, CFNN):**İleri beslemeli ağa çok benzeyen ağları, FFNN ağlarından farklı olarak kendinden sonraki her katmana bağlantı içerir yani giriş katmanının bütün katmanlara dolaysız yoldan bağlantısı vardır [83]. Çıkışın önceki tüm nöron tabakasıyla doğrudan bağlantısının olması ağın ilişkileri daha hızlı öğrenmesini sağlayabilir. Ağın temel yapısı Şekil 2.4'de görülmektedir.



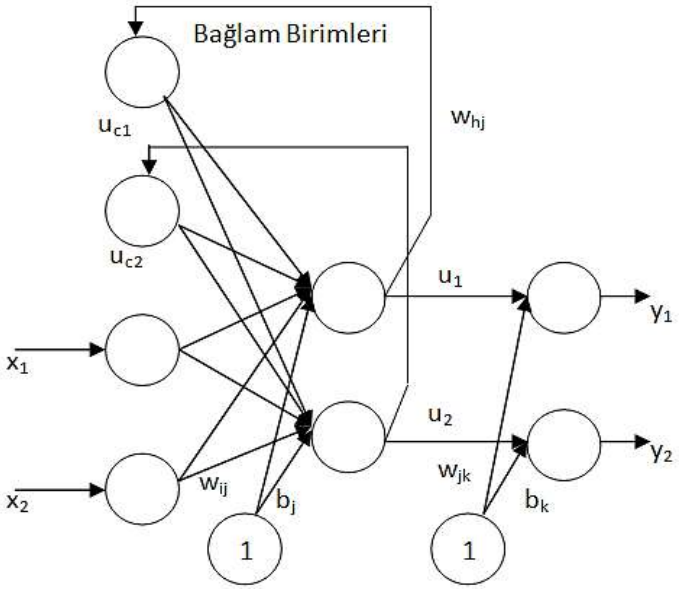

*Şekil 2.4.* Cascade Ağ Yapısı

**Rekabetçi Sinir Ağı (Competitive):** Rekabetçi öğrenme olarak bilinen competitive ağ yapısı danışmansız öğrenmeye dayanır. Hedef veriler ağa sunulmaz bu yüzden sınıflandırma problemleri için kullanılır. Ağdaki her nöronu farklı kalıplar üzerinde uzmanlaşmayı öğrenir böylece farklı girdi kümeleri için uzmanlaşmış olan nöronlar, özellik detektörlerine dönüşür. Bu yapı kohonen haritalama ve özellik vektörlerine dayanır [84]. Ağın temel yapısı Şekil 2.5' te görülmektedir [85].



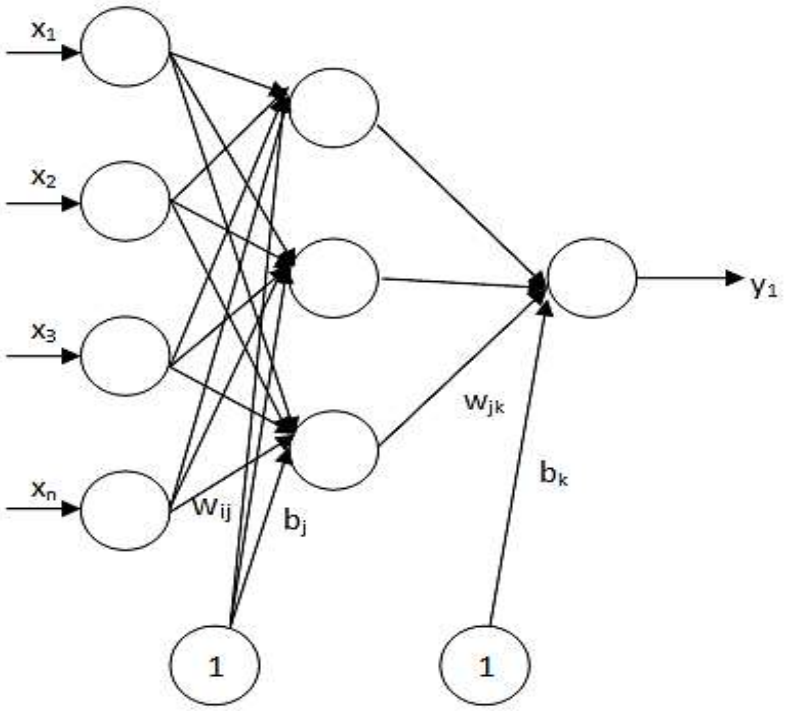

*Şekil 2.5.* Rekabetçi Ağ Yapısı

**Elman Sinir Ağı:** Geri besleme bağlantılarının ilk olarak kullanıldığı Elman ağları her gizli katmanının çıkışının kendi girişine tekrar gönderildiği bağlantılar içerir [86]. Tam dinamik türev hesaplamaları bu ağ yapısının sadece araştırma amacıyla kullanılmasına sebep olmuştur. Yerini RNN ve bu ağların özelleştirilmiş yapıları olan NAR ve NARX ağlarına bırakmıştır. Bir veya daha fazla gizli katmana sahip Elman ağları, gizli katmanlara yeterli sayıda nöron verildiğinde herhangi bir giriş çıkış ilişkisini keyfi olarak iyi öğrenebilir. Bu yüzden istenen düzeyde performansın sağlanabilmesi için tekrarlayan katmanlardaki nöron sayısının iyi ayarlanması çok önemlidir. Ağda kullanılan geri besleme bağlantıları ağın zamanla değişen girdiler için dinamik bir davranış göstermesini sağlamıştır [87; 88].Ağın temel yapısı Şekil 2.6'da görülmektedir.




*Şekil 2.6.* Elman Ağ Yapısı

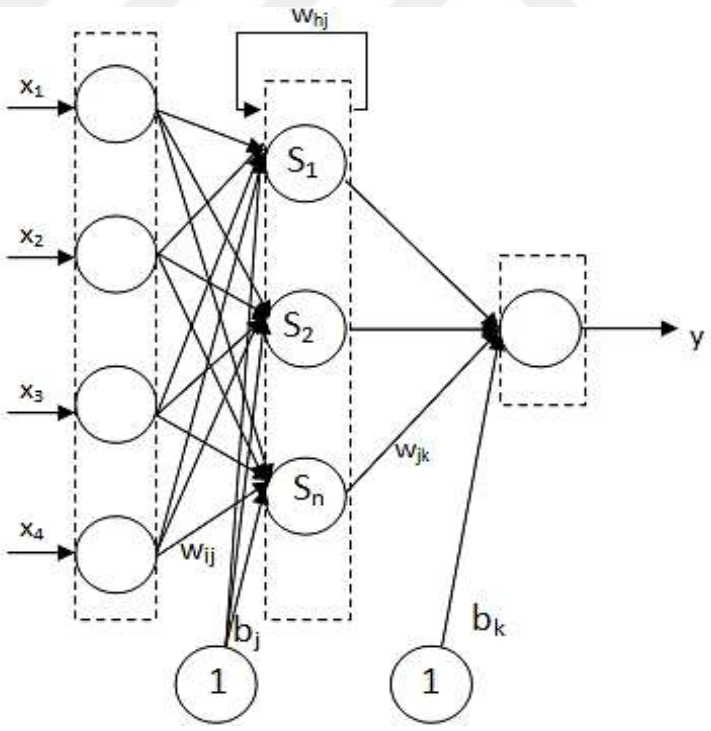
**İleri Beslemeli Geri Yayılımlı Sinir Ağı:** Ana sınıflandırmada da anlatılan ileri beslemeli ağlardır. Her katman sadece kendisinden bir sonraki katmana bağlantılar içerir. Hafızasız oldukları için statik ağ da denilmektedir [73]. Son katman ağın çıktısını üretir. Literatürde sıkça karşılaşılan ağ yapılarından biridir. Ağın temel yapısı Şekil 2.7'de görülmektedir.




*Şekil 2.7.* İleri Beslemeli Ağ Yapısı

İleri beslemeli ağ yapılarına farklı olarak zaman gecikmesi eklenerek dinamik özellik kazandırılabilir. Bu gecikme eğer her katmana verilirse dağıtılmış zaman gecikmeli sinir ağı (Distrubuted Time Delay Neural Network, DTDNN) ağ yapıları, sadece giriş katmanından sonraki ilk gizli katmana verilirse sınırlı dinamik tepki veren zaman gecikmeli sinir ağları (Time Delay Neural Network, TDNN) ağ yapıları elde edilir. Bu ağlar kazandıkları dinamik tepki verme özelliklerinden dolayı zaman serisi tahminleri için kullanılabilecek ağ yapılarıdır [89].

**Recurrent Sinir Ağları:** Çıkışın sadece o andaki girişe değil, aynı zamanda çıkışın ya da girişlerin önceki değerleriyle de ilgili olduğu ağ yapılarıdır, bu nedenle zaman serilerinin ileri elemanlarını tahmin etmekte yetersiz kalan FFNN'ler yerine tercih edilmektedirler [73].RNN'ler bu tez çalışmasının yöntemini oluşturmaktadır. Ağın temel yapısı Şekil 2.8'de görülmektedir.




*Şekil 2.8.* Tekrarlayıcı Ağ Yapısı

RNN lerde gizli katmandaki düğümlerin çıkışlarından girişlerine geri besleme bağlantıları mevcuttur. Bu şekilde bazı nöronlar sinaptik bağlantılar aracılığıyla diğer nöronlardan geri bildirim sinyali alırlar. Bu geri bildirimlerin, ağın öğrenme yeteneği ve performansı üzerinde derin bir etkisi vardır, bu etki sayesinde RNN'ler bir zaman serisi

probleminin dinamik özelliklerini başarılı bir şekilde modelleyebilir [90]. . Bu yüzden tahmin uygulamaları için sıkça tercih edilir. Toplam n tane çıkış nöronu ve m tane giriş elemanı olan tekrarlayıcı sinir ağını m+n tane giriş elemanı olan ileri beslemeli bir ağ gibi düşünülebilir.

Şekil 2.8'de gösterilen yapıda (t-1) anındaki çıkış katmana giriş olarak verilerek t anındaki çıkışı doğrudan etkiler. Yani her düğüm için o andaki (t) ve yakın geçmişteki olarak adlandırabileceğimiz (t-1) olarak iki giriş mevcuttur.

Şekil 2.8'de görülen yapı için çıktı Eşitlik (2.1)'de, gizli katman çıktısı Eşitlik (2.2)'de, çıktı katmanı net girdisi Eşitlik (2.3)'de ve ağın net çıktısı Eşitlik (2.4)'te verilmiştir.

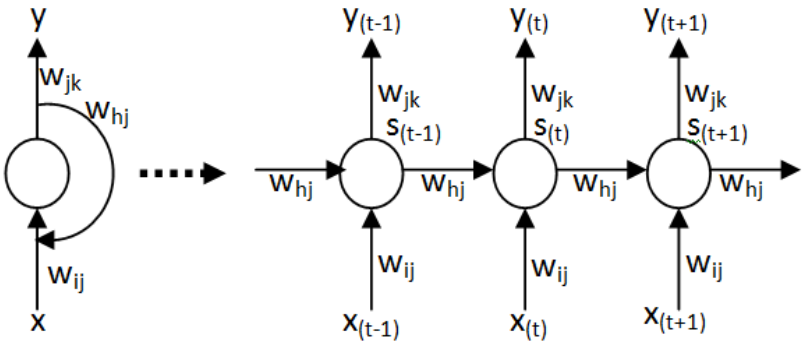
$$s_j = \sum_{i=1}^{N_i} (w_{ij} x_i + b_j) + \sum_{h=1}^{N_h} (w_{hj} h_j^{-1}) \tag{2.1}$$

$$h_j = f(s_j) \tag{2.2}$$

$$s_k = \sum_{j=1}^{N_h} (w_{jk} h_j + b_k) \tag{2.3}$$

$$y_k = f(s_k) \tag{2.4}$$

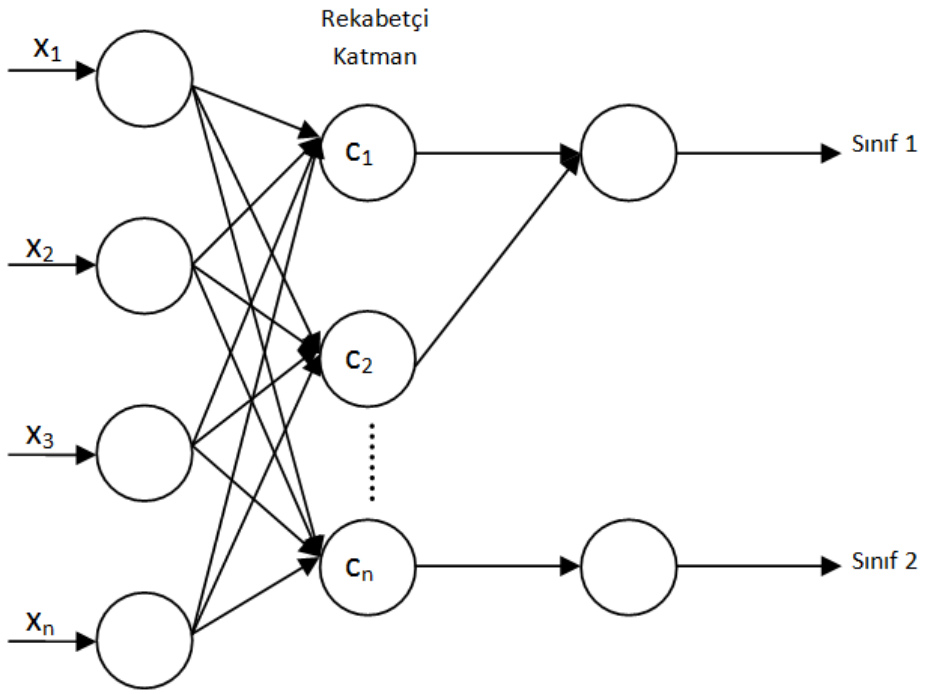
Şekil 2.9'da RNN ağının geri besleme bağlantıları açılarak FFNN ağına dönüştürülmüş şekli görülmektedir.




*Şekil 2.9.* RNN Ağ Yapısının Açık Şekli

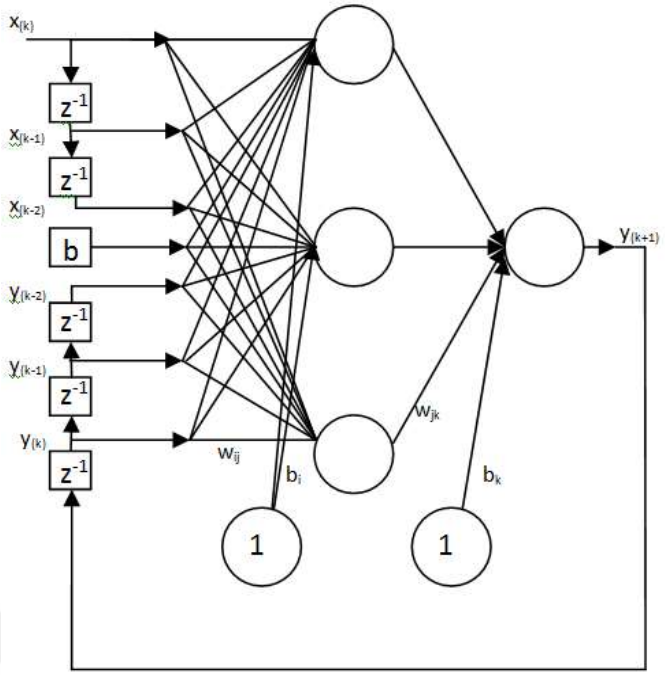
**LVQ(Learning VectorQuantatization):** Daha çok sınıflandırma amacıyla kullanılan bu ağ yapısının öğrenmesi temel çok katmanlı ağlara göre daha hızlı ve daha kolaydır.

Ağda denetimli öğrenme yöntemi kullanılır. Ağa çıktılar verilmez sadece elde edilen çıktıların doğruluğu veya yanlışlığı verilmektedir. Eğitimde amaç her iterasyonda girdi vektörüne yakın referans vektörünü bulmaktır [80]. Şekil 2.10'da ağın temel yapısı görülmektedir.



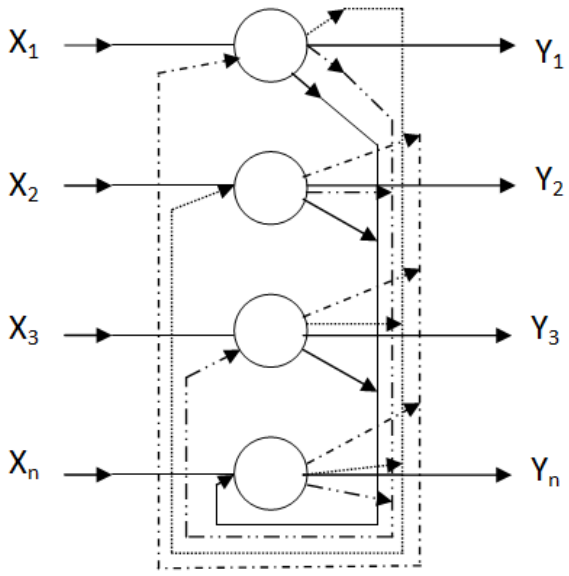

*Şekil 2.10.* LVQ Ağ Yapısı

**NARX:** Özelleştirilmiş bir RNN türü olan NARX, lineer olmayan zaman serilerinin modellenmesinde kullanılan dinamik yapıda bir ağ modelidir. Tam bağlantılı olmayan NARX ağları, sadece çıkış katmanından giriş katmanına bağlantı içerir; ancak performans açısından tam bağlantılı bir RNN ile aynı performansa sahiptir [91]. Diğer ağ yapılarından farklı olarak hem girişler hem geri besleme bağlantısı zaman gecikmeleri içermektedir. Zaman serileri tahmininde kullanılan NARX ağı, TDNN olarak düşünülebilir. NARX ağında en az bir kademe ileriden geri besleme ile önceki katmana gelen bir gecikme hattı bulunur. Yapılan çalışmalar NARX ağ yapısının benzer yapılarda olan Elman ve TDNN' e göre çok daha iyi sonuçlar ürettiğini göstermektedir [92]. Şekil 2.11'de ağın temel yapısı görülmektedir.



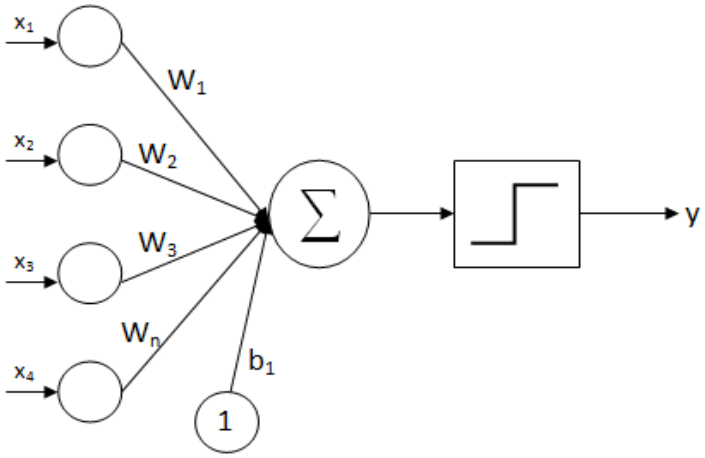

*Şekil 2.11.* NARX Ağ Yapısı

**Hopfield:** Tek katmanlı bir ağ yapısı olan hopfield ağ yapısında bu katmandaki her bir düğümün çıktısı kendisi dışındaki bütün düğümlere giriş olarak verilir. Bu katmandaki düğümler gizli katmandaki düğümler olarak görüldüğü gibi aynı zamanda hem giriş hem de çıkış düğümleridir. Her bir nöron aktivasyon değeri olarak -1 ve 1 ile ifade edilen binarydeğerler alır [93].Hopfield ağ yapısı, RNN'lerin temeli olarak görülmektedir [94]. Ağın temel yapısı Şekil 2.12'de görülmektedir.



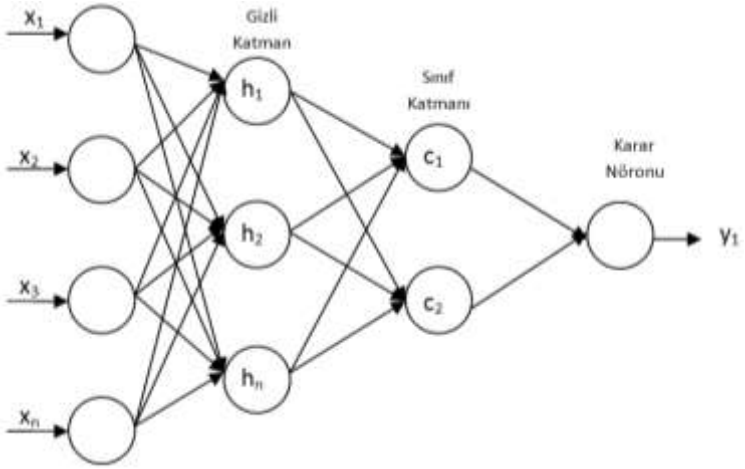

*Şekil 2.12.* Hopfield Ağ Yapısı

**Algılayıcı (Perceptron):**Yapay sinir ağlarının temelini oluşturan bu model, en eski ağ modellerinden biridir, gizli katman içermez. Oldukça basit olan perceptron ve (and) ve veya (or) problemleri için yeterlidir. Algılayıcının temel yapısı Şekil 2.13'de görülmektedir.



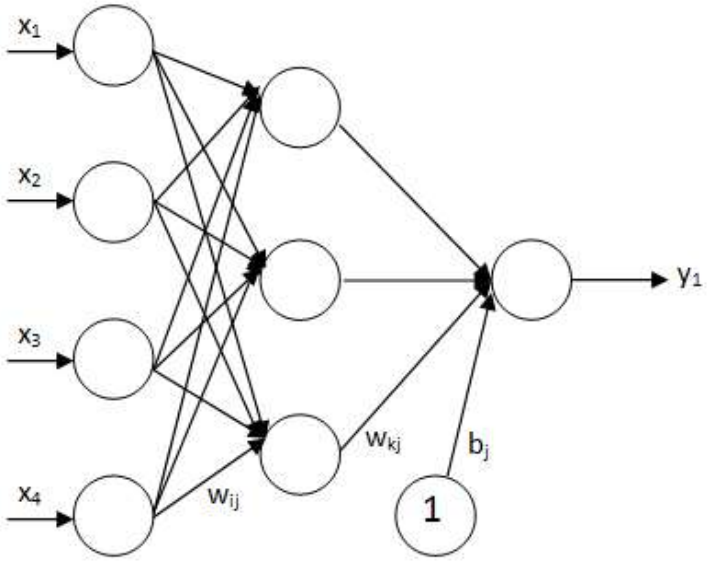

*Şekil 2.13.* Perceptron Ağ Yapısı

**Olasılıksal Ağ Yapısı (Probabilistic):** Sınıflandırma ve örüntü tanıma için özelleştirilmiş ileri beslemeli bir ağ yapısıdır. Şekil 2.14'te görülen ağ yapısı gelen giriş bilgilerini iki sınıfa ayırmaktadır, daha sonra genellikle sigmoid olmayan bir üstel aktivasyon fonksiyonundan geçirerek çıkış değeri üretmektedir [95].



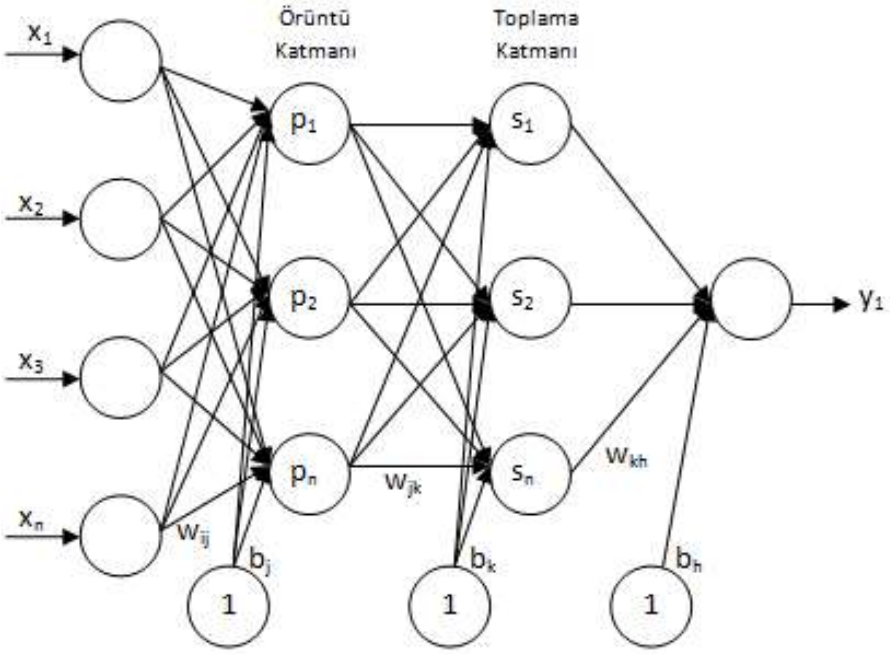

*Şekil 2.14.* Olasılıksal Ağ Yapısı

**Radyal Temelli Ağlar (Radial Basis Function, RBF):**Gaussian fonksiyonunun kullanıldığı ileri beslemeli ağlardır. Radyal temelli ağların yerel minimuma takılma sorunu bulunmamaktadır [3]. Ağın temel yapısı Şekil 2.15'de görülmektedir.




*Şekil 2.15.* RBF Ağ Yapısı

**Genelleştirilmiş Regresyon Ağları (Generalized Regression)**: Bu ağ yapısında birçok yapıdan farklı olarak iteratif öğrenme kullanılmaz. Sürekli verilerde uygun olan bu yapı Gaussian fonksiyonu kullandığı için RBF ağının bir benzeridir. Giriş çıkış vektörleri arasındaki rastgele fonksiyonu eğitim verisinden rahatlıkla bulabilen ağ, veri miktarı arttıkça yaklaşık olarak %100 lük öğrenme oranı yakalamaktadır [96]. Geri beslemeye ihtiyaç duymayan bu ağ yapıları tahmin, sınıflandırma ya da regresyon amacıyla kullanılabilir. Şekil 2.16'da ağın temel yapısı görülmektedir.




*Şekil 2.16.* Genelleştirlmiş Regresyon Ağ Yapısı

### 2.1.6. Yapay Sinir Ağlarında Kullanılan Eğitim Algoritmaları

Öğrenme algoritması ağın performansında doğrudan etkiye sahip olan parametrelerden biridir. ANN'lerden istenen performansın sağlanabilmesi için nöronlar uygun şekilde bağlandıktan sonra bağlantıların en uygun değerlerle ağırlıklandırılması gerekir.

Hedef veriyle ağın çıktısını yakınsamak amacıyla ağırlıkların sürekli güncellenmesine ağın eğitilmesi, ağın istenen performansa ulaşmasına ise ağın öğrenmesi denir [4].

Bilinmeyen parametrelerin belirli koşullar altında belirli sınırlamaları sağlayacak şekilde ayarlanmasına optimizasyon problemi adı verilmektedir. Yapay sinir ağları için en uygun ağrılıkların belirlenmesi kısıtsız optimizasyon problemi, eğitim algoritmaları da ağırlıkları güncelleyerek gerçek değere ulaşmayı hedefleyen optimizasyon algoritmalarıdır.

ANN'de kullanılan öğrenme algoritmaları insan beyninin de özelliklerinden olan sezgisel gücü içerisinde barındırdığından, klasik bilgisayar algoritmalarından farklıdır ve bu algoritmalar sayesinde ANN'lar çok farklı alanlarda, çok çeşitli uygulamalarda kullanılabilmektedir [97]. Hangi algoritmanın hangi ağda daha iyi sonuç vereceği kesin olarak tahmin edilemez.Yaygın olarak kullanılan öğrenme algoritmalarının kullandıkları tekniğe göre sınıflandırılması Tablo 2.3'de verilmiştir [79].

*Tablo 2.3.* Öğrenme Algoritmalarının Sınıflandırılması

| **Algoritmanın Kullandığı Teknik** | **Algoritmalar** |
|---|---|
| **Gradyan Düşüm (GDA)** | Gradientdescentbackpropagation (GD) |
| | Gradientdescentwithadaptivelearning rate backpropagation (GDA) |
| | Gradientdescentwith momentum backpropagation (GDM) |
| | Gradientdescentwith momentum andadaptivelearning rate backpropagation (GDX) |
| **Esnek Geri Yayılım (RP)** | Resilientbackpropagation(RP) |
| **Eşlenik Gradyan (CGA)** | Conjugategradientbackpropagationwithfletcher-reevesrestarts (CGF) |
| | Conjugategradientbackpropagationwithpolak/ribiérerestarts (CGP) |
| | Conjugategradientwithpowell/bealerestarts (CGB) |
| | Scale conjugategradientbackpropagation (SCG) |
| **Quasi-Newton (QNA)** | Bfgsquasi-newtonbackpropagation (BFGS) |
| | One-step secantbackpropagation (OSS) |
| **Levenberg-Marquardt (LM)** | Levenberg-marquardtbackpropagation (LM) |
| | Bayesianregulaziton (BR) |

Bu algoritmaların anlaşılabilmesi için yararlanılan Gradyan vektörü, Jakobyen matrisi ve Hessian Matrisi bilinmelidir. Üç kavramın da temelinde ağın tasarımı sırasında belirlenen performans fonksiyonun ağırlığa göre değişimi temel alınmaktadır, bu şekilde ağın gerçek veri setini modellemede gösterdiği uyum belirlenir [79].

**Gradyan Vektörü:** Hata fonksiyonunun bağlantı ağırlıklarına göre birinci dereceden tüm kısmi türevlerinin bir sütun vektörü şeklinde düzenlemesiyle oluşan vektöre gradyanvektörü adı verilir. $\nabla f$ ya da grad$f$ şeklinde gösterilebilir. Vektörel gösterimi Eşitlik (2.5)'de verilmiştir.

$$\nabla \mathrm{f} = \begin{bmatrix} \frac{\partial_{\mathrm{E}}}{\partial_{\mathrm{w_{11}}}} \\ \vdots \\ \frac{\partial_{\mathrm{E}}}{\partial_{\mathrm{w_{ij}}}} \end{bmatrix} \tag{2.5}$$

**Jakobyen matrisi:** Jakobyen matrisi gradyanvektöründen faklı olarak hata fonksiyonundan değil, tüm giriş örneklerini kullanarak oluşturduğu hata vektöründen yararlanır. Hata vektörünün tüm elemanlarının bağlantı ağırlıklarına göre birinci dereceden kısmi türevlerini içerir. Jakobyen matrisi vektörel iken, gradyan vektörü skalerdir.

Hata vektörü Eşitlik (2.6)'da Jakobyen matrisi ise Eşitlik (2.7)'de verilmiştir.

$$e = \begin{bmatrix} e_{11} \\ \vdots \\ e_{1\mathrm{n}} \\ \vdots \\ e_{\mathrm{p1}} \\ \vdots \\ e_{\mathrm{pn}} \end{bmatrix} \tag{2.6}$$

$$J = \begin{bmatrix} \frac{\partial_{\mathrm{e_{11}}}}{\partial_{\mathrm{w_1}}} & \cdots & \frac{\partial_{\mathrm{e_{11}}}}{\partial_{\mathrm{w_N}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial_{\mathrm{e_{pn}}}}{\partial_{\mathrm{w_1}}} & \cdots & \frac{\partial_{\mathrm{e_{pn}}}}{\partial_{\mathrm{w_N}}} \end{bmatrix} \tag{2.7}$$

**Hessian Matrisi:** Hata fonksiyonununtüm bağlantı ağırlıklarına göre ikinci dereceden kısmi türevleri Hessian matrisini verir. Matris simetriktir ve klasik optimizasyon problemleri için önemlidir ancak hesaplama yükü fazladır. Hessian matrisi Eşitlik (2.8)'de verilmiştir.

$$H = \begin{bmatrix} \frac{\partial^2 \mathrm{E}}{\partial {w_{11}}^2} & \cdots & \frac{\partial^2 \mathrm{E}}{\partial w_{11} w_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathrm{E}}{\partial w_{11} w_N} & \cdots & \frac{\partial^2 \mathrm{E}}{\partial {w_N}^2} \end{bmatrix} \quad (2.8)$$

**Gradyan Azaltma Yöntemine Dayalı Algoritmalar**

Bu algoritmalarda hata fonksiyonunu minimize edecek ağırlık değerlerinin hesaplanabilmesi için gradyanvektöründen yararlanılır. Standart gradyan düşüm algoritmasına momentum terimi eklenmesiyle GDM, adaptif öğrenme oranı eklenmesiyle GDA ve her ikisinin birlikte kullanılmasıyla GDX algoritmaları geliştirilmiştir.

Standart gradyan düşüm algoritmasının çalışma mantığı şu şekildedir [98]:

**1.adım:**Bir başlangıç çözümü alınarak ($x^0$) iterasyon sayacını sıfırlanır (t=0) ve algoritmanın çalışmasını bitirecek bir ya da daha fazla durdurma kriteri seçilir (Ɛ).

**2.adım:**$x^t$ noktasında gradyan $g^t$ ve $\|g^t\|$ hesaplanır eğer çözüm istenen tolerans değerinden küçük ($\|g^t\| < Ɛ$) ise araştırma durdurulur aksi takdirde devam edilir.

**3.adım:**Araştırma yönü $d^t = -g^t$ denkleminden yararlınalarak belirlenir.

**4.adım**:$\alpha^t$, $f(x^t + \alpha^t d^t)$ ifadesini minimize edecek adım büyüklüğü olmak üzere $x^{t+1} = x^t + \alpha^t d^t$ denkleminden yararlanarak yeni nokta yani yeni olası çözüm hesaplanır.

**5.adım:**t=t+1 yapılarak 2.adıma dönülür.

**GD:** Standart gradyan düşüm algoritmasıdır. Ağırlık ve vektörler performans fonksiyonunun negatif gradyanı yönünde güncellenir.Elde edilen bu vektör öğrenme katsayısı ile çarpılarak ağırlıkların değişimi bulunur.

Ağırlıkların güncellenmesi için kullanılan denklem Eşitlik (2.9)'da, yeni ağırlıkların bulunmasına ilişkin denklem Eşitlik (2.10)'da verilmiştir.

$$\Delta w_{ij} = -\alpha g_{ij} = -\alpha \frac{\partial E}{\partial w_{ij}} \quad (2.9)$$

$$w_{k+1} = w_k \Delta w_{ij} \quad (2.10)$$

Standart gradyan düşüm algoritmaları basit yapılarına rağmen hesaplama verimliliği ve sağladığı yüksek performans nedeniyle ANN'larda en çok tercih edilen algoritmalardan biri olmuştur; ancak uygulamalarda yakınsama hızları çok yavaştır bu nedenle momentum terimi ve adaptif öğrenme oranı parametreleri eklenerek yakınsama hızı artırılmak istenmiştir [99].

**GDA:** Standart eğim düşüm algoritmasında öğrenme oranı her zaman sabittir. Algoritmanın performansı da bu öğrenme oranıyla doğrudan ilişkilidir. Eğer bu oran çok büyük seçilirse sonuç belli bir noktada salınım yapabilir, yerel minimumu atlayabilir ve algoritma ağın kararsız hale gelmesine neden olur. Çok küçük seçildiği durumlarda ise ideal davranışa yaklaşırken, yakınsama hızı azalacak ve hesaplama yükü artacaktır. Sabit olan bu oranın seçiminde ağın eğitiminden önce belirlenerek uygulanabilecek pratik bir yöntem mevcut değildir. Algoritma, performans yüzeyinde gezdiği sürece bu optimum ağırlık değişir. Bu sebeplerden dolayı sabit olan oran yerine eğitim süresince değişebilen adaptif bir oran seçilebilmesi performansı artıracaktır.

Bu algoritma için, başlangıçta ağ çıktısı ve hata değerleri hesaplandıktan sonra her bir iterasyonda o anki öğrenme oranına göre ağırlık ve bias değerleri güncellenerek yeni çıktılar ve hatalar bulunur. Eğer bulunan yeni bias ve ağırlık değerleri için performans eskisinden kötüyse eski değerlerle devam edilir ve öğrenme oranı azaltılır. Yeni performansın eskisinden iyi olduğu durumlarda ise yeni ağırlık ve bias değerleri saklanır ve öğrenme oranı artırılarak devam edilir. Salınıma neden olmayan en büyük adım ($\alpha$) seçilmeye çalışılarak yakınsama hızı artırılır [79].

**GDM:** Standart algoritmaya çözüm için momentum ($\mu$) terimi eklenmiştir. Bu terim ağın hata yüzeyindeki salınımları önlemesinin, yerel mimimumu bulmasına yardımcı olmasının yanı sıra hata yüzeyindeki son durumları yakalamaya da yarar [100]. Ancak momentum katsayısı seçiminin başka kesin bir çözümü olmayan başka bir optimizasyon problemi olması ve engellenemeyen bir şekilde yakınsamayı yavaşlatması bu algoritmanın dezavantajlarıdır [99; 101].

Ağırlıkların güncellenmesi için kullanılan Eşitlik (2.11)'de, yeni ağırlıkların bulunmasına ilişkin Eşitlik (2.12)'de verilmiştir.

$$\Delta w_k = \Delta w_k - \Delta w_{k-1} \tag{2.11}$$

$$w_{k+1} = \mu \Delta w_k - \alpha g_k \tag{2.12}$$

**GDX:** Standart eğim düşüm algoritmasına momentum ve adaptif öğrenme oranı terimlerinin birlikte eklenerek geliştirilen gradyan düşüm algoritmasıdır [79]. Geri yayılım ile ağırlık ve bias değerleri hesaplandıktan sonra performans için öğrenme oranı ve momentum terimleri, ağırlık ve bias değerleri için yalnızca momentum terimi eklenerek bu parametreler için yeni değerler elde edilir. Ağırlıkların güncellenmesi sırasında kullanılan Eşitlik (2.13)'de verilmiştir.

$$\Delta w_{ij(n)} = \mu \Delta w_{ij(n-1)} - \alpha\mu \frac{\partial E(n)}{\partial w_{ij}(n-1)} \tag{2.13}$$

α ve μ parametrelerinin seçimi uygunlaştıkça kararlılık ve yakınsama hızı artacaktır [100].

**Eşlenik Gradyan Algoritmaları**

Ağırlık ve bias değerlerini en hızlı iniş yönünde güncelleyen GD algoritmalarına göre daha hızlı yakınsamayı sağlamak için ardışık olan güncelleme vektörlerinin birbirine dik olması amaçlanır [102]. Hızlı yakınsama ve düşük bellek gereksinimi sayesinde GDA'lara göre daha başarılı algoritmalardır, matris depolama ihtiyacı olmadığından çok değişkenli fonksiyonlar için de etkilidir; ancak geniş ağ yapılarında kararsızlığa meyillidir [79]. Eşlenik gradyan algoritmalarının çalışma mantığı şu şekildedir:

**1.adım:**Bir başlangıç çözümü alınarak ($x^0$) iterasyon sayacını sıfırlanır (t=0); bir yakınsama parametresi seçilir (Ɛ) ve araştırma yönü $d^0 = -g^0$olarak alınır eğer $\|g^0\| < \varepsilon$ ise araştırma durdurulur; aksi takdirde 4.adıma gidilir.

**2.adım:**$x^t$ noktasında gradyan$g^t$ ve $\|g^t\|$'yi hesaplanır eğer $\|g^0\| < \varepsilon$ ise araştırmayı durdurulur; aksi takdirde devam edilir.

**3.adım:**Eşlenik gradyan yönü $\beta^t$ kullanılan CGD algoritmasına göre elde edilen değer olmak üzere $d^t = -g^t + \beta^t d^{t-1}$denkleminden yararlanarak hesaplanır.

**4.adım:**$\alpha^t$, $f(x^t + \alpha^t d^t)$ ifadesini minimize edecek adım büyüklüğü olmak üzere $x^{t+1} = x^t + \alpha^t d^t$ denkleminden yararlanarak yeni nokta yani yeni çözüm hesaplanır.

**5.adım:**t=t+1 olmak üzere ikinci adıma dönülür. [98]

Burada eşlenik gradyan yönü ($\beta_k$) hesaplanırken kullanılan tekniğe göre CGF ve CGP algoritmaları, sıfırlama noktasında kullandığı teknikten dolayı CGB ve kullandığı çizgisel arama işlemi yönteminden dolayı SCG algoritmaları ortaya çıkmıştır.

**CGF:** Bu algoritma Eşitlik (2.14)'de görüldüğü üzere gradyan yönünü mevcut gradyanın karesinin, bir önceki gradyanın karesine oranı şeklinde hesaplar.

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}} \tag{2.14}$$

Adaptif öğrenme oranını içeren GD algoritmalarına göre çok daha hızlıdır ayrıca probleme göre değişmekle birlikte RP algoritmasından da hızlı olabilmektedir. Hafıza gereksinimi ve hesaplama yükü daha basit algoritmalar göre biraz daha fazladır; ancak geniş yapıdaki ağlar için uygundur [103].

**CGP:** Bu algoritma Eşitlik (2.15)'de görüldüğü üzere gradyent yönünü bir önceki gradyandaki değişim ile mevcut gradyanın çarpımının bir önceki gradyanın karesine oranı şeklinde hesaplar [103] .

$$\beta_k = \frac{g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}} \tag{2.15}$$

**CGB:** Bütün CG algoritmalarının arama yönü periyodik olarak negatif gradyana eşitlenir. Standart olarak sıfırlama (reset)noktası ağın parametre sayısının iterasyon sayısına eşit olduğu andır. Ancak CGB algoritmalarında farklı olarak mevcut diklik ile bir önceki diklik arasındaki oran kullanılır. Yeni sıfırlama noktası Eşitlik (2.16)'daki şart sağlandığında belirlenir [104].

$$|g_{k-1}^T g_k| \geq 0.2[\![g_k]\!]^T \tag{2.16}$$

**SCP:** Diğer eşlenik gradyan algoritmaları CGF, CGB ve CGP algoritmalarında çizgisel arama işlemi her iterasyonda tekrar gerçekleştirilir. Bu nedenle bu algoritmalarda sayısal işlem yoğunluğu fazladır. Bu yoğunluğu azaltmak amacıyla CGD algoritmasıyla bir sonraki bölümde anlatılacak olan LM algoritmasında kullanılan modele bağlı bölge yaklaşımı birlikte kullanılmaktadır [104].

**Esnek Geri Yayılım Algoritmaları:**

**RP:** Çok katmanlı algılayıcılar tipik olarak gizli katmanlarında sigmoid aktivasyon fonksiyonunu kullanırlar. Sonsuz girişe karşı sınırlı bir cevap sağlama özelliği olan bu fonksiyonun girişi büyüdükçe, eğimi sıfıra doğru gitmektedir. Bu da standart GD algoritmaları için gradyanın çok küçük yerel minimum değerine yakınlaşıldığı varsayımıyla optimum sonuca çok uzakta olunmasına rağmen ağırlık ve bias değerlerinin düşük değerlerle güncellenmesine sebep olur. Türev büyüklüğünden kaynaklanan bu zararlı etkileri yok etmek için bu algoritmada türevin sadece işaretinden yararlanılarak ağırlıklar ve bias değerlerinin güncellenmesi için yön kararı verilir. Değişim miktarı için ayrı bir güncelleme değeri kullanılır [105]. Algoritmanın genel akışı şu şekildedir:

**1.adım:**Bir başlangıç çözümü alınarak ($x^0$) iterasyon sayacını sıfırlanır (t=0); bir yakınsama parametresi seçilir (Ɛ) ve araştırma yönü $d^0 = -g^0$olarak alınır eğer $\|g^0\| < Ɛ$ise araştırmayı durdurulur; aksi takdirde 4.adıma gidilir

**2.adım:**İki ardışık adım arasında, türev işaret değiştirmiyorsa $f'(w(t)) \times f'(w(t)) > 0$ise yeni adım boyutu $d(t) = d(t-1) \times n^+$ eşitliğini sağlayacak şekilde artırılır, işaret değiştiriyorsa türev işaret değiştiriyorsa $f'(w(t)) \times f'(w(t)) < 0$ise yeni adım boyutu $d(t) = d(t-1) \times n^-$ eşitliğini sağlayacak şekilde azaltılır.

**3.adım:**Ağırlıklardaki değişim $\Delta w_t = -sign(g^t \times d^t)$ şeklinde güncellenir.

**4.adım:**t=t+1 alarak ikinci adıma git [102].

Eğer iki ardışık adım arasında türevin işareti değişiyorsa algoritma bir yerel minimum atlandığı varsayımıyla adım boyutunu $n^-$ ile çarparak azaltır, işaret değişmiyorsa yerel minimuma yaklaşmak için $n^+$ ile çarparak artırır. Ağırlıklar osilasyona giderse ağırlık

değişimi azaltılarak düzeltilmeye çalışılır. Eğer ağırlıklar birkaç iterasyon süresince aynı yönde değişime devam ederse ağırlık değişim büyüklüğü artırılır [102].

**Quasi-Newton Algoritmaları**

Hızlı yakınsama yönünden CGA'ya benzeyen algoritmalardır; ancak Hessian matrisi içermelerinden dolayı hesaplama yükü daha ağır ve karmaşıktır. Algoritmanın genel akışı şu şekildedir:

**1.adım:**Bir başlangıç çözümü al ($x^0$) ve iterasyon sayacını sıfırla(t=0); durdurma kriteri için bir tolerans seç (Ɛ).

**2.adım:**$x^t$ noktasında gradyan$g^t$ ve $\|g^t\|$'yi hesapla eğer $\|g^t\| < Ɛ$ ise araştırmayı durdur; aksi takdirde devam et.

**3.adım:**Hessian matrisi $H(x^t)$' yi hesapla

**4.adım**:$\alpha^t$, $f(x^t + \alpha^t d^t)$ ifadesini minimize edecek adım büyüklüğü olmak üzere $x^{t+1} = x^t + \alpha^t d^t$ denkleminden yararlanarak yeni noktayı hesapla.

**5.adım:**t=t+1 alarak ikinci adıma git [98].

**BFGS(BFG):** BFGS, Quasi-Newton yöntemini kullanarak bias ve ağırlıkları güncelleyen algoritmadır. Eşlenik gradyan düşüm yöntemini kullanan algoritmalara (CG, CGB, CGF, CGP) daha hızlı yakınsama sağlayan bir alternatif olarak geliştirilmiştir. Yüksek hesaplama ve bellek ihtiyacından dolayı geniş yapılı ağlar için önerilmemektedir [79]. CGD algoritmalarına göre çok daha yüksek hesaplama gücü isteyen bu algoritma n tane bias ve ağırlık hücresine sahip bir ağ için n×n boyutunda Hessian matrisi içerir. Ağ yapısının geniş olduğu yapılarda CGD ve RP algoritmaları daha iyi birer alternatif olabilmektedir [103]. Yeni çözüm Eşitlik (2.17)'de gösterildiği şekilde hesaplanır.

$$x_{k+1} = x_k - H_k^{-1} g_k \tag{2.17}$$

**OSS:**BFGS algoritmasının CGD'ye nazaran her iterasyonda daha fazla hesaplama yükü ve bellek gerektirmesi, QN'in hızlı yakınsama özelliği ile CGD'nin düşük bellek ihtiyacı arasında ikincil bir alternatife ihtiyaç duyulmasına sebep olmuştur [106]. Bu

boşluğu doldurmak için geliştirilen OSS algoritması BFGS algoritmasının aksine Hessian matrislerinin tamamını hesaplamaz. Her iterasyonda Hessian matrisinin yaklaşık değeri hesaplanır ve güncelleme gradyan fonksiyonu ile yapılır. Bu şekilde Hessian matrisi tersinin hesaplanmasına gerek kalmaksızın yeni yön belirlenir [103]. Bu algoritma BFGS'ye göre oldukça az bellek ihtiyacına sahipken, CGD'ye göre ihmal edilebilir düzeyde fazla bellek ihtiyacına sahiptir [107].

**LM Algoritmaları**

**LM:** GD ve Newton algoritmaları birlikte kullanılarak geliştirilmiştir. Quasi Newton yöntemlerine benzer şekilde Hessian Matrisini hesaplamaya gerek kalmaksızın ikinci düzey eğitim sağlar [101].Gradyan Eşitlik (2.18)'de gösterildiği şekilde hesaplanır.

$$g = J^T e \tag{2.18}$$

Jakobyen matrisi Hessian matrisine göre çok daha basittir. p, eğtim örnek sayısı; m, çıkış sayısı; n ağırlık sayısı; μ kombinasyon katsayısı ve j=(pxm)xn boyutunda Jakobyen matrisi, e=(pxm)x1 boyutunda hata vektörü olmak üzere ağırlıkların güncellenmesi için kullanılan denklem Eşitlik (2.19)'da, yeni ağırlıkların bulunmasına ilişkin denklem ise Eşitlik (2.20)'de verilmiştir.

$$\Delta w = (J^T.J + \mu I)^{-1} J^T e \tag{2.19}$$

$$w_{k+1} = w_k - [J^T J + \mu I]^{-1} J^T e \tag{2.20}$$

$\mu$ çok büyük seçilirse dik iniş metodu, çok düsük secilirse Newton metodu gibi işlem yapar. Amaç fonksiyonu Newtona yaklaştırmaktır bu yüzden her başarılı adımdan sonra, adım boyutu küçültülür. Bu yöntem 100'den fazla ağırlık içeren ağ yapılarında en uygun teknik olarak bilinmektedir [103]. E uygunluk değeri ve k sabit olmak üzere yeni kombinasyon katsayısının hesaplanmasına ilişkin denklemler Eşitlik (2.21)'de verilmiştir.

$$\mu(n) = \begin{cases} \mu(n-1)k, & E(n) > E(n-1) \\ \frac{\mu(n-1)}{k}, & E(n) \leq E(n-1) \end{cases} \tag{2.21}$$

**BR:** Bayesian Regulazition yöntemi olarak bilinen bu algoritma ağırlık ve bias değerelerini LMoptimizasyon yöntemine göre günceller.

Karesel hata ve ağırlıklar birlikte minimize edilmeye çalışılarak ağın genelleme yapabilmesi için uygun kombinasyonlar seçilir. Bu fonksiyon hesaplamalar için Jakobyen matrisini kullanılır.Bu yüzden ağın performansını değerlendirirken MSE kriterini kullanmak daha uygun olacaktır.

**R:** Her bir iterasyonda eğitim vektörleri ya da diziler ağa farklı sıralarda sunulur. Rastgele olarak gelen bu veri setlerine göre ağırlık ve bias değerleri güncellenerek en uygun çözüme ulaşılmaya çalışılır [89].

Yukarıda anlatılan algoritmalar literatürde yapay sinir ağlarını eğitmek amacıyla sıklıkla kullanılan algoritmalardır. Bu algoritmaların hibrit şekilde birleştirilerek ya da mevcut bazı iyileştirme yöntemi getirilerek bu algoritmaların sayıları giderek artırılmaktadır.

## 2.2. Yapay Arı Kolonisi (Artıfıcıal Bee Colony, ABC) Algoritması

Sezgisel algoritmalar temelde problemin yapısından bağımsız, daha az kontrol parametresi ve daha az işlemle yeni çözümler üretmeyi hedefleyen algoritmalardır. Anlaşılabilirlik açısından kolay olan bu algoritmalar klasik algoritmalara göre daha az hesaplama yüküne sahip ve gerçek dünya problemlerine daha kolay uygulanabililirler. Bu nedenle klasik algoritmalarla kıyaslanabilmesi için sezgisel algoritmalardan olan ABC algoritmasından yararlanılmıştır.

Arıların akıllı yiyecek arama davranışı modelinden esinlenerek Karaboğa tarafından önerilen yapay arı kolonisi algoritması optimizasyon problemlerinde kullanılmaktadır [5].

Algoritmanın akışı genel olarak şu şekildedir:

**1.adım:** Başlangıç için yiyecek kaynağı bölgelerin üretilmesi

REPEAT

**2.adım:** Görevli arıların belirlenen bölgelere gönderilmesi ve kaynaktaki nektar miktarının hesaplanması

**3.adım:** Kaynakların gözcü arılar tarafından seçilme ihtimalini belirleyen olasılık değerlerinin hesaplanması

**4.adım:** Gözcü arıların bu olasılık değerine göre yiyecek kaynağı bölgesi seçmesi ve nektar miktarlarının hesaplanması

**5.adım:** Kaynaktan ayrılma kriterlerine göre bırakılacak kaynakların bırakılması ve kaşif arı üretilmesi

**6.adım:** En iyi çözümün saklanması
UNTİL(Çevrim sayısı= Maksimum çevrim sayısı )

ABC algoritması için temel unsurlar yiyecek kaynakları, görevli işçi arılar kaşif arı ve gözcü arı olarak ayrılan görevsiz işçi arılardır. Yiyecek bulma davranışı için isealgoritmanın genel akışında 2. adım ve 5. adımda görüldüğü üzere yiyecek kaynağına yönelme, kaynağı devre dışı bırakma olmak üzere iki mod vardır.

**Gerçek Arılarda Yiyecek Arama Davranışları:**

Gerçek arılarda görülen sürü zekası beraberinde iş bölümü ve kendi kendine organize olabilme yeteneğini getirmektedir. Bu davranış modeli için Tereshkayiyecek kaynakları, görevli işçi arılar ve görevsiz işçi arılar bileşenlerini kullanarak arıların yiyecek kaynağına yönelme ve kaynağı bıralkma olarak iki modda çalıştığını öne sürmekte ve bileşenlerini şu şekilde açıklamaktadır [98]:

Yiyecek kaynakları besin elde edilen kaynaklardır. Kaynak zenginliği basitlik açısından kaynağın değeri için tek kriterdir. Görevli işçi arılar, keşfedilen kaynaklardan nektarı kovana getirmekle yükümlüdür. Kaynağın kalitesi yeri gibi kaynağa ait bilgileri kovanda bekleyen arılara aktarmakla yükümlüdürler. Görevsiz işçi arılar, kaynak arayışı içindedirler vekaşif arılar ve gözcü arılar olmak üzere iki gruba ayrılırlar. İçgüdüsel ya da herhangi bir dış etmene bağlı olarak rastgele yiyecek arayanlar kaşif arılar; görevli arılardan aktarılan bilgi kılavuzluğunda yeni kaynağa yönelenler ise gözcü arılardır.

Bilginin oluşumu için en önemli kısım bilginin arılar arasında paylaşımıdır. Bu paylaşım yöntemlerinden biri ise kovanlardaki dans alanlarında yapılan danslardır. Bilgi aktarımı için arı dans ederken diğer arılar kaynağın miktar ve koku bilgisi için ona antenleri ile dokunur. Farklı alanlarda yapılan bu dans ile arılar bu kaynağa yönlendirilir. Görevini tamamlamış olan arı kaynağa tekrar gider.Kovana olan uzaklık,

çiçek nektarı kıvamı, besinin genel durumu, kalitedek değişimler ya da hava koşulları gibi etmenler dansı etkilese de en önemli etmenler nektarın ne kadar kolay aktarıldığı ve tatlılığıdır. Nektarın tatlığı dansın eşik değerini belirlemektedir. Bu danslar ile yiyecek getiricilerin diğer yiyecek getiricileri yönlendirmek için hedefe ait yön bilgisini paylaşması gerekir. Bu bilgiye göre güneşten yararlanıp bu şekilde hedeflerini bulurlar.

**Kaynak Kovan Bilgisine Göre Danslar**

**Çember dansı:** Kovandan 50 – 100 metre uzakta bulunan bir kaynak keşfeden tarlacı arı saat yönünde ve aksi yönde hızlı hareketlerle çemberler çizerek diğer arılara çok yakında bir kaynak olduğunu işaret eder. Bu iletişim biçiminde uzaklık hakkında bilgi verilmesine karşın yön ve konum açık değildir. Arılar bu durumu biraz da olsa kolaylaştırmak için gelişmiş koku alma duyularından da yararlanırlar. Çember dansı yapan arının üzerine sinmiş olan çiçek kokusunu dikkatle inceleyen diğer arılar, dışarı çıktıklarında bu kokuyu esas alarak yönlerini daha da çabuk bulabilirler.

**Sallantı Dansı:** Daha uzaktaki kaynaklara dair bilgi vermeyi amaçlayan bu dans çember dansına göre daha karmaşıktır. Bilgiyi getiren arı kovan içindeki asılı petekler üzerinde yerini alarak önce düz bir çizgi üzerinde vücudunun arka kısmını sağa sola sallayarak koşar. Ardından sağa ya da sola keskin bir dönüş yapıp, vücudunu sallamayı bırakarak yarım çember şeklinde bir eğriyi takip ederek dansına başladığı noktaya döner. Başlangıç noktasına ulaştığında aynı düz çizgiyi takip ederek sallantı dansını tekrarlar; ve bu defa ilk seferkinin aksi yöne dönerek diğer yarım çemberi de çizer ve başlangıç noktasına döner. Yön bilgisi sallantılı geçen düz çizgi üzerindeki dans sırasında aktarılır. Kovandaki petekler yerçekimi doğrultusunda düşey konumda bulunmaktadırlar. Arılar, bu yerçekimsel doğrultuyu güneşi temsil etmek için kullanırlar. Eğer arı, yerçekimine tam ters yönde dans ederse arı kaynağa gitmek için güneşe doğru uçması gerektiği bilgisini alır. Dansın yukarı yönden ne yönde ne kadar saptığına bakılarak uçulacak doğrultu da belirlenir. Örneğin, sallantı dansının yapıldığı çizgi sağa doğru $20^o$ eğikse, kovandan çıkılıp güneş karşıya alındıktan sonra, saat yönünde $20^o$ dönüp uçmak gerekir. Uzaklık bilgisi ise doğrusal sallantı dansının süresi ile alakalıdır. Kabaca 1 saniyelik sallantı, 1000 metre uzaklıktaki bir kaynağa işaret eder [108].

Gözcü arılar bu dansları izleyerek seçecekleri kaynağa karar verir. Zengin kaynakların bilgi aktarımı fazla olduğundan seçilme olasılıkları fazladır.

Yiyecek arama davranışının genel akışı şu şekilde özetlenebilir:

**1.adım:**Kaşif arılar rastgele yiyecek aramaya başlarlar.

**2.adım:** Yiyecek bulan kaşif arı artık görevli arıdır ve kovana rastgele nektar taşır. Daha sonra ya bu kaynağa geri döner ya da alanda dans ederek gözcü arılara bilgi aktarımı sağlar. Kaynak bitince görevli arı tekrar kaşif arıya dönüşür ve bu adım başa sarar.

**3.adım:** Gözcü arılar zengin kaynakları gösteren dansları izleyerek kaliteye göre frekanstan kaynağa yönelirler.

**Algoritmanın Gerçek Yiyecek Arama Davranışından Farkları**

Gerçek yiyecek arama davranışlarından farklı olarak, algoritmanın uygulanabilirliğini artırmak amacıyla bazı kabuller yapılmıştır. Bunlar:

1. Görevli arı sayısı yiyecek kaynağı sayısına eşittir. Yani bir kaynak nektarı ancak bir görevli arı tarafından kullanılabilir.

2. Gerçek yiyecek arama davranışında tükenmiş kaynağın arısı olan kaşif arı algoritmada yoktur. Yani işçi arı sayısı gözcü arı sayısına eşittir.

3. Optimizasyon probleminin olası çözümleri yiyecek kaynaklarının yerlerine, çözümün uygunluğu da nektar miktarına karşılık gelmektedir.

4. ABC algoritması en fazla nektara sahip kaynağın yerini bularak probleme göre istenen minimum ya da maksimum çözümü önermektedir.

**Gerçek Yiyecek Arama Davranışının Algoritmaya Uyarlanması**

Algoritmanın akışı çözüm kümesini oluşturacak yiyecek kaynağı bölgelerinin üretilmesi, çözümün uygunluğu için işçi arıların bu bölgelere gönderilmesi ve daha sonra bu kaynaklara yani çözümlere seçim için uygunluk ve olasılık değeri atanması olarak üç başlıkta incelenebilir.

**Başlangıç Yiyecek Kaynağı Bölgelerin Üretilmesi**

Algoritmanın öncelikli hedefi gerçek yiyecek arama davranışındaki yiyeceklere karşılık gelen çözümlerin üretilmesidir. Bunun için Eşitlik (2.22) ile ifade edildiği şekilde her parametrenin alt üst sınırları arasında rastgele değerler üretilir

$$x_{ij} = x_j^{min} + rand(0,1)(x_j^{max} - x_j^{min}) \quad (2.22)$$

i=1…SN (SN: Kaynak sayısı) , j=1…D (D optimize edilecek parametre sayısı) , $x_j^{min}$ j.parametre alt sınırı ve $x_j^{max}$ j. parametre üst sınırını göstermektedir.

Başlangıç aşamasında her kaynağın geliştirilememe sayacı olarak bilinen failure sayacı sıfırlanmaktadır. Başlangıç aşamasından sonra yiyecek kaynakları görevli, gözcü ve kaşif arı süreçlerinden geçirilerek daha iyisi bulunmaya çalışılır. Durdurma kriteri olarak hata fonksiyonu, sayaç değeri ya da uygulamaya göre istenen herhangi bir kriter kullanılabilir.

**İşçi Arıların Kaynak Bölgelerine Gönderilmesi**

Bu aşamada öncelikli olarak gerçek davranıştan farklı olan kabullerde birinci maddede söylendiği gibi her kaynağın bir görevli arısı olduğu kabul edilmektedir. Bu işçi arı kendi yiyecek kaynağına komşu olan yeni kaynağı bulur ve kalitesini değerlendirir. Eğer yeni kaynak, kendi kaynağından daha iyiyse hafızasına bu kaynağın bilgisini alır. Bu işlem komşuluk ilişkisinden de yararlanılarak Eşitlik (2.23) şeklinde ifade edilir.

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \quad (2.23)$$

Bu aşamada φ=[-1:1] olmak üzere başka bir komşu çözümün j. parametresinin farkları alınıp mevcut parametreye eklenir.

$(x_{ij} - x_{kj})$farkı azaldıkça yeni çözüm, eski çözüme yaklaşacaktır. Yerel minimum ya da maksimum noktasına yani optimum çözüme yaklaştıkça bu değişim miktarı da azalacaktır.

$v_{ij}$sınır değerlerini aştığı takdirde yeni çözüm geçerli olmayacak, yeni çözüm olarak sınır değeri kabul edilecektir. Yeni kaynağın sınırlara göre durumu Eşitlik (2.24)'deki şekilde ifade edilir.

$$\boldsymbol{v_{ij}} = \begin{cases} x_j^{min},\ \boldsymbol{v_{ij}} < x_j^{min} \\ \boxed{\boldsymbol{v_{ij}},\ x_j^{min} \leq \boldsymbol{v_{ij}} \leq x_j^{max}} \\ x_j^{max},\ \boldsymbol{v_{ij}} > x_j^{max} \end{cases} \blacktriangleright \text{Sınırlar içerisinde kaldığından artık yeni bir kaynağı temsil eder.} \quad (2.24)$$

$x_i$ ile gösterilen her bir kaynak için bu kaynağın tek bir parametresi (j) değiştirilir. Bu şekilde $x_i$ komşuluğunda $v_i$ yeni kaynağı bulunur. Aç gözlü seçme işlemi adı verilen işleme göre elde edilen $v_i$ çözümü $x_i$ çözümünden daha iyiyse hafızaya artık bu kaynak yeni çözüm olarak alınır ve $x_i$ silinir. Aksi takdirde yeni çözüm eski çözüme nazaran daha kötüyse failure sayacı çözüm geliştirilemediği için bir artar.

Her bulunan yeni kaynağa $\text{fitness}_i$ uygunluk değeri atanır. Bu değer kaynağın kalitesini göstermektedir. Uygunluk değeri arttıkça kaynağı seçecek arı sayısı da artacaktır. Bu işlemin matematiksel ifadesi Eşitlik (2.25)'de verilmiştir.

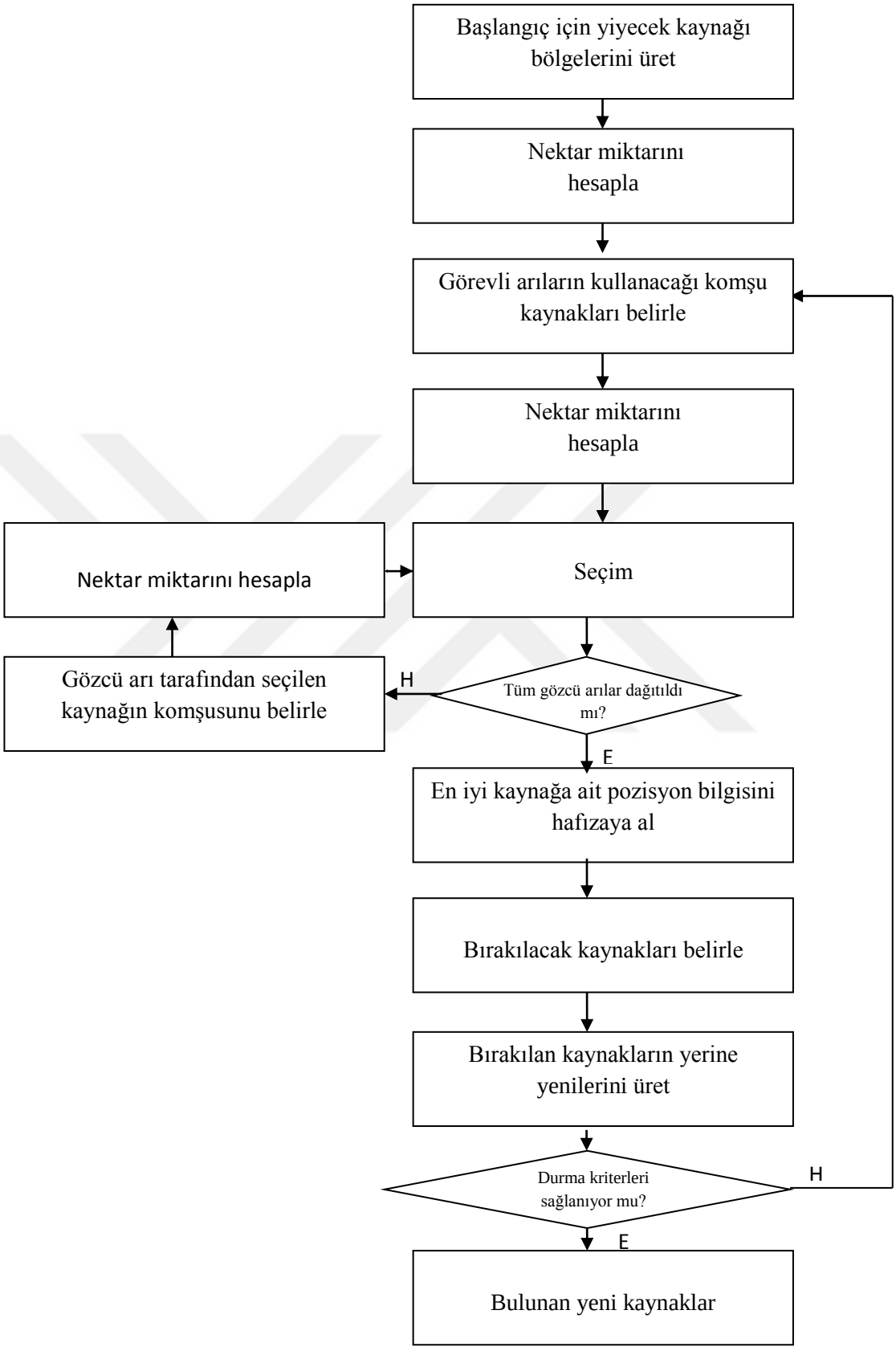
$$\text{fitness}_i = \begin{cases} 1 + \text{abs}(f_i),\ f_i < 0 \\ 1/(1 + f_i),\ f_i \geq 0 \end{cases} \quad (2.25)$$

**Gözcü Arıların Seleksiyonda Kullanıacakları Olasılık Değerinin Hesaplanması (Dans Benzetimi)**

Dans ile aktarılan nektar miktarı ile kaynak seçimi olasılığı arasında bir oran mevcuttur. Bu oran uygunluk değerleri ile Eşitlik (2.26)'deki şekilde hesaplanır:

$$p_i = \frac{fitness_i}{\sum_{n=1}^{SN} fitness_n} \quad (2.26)$$

ABC algoritmasınanın akış diyagramı Şekil 2.16'da görülmektedir [109].




*Şekil 2.17.* ABC Algoritması Akış Şeması

## 2.3. Performans Değerlendirme Kriterleri

Yapay sinir ağlarında, ağın gerçek verileri modellemede ne kadar başarılı olduğunun belirlenmesi amacıyla hata fonksiyonlarından, regresyon eğrilerinden ve bu eğrilerden elde edilen determinasyon katsayılarından ayrıca sınıflandırma problemleri için ROC eğrileri ve karışıklık matrislerinden yararlanılır.

### 2.3.1. Hata Fonksiyonları

Performans ölçümünde geçmişten günümüze kadar hesaplama hızı, tahmin doğruluğu, yakınsama hızı ve yapay sinir ağları için ağın başarısı gibi çok çeşitli amaçlarla farklı kriterlerden yararlanılmıştır. Bu hata fonksiyonlarını temelde mutlak hata fonksiyonları, oransal hata fonksiyonları, göreceli hatalar ve simetrik hatalar olmak üzere dört gruba ayırmamız mümkündür [110].

**Mutlak Hata Fonksiyonları**

Bu fonksiyonlar gerçek değerle tahmin edilen değer arasındaki farkı temel alarak bu hata değeri üzerinde işlemler yapar. Literatürde yaygın olarak karşılaşabileceğimiz hata türleri olmasına rağmen ölçüm aralığı ve dış etkenlerin performansa etkisi gerçek dünya verileriyle çalışırken karşılaşılan en büyük sorunlardandır.

Mutlak hataya dayalı bütün fonksiyonlarda karşılaşılan hata değeri $e_t$, hedef değer $f_t$ ve tahmin edilen değer yani bu tez çalışması için ağın çıktısı $y_t$ olmak üzere Eşitlik (2.27)'de gösterildiği şekilde hesaplanır.

$$e_t = f_t - y_t \tag{2.27}$$

**MAE:** Yaygın olarak kullanılan hata kriterlerinden olan MAE, ortalama mutlak hata şeklinde ifade edilebilir ve Eşitlik (2.28)'de gösterildiği şekilde hesaplanır.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|e_i| = \mathrm{mean}\,|e_i| \tag{2.28}$$

**MSE:** Mutlak hatadan yararlanarak hesaplanan bu değer ortalama karesel hata olarak ifade edilebilir ve Eşitlik (2.29)'de gösterildiği şekilde hesaplanır.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(e_i{}^2) = \mathrm{mean}\,(e_i{}^2) \tag{2.29}$$

**RMS:** Ortalama karesel hatanın karekökü olarak ifade edilebilen bu değer Eşitlik (2.30)'da gösterildiği şekilde hesaplanır.

$$\mathrm{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i{}^2)} = \sqrt{\mathrm{mean}\,(e_i{}^2)} \tag{2.30}$$

**Oransal hata fonksiyonları**

Literatürde yine sıkça kullanılan hatanın hedef değere oranı kullanılarak geliştirilen fonksiyonlardır. Oran $p_t$ ile ifade edilmek üzere Eşitlik (2.31)'de gösterildiği şekilde bulunur.

$$p_t = \frac{e_t}{f_t} \tag{2.31}$$

**MAPE (Mean Absolute Percentage Error):**Mutlak hatanın yüzde ortalaması şeklinde ifade edebileceğimiz bu değer Eşitlik (2.32)'de gösterildiği şekilde hesaplanır.

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n} 100.|p_i| = \mathrm{mean}(100.|p_i|) \tag{2.32}$$

**Simetrik Hata Fonksiyonları**

Oransal hatalarda hatanın mutlak değeriyle değil kendisiyle ilgilenildiği için simetrik olmayan fonksiyonlardır. Çok büyük hata ya da çok küçük hata değerlerinde gerçekçi olmayan sonuçlar üretebileceğinden simetrik hata fonksiyonları geliştirilmiştir.

Simetrik hata fonksiyonlarında temel aldığımız oran $s_t$ ile ifade edilecek olursa Eşitlik (2.33)'de gösterildiği şekilde hesaplanır.

$$s_t = \frac{|e_t|}{(y_t + f_t)} \tag{2.33}$$

**SMAPE:** Simetrik hata fonksiyonlarından en yaygın olarak kullanılanıdır. Eşitlik (2.34)'de gösterildiği şekilde hesaplanır.

$$\mathrm{SMAPE} = \frac{1}{n}\sum_{i=1}^{n} 200.|s_i| = \mathrm{mean}(200.|s_i|) \tag{2.34}$$

Oransal hatalara baktığımızda simetrik ya simetrik olmayan her iki durumda da paydanın 0 olması durumunda yani simetrik olmayan fonksiyon için hedef değerin sıfıra eşit olması, simetrik fonksiyon içinse hedef değer ile ağın çıktısının yani tahmini

değerin toplamının sıfır olması durumunda fonksiyon tanımsız olacaktır. Bu sebeple SMAPE fonksiyonu üzerinde bazı değişikliklerle modifiye edilmiş SMAPE olan MSMAPE (Modified SMAPE) fonksiyonu geliştirilmiştir.

Yukarıdaki hata fonksiyonlarından farklı olarak karşılaşabileceğimiz çok fazla sayıda fonksiyon vardır; ancak hata kriteri karmaşıklaştıkça uygulanabilirliği de o kadar zorlaşacaktır. Yukarıdaki kriterler seçilirken yapay sinir ağları ve zaman serisindeki çalışmalar için uygun olması göz önüne alınmıştır. Bu kriterlerin çok sayıda seçilme amacı ise literatürde ki benzer çalışmalar ile performans karşılaştırılması yapılırken sorun yaşamamaktır. Diğer hata fonksiyonları Tablo 2.4'de gösterilmiştir.

*Tablo 2.4.* Hata Fonksiyonları

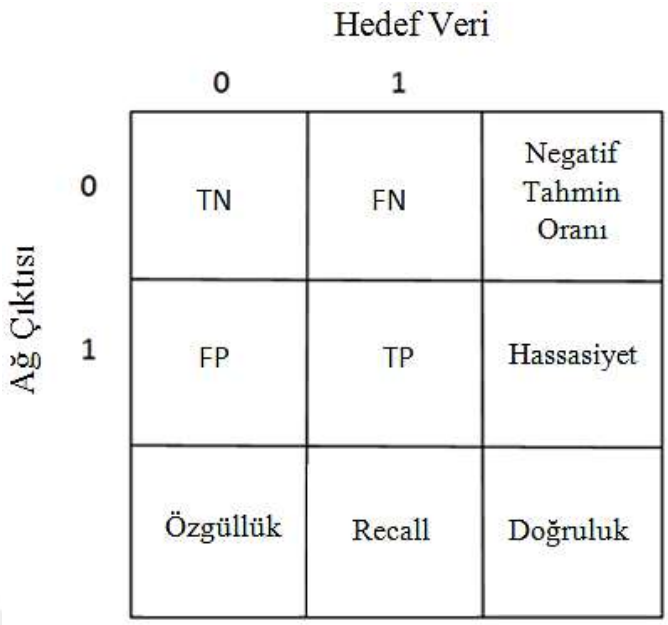
| | |
|---|---|
| Mutlak Hata Fonksiyonları | Mae, Mdae, Mse, **Rms** |
| Yüzdelik Hata Fonksiyonları | Mape, Mdape, Rmspe, Rmdspe |
| Simetrik Hatalar | **Smape**, Smdape, Msmape |
| Göreceli Hata Fonksiyonları | Mase, Rmsse |
| Diğer | Nrmse, İnrse, Std_Ae, Std_ Ape |

Bu hata fonksiyonlarının hangi uygulamaya uygulanacağı konusunda kesin bir kriter olamamakla birlikte yanlış seçim yapıldığında performans için yanlış bilgi vereceği gözden kaçırılmamalıdır.

### 2.3.2. Sınıflandırma Modellerinde Başarı Kriterleri

Yapay sinir ağlarında sınıflandırma problemlerinde yaygın olarak kullanılan ölçütler karışıklık matrisi ve ROC eğrileridir.

**Karışıklık (Konfüzyon) Matrisi**




*Şekil 2.18.* Karışıklık Matrisi

Sınıflandırma problemlerinde yaygın olarak kullanılan ikili sınıflandırma için pozitif ya da negatif olarak iki durum söz konusudur. Şekil 2.17'de görülen karışıklık matrisinde satırlar tahmin edilen değerleri yani ağın çıkışlarını gösterirken sütunlar gerçek çıkışları yani hedef verilere karşılık gelmektedir. Üst sol köşeden alt sağ köşeye kadar olan köşegen üzerindeki hücrelerse doğru olarak yapılan tahminleri göstermektedir. Ters köşegen üstündeki hücreler doğru yapılamayan tahminleri karakterize etmektedir.

Matrisin en sağındaki sütün her bir sınıfa ait tahmin edilen doğru ve yanlış sınıflandırılmış tüm örneklerin yüzdelerini gösterir. Bu metriklere yukarıdan aşağıya doğru sırasıyla negatif tahmin oranı ve hassasiyet denir. Matrisin en altındaki satır ise doğru ve yanlış sınıflandırılmış tüm hedef değerlerin yüzdelerini gösterir. Bu metriklere soldan sağa sırasıyla özgüllük ve recall denir. Matrisin sağ alt köşesi ise genel doğruluğu gösterir.

Hassasiyet: Eşitlik (2.35)'de görüldüğü üzere ağın ürettiği pozitif durum sayısının (True Positive, TP), gerçek durumdaki pozitif durum sayısının yani TP ve yanlış pozitifin (False Positive, FP) toplamına oranıdır.

$$\text{Hassasiyet} = \frac{\text{TP}}{\text{TP+FP}} \tag{2.35}$$

Negatif Tahmin Oranı: Ağın ürettiği negatif durum sayısının gerçek durumdaki negatif durum sayısına oranıdır. Bu oran farklı olarak Eşitlik (2.36)'daki şekilde de ifade edilebilir.

$$\text{Negatif Tahmin Oranı} = \frac{\text{TN}}{\text{TN+FN}} \tag{2.26}$$

Doğruluk(Accuracy):Performans kriterleri açısından en önemlisi sayılabilecek doğruluk kriteri doğru tahminlerin tüm tahminlere oranı şeklinde ifade edilebilir ve Eşitlik (2.37)'deki şekilde ifade edilir.

$$\text{Doğruluk} = \frac{\text{TP+TN}}{\text{TP+FP+FN+TN}} \tag{2.37}$$

**Recall:** Doğru olarak tahmin edilen pozitif değerlerin, gerçek pozitiflere oranıdır; pozitif durumların ne kadar başarılı tahmin edildiğini gösterir. Eşitlik (2.38)'deki şekilde ifade edilir.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{2.38}$$

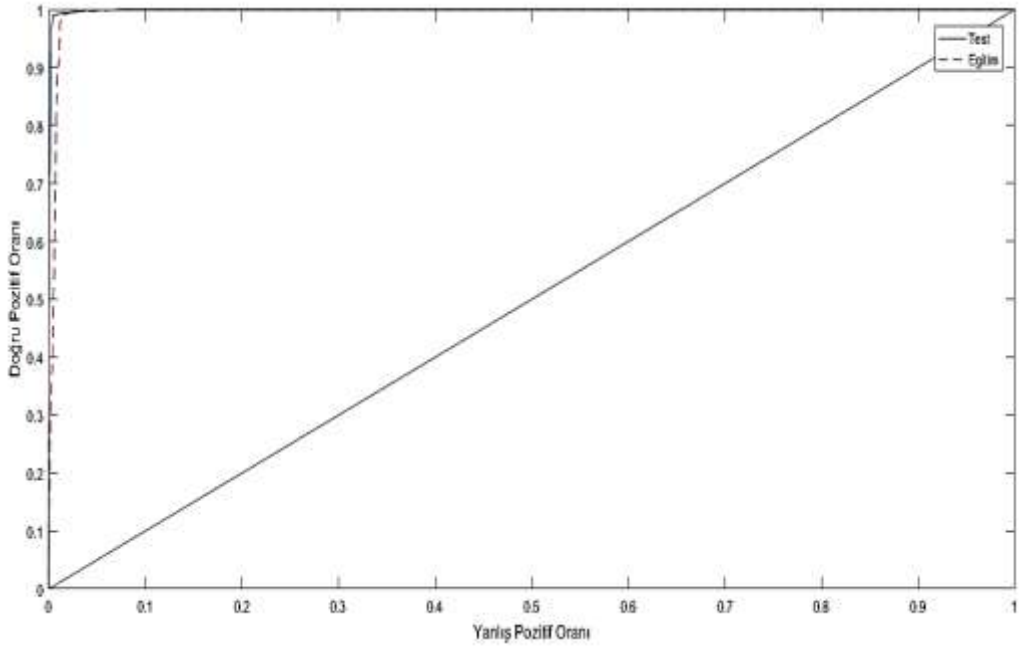
**F1 score -** F1 Score hassaslık ve recall değerlerinin harmonik ortalamasıdır. Bu yüzden tüm yanlış pozitifler ve yanlış negatifler hesaba katılır. İkili sınıflandırmanın tek ölçüt ile değerlendirilmesi için kullanılır. Bu parametre Eşitlik (2.39)'daki şekilde ifade edilir.

$$\text{F1 Score} = 2\times\frac{\text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.39}$$

**ROC Eğrisi**

Tıp uygulamalarında tanı testlerinin değerlendirilmesinde yoğun olarak kullanılan ROC eğrileri, son yıllarda makine araçlarının değerlendirilmesinde de önemli bir yer edinmiştir. ROC eğrisi analizinde temel hedef FP ve FN oranlarının minimize edilerek hassasiyet ve ayırt edilebilirlik değerlerinin maksimize edilmesidir. [3].

ROC eğrileri ANN'lar ile kullanıldığında uygun modelin seçiminde yardımcı olur. Test ve eğitim eğrilerini içeren örnek bir ROC eğrisi Şekil 2.18'de verilmiştir.




*Şekil 2.19.* ROC Eğrisi

ROC eğrisi ile çapraz doğru arasındaki alan 1 değerine yaklaştıkça sınıflandırmanın gücü artacaktır. Bu alana göre ağın performansı arasındaki alan şu şekilde verilmektedir:

0.90-1 arası mükemmel, 0.80-0.90 arası iyi, 0.70-0.80 arası orta, 0.60-0.70 arası zayıf, 0.50-0.60 arası başarısızdır [3].

### 2.3.3. Regresyon Eğrisi

Regresyon fonksiyonu iki veri dizi arasındaki ilişkiyi en iyi temsil eden fonksiyondur. Yapay sinir ağlarında ağın çıktısı ve hedef çıktı arasındaki ilişkiyi gösterir. α kesim noktasını, β ilgili bağımsız değişkenin regresyon katsayısını ve Ɛ değerini göstermek üzere**:**

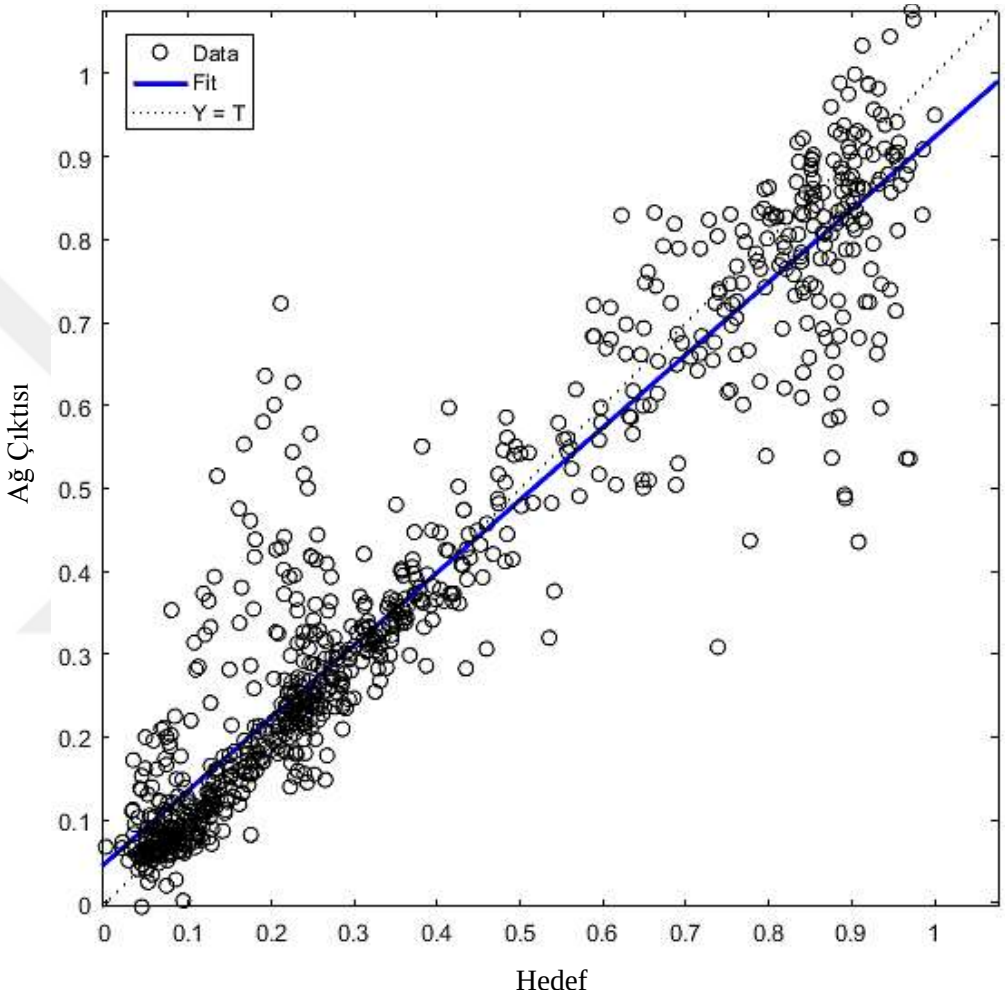
Tek değişkenli basit lineer regresyon Eşitlik (2.40) şeklinde ifade edilir.

$$y = \alpha + \beta x \pm \varepsilon \tag{2.40}$$

Çok değişkenli lineer regresyon denklemi Eşitlik (2.41) şeklinde ifade edilir.

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \pm \varepsilon \tag{2.41}$$

Bu denklemlerde amaç veri noktalarını en iyi temsil edecek bir fonksiyonun bulunmasıdır. Eğri oluşturulurken amaç en küçük kareler yöntemi ile hedeflenen veri noktalarına olan uzaklığın minimum olduğu $f$ fonksiyonunun eğrisinin bulunmasıdır [111]. Regresyon eğrisine örnek olarak Şekil 2.20 gösterilebilir.




*Şekil 2.20.* Regresyon Eğrisi

**Determinasyon Katsayısı:** Regresyon eğrilerinden yararlanılarak ifade edilen bu terim deneysel verilerin doğrusal eğriye ne kadar iyi uyduğunun göstergesidir. $R^2$ ile ifade edilir. $R^2$ =1 ise kusursuz bir doğrusal eğri vardır. Veri noktası ne kadar çok ise güvenilirlik o kadar yüksektir.

$R^2$ regresyon denkleminin verilere ne derece uyum sağlayabildiğinin göstergesidir. Bağımlı değişkendeki değişimin ne kadarının bağımsız değişkenler tarafından açıklanabildiğini gösterir. 0 ile 1 arasında değişen bu değer için 1, bağımlı değişkendeki varyansın %100 oranında bağımsız değişken tarafından açıklanabildiğini gösterir [112].

$R^2$ ile gösterilen determinasyon katsayısı, korelasyon katsayısının karesidir. Bu değer 0.9 çıkıyorsa toplam değişimin %90 ı kurullan modelle, bağımsız değişken tarafından belirlenmiştir, %10 u tesadüfen meydana gelmiştir [113].

# 3. BÖLÜM

# BULGULAR

Bu bölümde tez kapsamında yapılan tüm çalışmalar alt başlıklar şeklinde verilmiştir. Çalışmaların tamamı 3 GB RAM'e sahip, INTEL Core i3 işlemcili bilgisayarda Windows 7 işletim sistemi üzerinde MATLAB 2016 programlama dili ile gerçekleştirilmiştir.

Bölüm 3.1'de internet yoğunluğu verisinden yararlanılarak yapay sinir ağları ağ yapılarından FFNN, CFNN, Elman ve RNN karşılaştırılmıştır.

Bölüm 3.2'de bir önceki bölümde karşılaştırılmaları yapılan ağ yağılarından en yüksek performansı sağlayan RNN ağı, Bölüm 2.1.3.4'de anlatılan eğitim algoritmalarıyla eğitilmiş ve farklı performans kriterleriyle değerlendirilerek en uyguneğitim algoritması bulunmaya çalışılmıştır. Bu bölümde en optimum sonuç LM algoritması ile alınmıştır.

Bölüm 3.3'de geleneksel algoritmaların sahip olduğu hesaplama karmaşıklığı, dezavantajların giderilmesi amacıyla ve gerçek dünya problemlerine daha kolay uygulanabilirliği nedeniyle RNN ağının eğitilmesinde sezgisel algortimalardan olan ABC algoritması kullanılmış ve Bölüm 2.2'de anlatılan algoritmaya özel parametrelerin performansa etkisi incelenmiştir. Bu bölümde her problem için değişebilen bu parametrelerin optimum şekilde ayarlanabilmesi hedeflenmiştir.

Bölüm 3.4'de internet yoğunluğu verisiüzerinde ABC ile LM algoritmalarının performanslarının karşılaştırılması yapılmıştır.

Bölüm 3.5'de ABC algoritması ile eğitilen RNN ağı için ANN eğitiminde sıkça kullanılan temel problemler olan XOR,3-bit parity ve encoderdecoder problemleri için performans metrikleri verilmiştir.

Bölüm 3.6'da zaman serileri, IoT ve mekatronik uygulamalara ait veri setleriyle çalışılmış ve elde edilen sonuçlar yorumlanmıştır.

Bu bölümde verilen tüm tablolarda, en başarılı sonuçlar kalın fontlarla ifade edilmiş ve griye boyanmıştır.

### 3.1. Ağ Yapılarının Karşılaştırılması

Ağ yapılarının karşılaştırılması için kullanılan veri seti [114], 11 ayrı Avrupa şehrinde merkezlere yerleştirilmiş olan internet servis sağlayıcılarının sunduğu internet trafiği verilerini bit formatında içermektedir. Veri akışı 7 Ocak 2005 saat 06.57'de başlayarak 31 Ağustos 2005 saat 11.17'ye kadar devam etmiştir. Bu sürede toplam 1231 veri elde edilmiştir.

Ağ yapılarının eğitimi sırasında veri seti Mackey Glass zaman serileri formatına çevrilmiştir. Buna göre (t-18),(t-12),(t-6) ve t anındaki veriler ağa girdi olarak verilir ve sonucunda (t+6) anındaki çıktı tahmin edilmeye çalışılır.

Bu çevrimden sonra 1207 örnek içeren veri seti; Şekil2.9' da verilen RNN ağına benzer şekilde 4 giriş, 1 çıkıştan oluşmaktadır. Bu örneklerin %70'i eğitim (845 örnek), %30'u (362 örnek) ise test aşamasında kullanılmıştır. Veri setine ilişkin giriş çıkış bilgileri Tablo 3.1'de verilmiştir.

*Tablo 3.1.* İnternet Trafiği Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | (t-18) anında alınan internet trafiği verisi (bit) |
| 2.Giriş | (t-12) anında alınan alınan internet trafiği verisi (bit) |
| 3. Giriş | (t-6) anında alınan alınan internet trafiği verisi (bit) |
| 4.Giriş | (t) anında alınan alınan internet trafiği verisi (bit) |
| Çıkış | (t+6) anı için tahmin edilen alınan internet trafiği verisi (bit) |

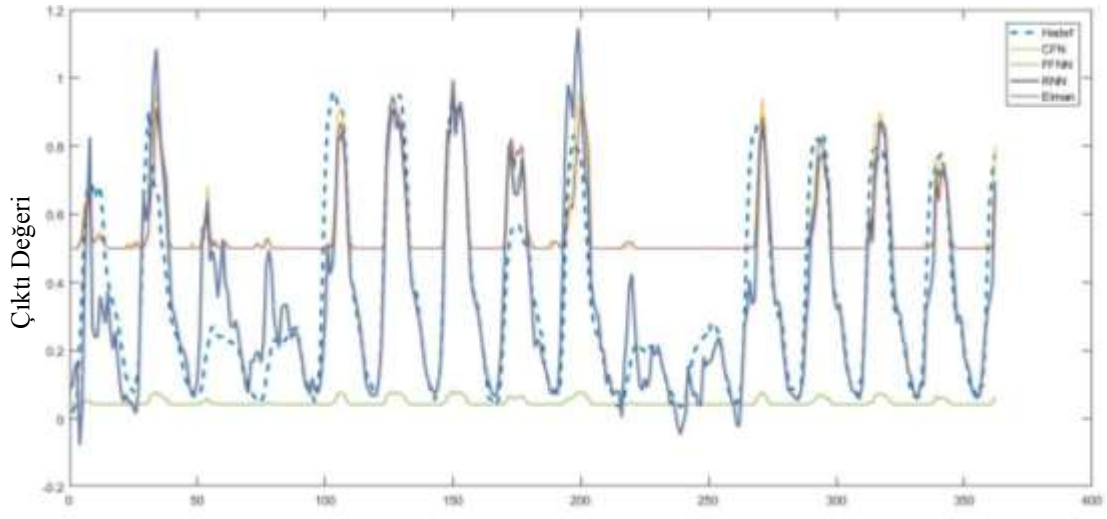
Ağların hepsi 10 nöron içeren tek ara katmana sahiptir, eğitim algoritması olarak LM ve aktivasyon fonksiyonu olarak sigmoid aktivasyon fonksiyonu kullanılmıştır. Elman ve RNN ağlarına bu ağlardan farklı olarak içerdikleri geri besleme bağlantıları sebebiyle (t-1) anından gelen verileri alan gecikme faktörleri eklenmiştir.

Tablo 3.2'de CFNN, Elman, FFNN ve RNN ağlarının RMS ve SMAPE hata kriterlerine göre karşılaştırılması yapılmıştır. Tablodan da görüleceği üzere en iyi sonuç her iki hata kriteri için de RNN ağından alınmıştır. Ortalama değerlere bakıldığında RMS hata kriteri için RNN 0.167 iken en yakın sonuç Elman ağı ile, SMAPE hata kriteri için RNN 0.037 iken en yakın sonuç CFNN ağı ile elde edilmiştir. Her iki sonuca da bakıldığında RNN ağının çok gerisinde kaldıkları açık bir şekilde görülmektedir.

*Tablo 3.2.* Ağ Yapılarının Performans Metriklerine Göre Karşılaştırılması

| | | | RMS | | | SMAPE | | |
|---|---|---|---|---|---|---|---|---|
| AĞ | Min | Ort | Max | Std | Min | Ort | Max | Std |
| CFNN | 0.291 | 0.298 | 0.307 | 0.004 | $4.4.10^{-4}$ | 0.054 | 0.108 | 0.003 |
| Elman | 0.292 | 0.295 | 0.299 | 0.005 | $5.5.10^{-5}$ | 0.075 | 0.108 | 0.038 |
| FFNN | 0.293 | 0.299 | 0.326 | 0.007 | 0.007 | 0.061 | 0.108 | 0.029 |
| RNN | 0.140 | **0.167** | 0.207 | 0.016 | 0.001 | **0.037** | 0.102 | 0.031 |

Şekil 3.1'de ağların 30 koşma için ürettikleri 362 veriden oluşan çıktıların her bir veri için ortalaması alınarak, hedef veriler ile ilişkisi gösterilmiştir. Grafikten de anlaşılacağı üzere RNN diğer ağ yapıları ile karşılaştırıldığında verileri en iyi şekilde takip eden ağ yapısı olurken diğer ağ yapıları çok benzer performanslar göstermekle birlikte en kötü sonuç Elman ve FFNN ağından alınmıştır.



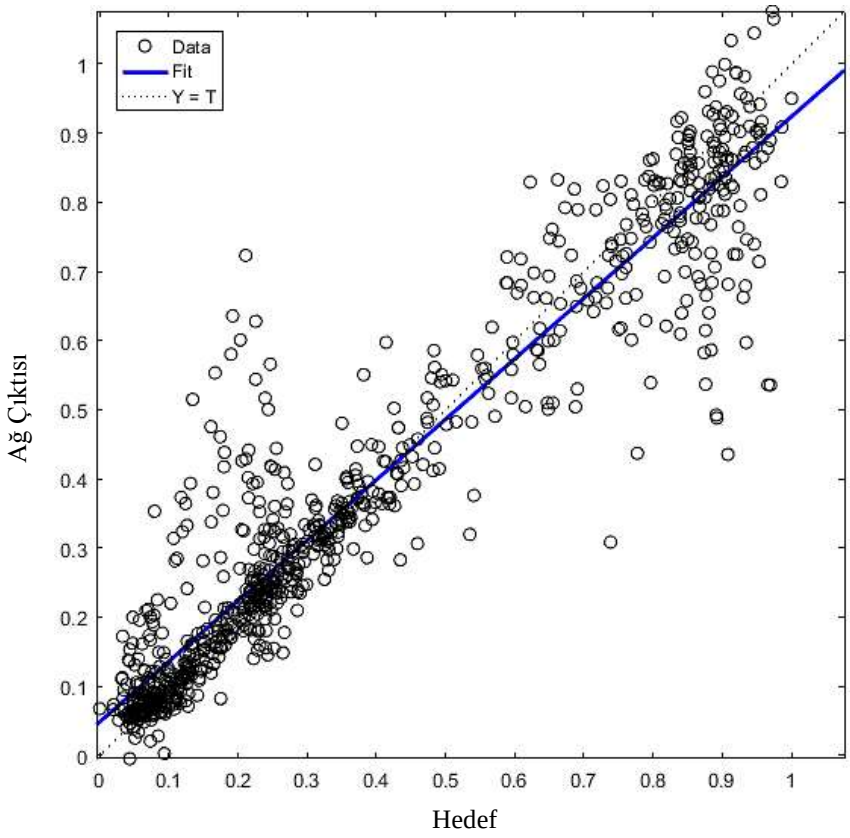

*Şekil 3.1.* CFNN, Elman,FFNN ve RNN Ağı Çıktılarının Hedef Verilerle İlişkisi

Bölüm 2.1.6'da anlatılan determinasyon katsayısı değerlerine bakıldığında elde edilen sonuçlar Tablo 3.3'de görülmektedir. Sonuçlar incelendiğinde eğitim verileri için de test verileri için de RNN ağının performansı diğer ağları geride bırakmıştır. Eğitim verisi için ağın başarısının %94.8 oranında, test verisi içinse %87.2 oranında olduğu görülmektedir.

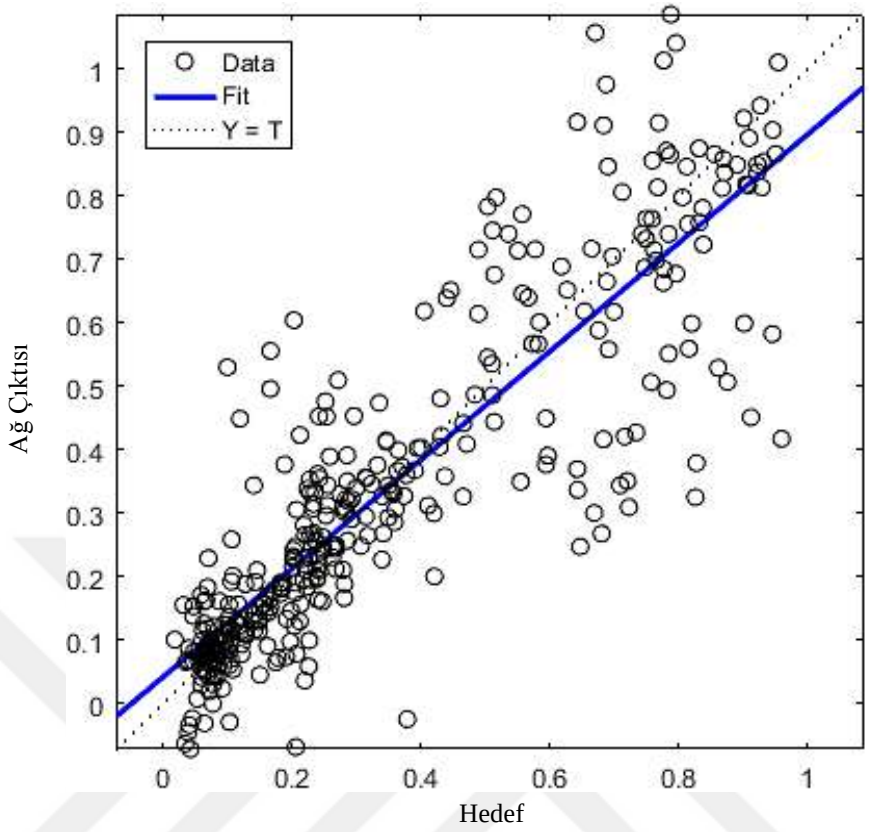
*Tablo 3.3.* CFNN, Elman, FFNN Ve RNN Ağlarının Eğitim Ve Test Verileri için Determinasyon Katsayıları

| | $R^2$ Değeri | |
|---|---|---|
| | Eğitim | Test |
| CFNN | 0.859 | 0.744 |
| Elman | 0.851 | 0.731 |
| FFNN | 0.868 | 0.745 |
| RNN | **0.948** | **0.872** |

RNN ağı, eğitim ve test verileri için regresyon eğrileri Şekil 3.2 ve Şekil 3.3'de görülmektedir. Tablo 3.3'de verilen değerler de göz önüne alınarak şekiller incelendiğinde ağın ürettiği eğrinin hedef verilere oldukça yakın olduğu görülmektedir.

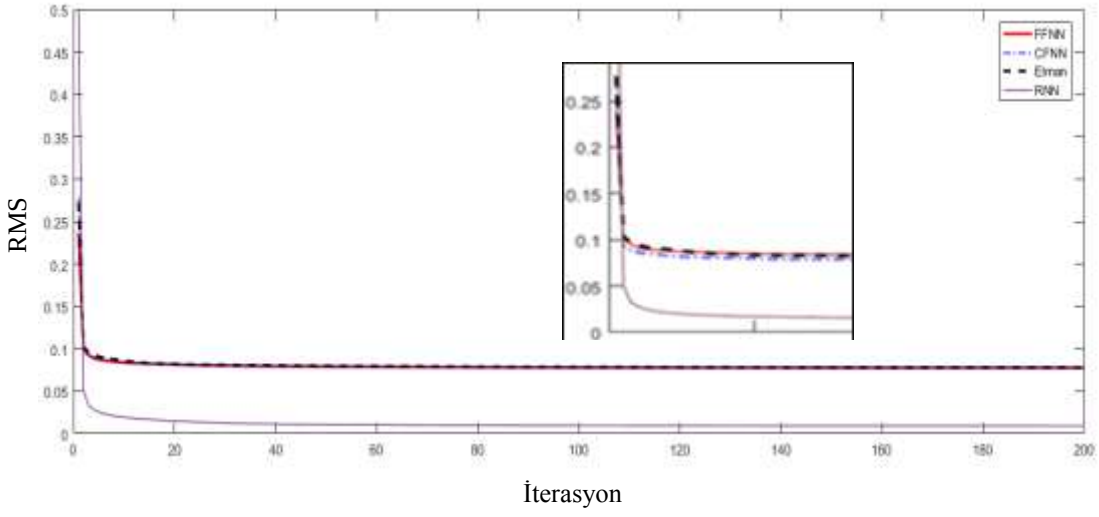



*Şekil 3.2.* RNN Eğitim Verisi için Regresyon Eğrisi




*Şekil 3.3.* RNN Test Verisi İçin Regresyon Eğrisi

CFNN, Elman, FFNN ve RNN ağlarına ait 200 iterasyon için yakınsama grafikleri test verileri için Şekil 3.4'de görülmektedir. RNN ağının 20.iterasyondan itibaren hızlı bir şekilde hatayı sıfıra yakınsadığı görülmektedir.




*Şekil 3.4.* FFNN, CFNN, Elman Ve RNN Ağları Test Verileri İçin Yakınsama Sonuçları

Farklı veri setleriyle farklı sonuçlar elde etmek mümkün olsa da, bu bölümde verilen tablo ve şekillerde de görüldüğü gibi zaman serileri için RNN ağı diğer ağlara göre hem performans kriterleri açısından hem yakınsama performansı açısından daha başarılıdır.

### 3.2. Eğitim Algoritmalarının Karşılaştırılması

Bölüm 3.1'de anlatılan internet trafiği verisi aynı şekilde %70'i eğitim %30'u test verisi olarak ayrılarak tek ara katman ve bu katmanda 10 nöron bulunan, aktivasyon fonksiyonu olarak sigmoid fonksiyonu seçilen RNN ağ yapısı Bölüm 2.1.2'de anlatılan eğitim algoritmaları ile karşılaştırılmıştır. Deneyler 30 koşma ve her koşmada 1000 iterasyondan yararlanılarak tekrarlanılmış ve sonuçlar bu 30 koşmanın ortalaması alınarak verilmiştir. Elde edilen sonuçlar Tablo 3.4'de gösterilmiştir.

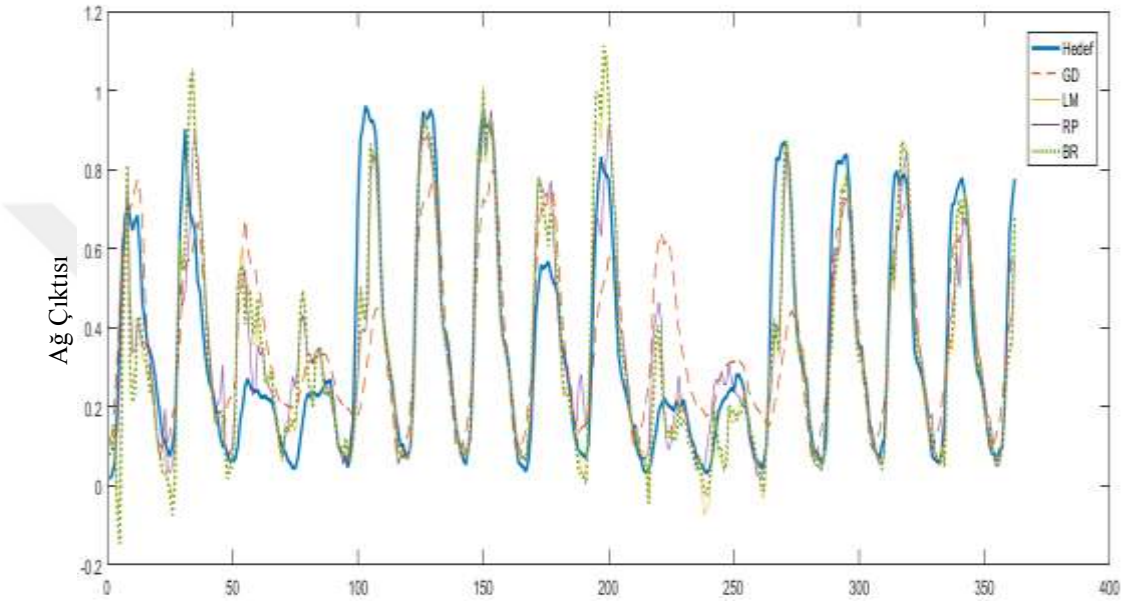
*Tablo 3.4.* RNN Ağı İçin Eğitim Algoritmalarının Karşılaştırılması

| | | RMS | | | | SMAPE | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Min | Ort | Max | Std | Min | Ort | Max | Std |
| GD | GD | 0.176 | 0.185 | 0.193 | 0.004 | 0.044 | 0.065 | 0.084 | 0.011 |
| | GDA | 0.156 | 0.161 | 0.171 | 0.003 | 0.012 | 0.039 | 0.084 | 0.020 |
| | GDM | 0.176 | 0.185 | 0.193 | 0.004 | 0.044 | 0.065 | 0.084 | 0.010 |
| | GDX | 0.144 | 0.153 | 0.161 | 0.004 | 0.001 | 0.041 | 0.096 | 0.024 |
| RP | RP | 0.131 | **0.149** | 0.165 | 0.010 | 0.004 | 0.078 | 0.174 | 0.041 |
| BR | CGF | 0.141 | 0.159 | 0.188 | 0.011 | 0.007 | 0.063 | 0.203 | 0.046 |
| | CGP | 0.143 | 0.157 | 0.186 | 0.010 | $9.5.10^{-4}$ | 0.041 | 0.116 | 0.027 |
| | CGB | 0.140 | 0.162 | 0.184 | 0.011 | 0.003 | 0.055 | 0.165 | 0.039 |
| | SCG | 0.129 | 0.158 | 0.190 | 0.013 | 0.003 | 0.056 | 0.165 | 0.041 |
| QN | BFGS | 0.128 | 0.165 | 0.191 | 0.014 | $1.6.10^{-4}$ | 0.048 | 0.115 | 0.030 |
| | OSS | 0.138 | 0.157 | 0.182 | 0.011 | $7.8.10^{-5}$ | 0.060 | 0.135 | 0.035 |
| LM | LM | 0.140 | 0.167 | 0.207 | 0.016 | 0.001 | **0.037** | 0.102 | 0.031 |
| | BR | 0.139 | 0.158 | 0.175 | 0.011 | $1.6.10^{-4}$ | 0.042 | 0.128 | 0.032 |

Tablo 3.3 den görüldüğü üzere algoritmalar arasında çok büyük bir üstünlük olmamasına karşın göreceli hata kriteri olan SMAPE için en iyi sonuç LM algoritmasından mutlak hataya dayalı ölçüm parametreleri RMS kriteri için ise en iyi sonuç RP algoritmasından alınmıştır. Buna karşın SMAPE hata kriterine bakıldığında

en kötü sonuç RP algoritmasından alınmıştır. En kötü sonuçlar ise GD algoritmalarından alınmıştır.

Şekil 3.5'de en kötü performansa sahip GD ve başarılı sonuçlar üreten LM,BR ve RP algoritmalarına ilişkin grafikler daha net görülebilmesi için verilmiştir.



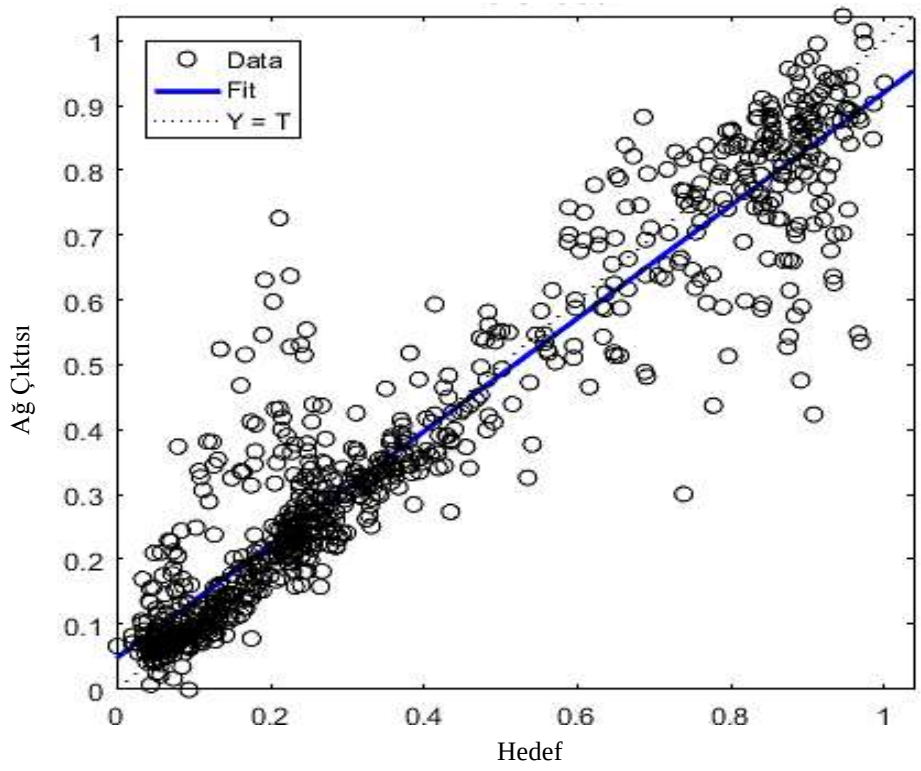

*Şekil 3.5.* GD, LM, RP ve BR ile Eğitilen RNN Ağı Çıktılarının Hedef Verilerle İlişkisi

Tablo 3.5'de eğitim algoritmalarına göre ağın determinasyon katsayıları görülmektedir. En kötü sonuçlar GD algoritmalarından alınmıştır. RP, CGA, QN ve LM algoritmalarına bakıldığında ise yakın sonuçlar üretmişlerdir. En başarılı sonuçlar ise eğitim verileri için LM ve BFGS algoritmalarından alınırken test de bu iki algoritma yakın sonuçlar üretmekle birlikte BFGS algoritması daha başarılı olmuştur. LM ve BFGS algoritmaları eğitim iin verileri modellemekte %94.8 oranında başarı sağlarken en kötü sonuç %77.2 oranında başarı sağlayabilen GD ve GDM algoritmalarından alınmıştır. Test verileri için en başarılı algoritma %87.6 oranında başarı sağlayan BFGS olmakla birlikte LM algoritması da %87.2 oranında başarı sağlamıştır. En kötü sonuç ise %74.2 oranında eğitim verilerinde olduğu gibi GD ve GDM algoritmalarından alınmıştır.

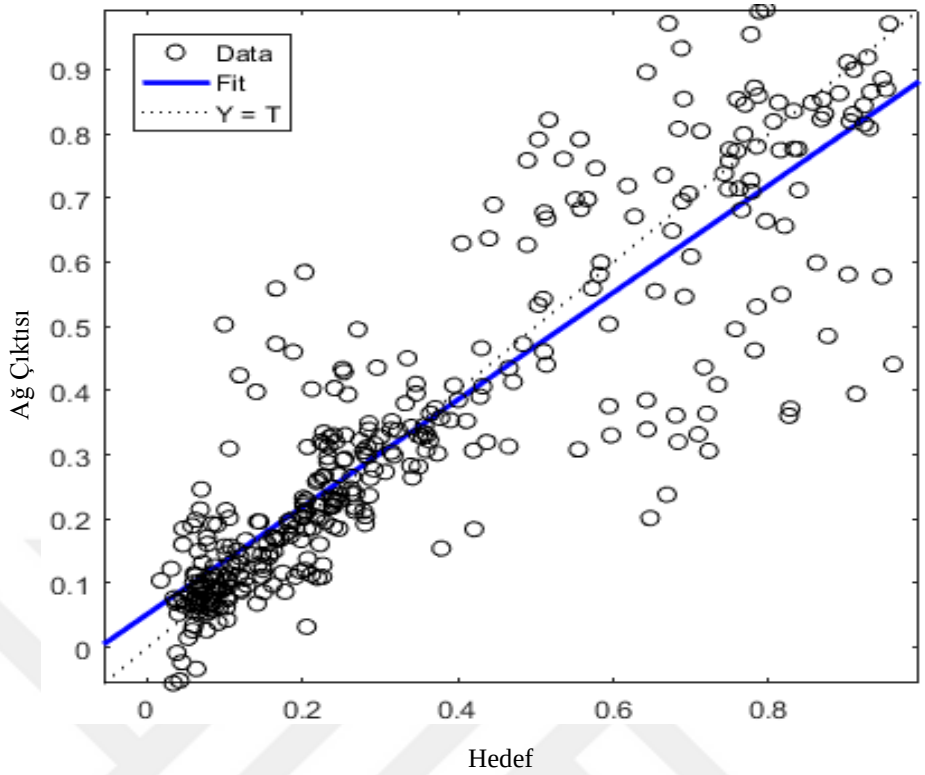
*Tablo 3.5.* Determinasyon Katsayısı İle RNN Ağı İçin Eğitim Algoritmalarının Karşılaştırılması

| | | $R^2$ Değeri | |
|---|---|---|---|
| | | Eğitim | Test |
| GD | GD | 0.772 | 0.749 |
| | GDA | 0.846 | 0.820 |
| | GDM | 0.772 | 0.749 |
| | GDX | 0.883 | 0.848 |
| RP | RP | 0.920 | 0.872 |
| CGA | CGF | 0.939 | 0.871 |
| | CGP | 0.938 | 0.875 |
| | CGB | 0.940 | 0.867 |
| | SCG | 0.939 | 0.875 |
| QN | BFGS | **0.948** | **0.876** |
| | OSS | 0.929 | 0.866 |
| LM | BR | 0.945 | 0.870 |
| | LM | **0.948** | 0.872 |

İnternet trafiği verisi için LM algoritması ile eğitilen RNN ağına ait regresyon eğrisi eğitim ve test verisi olmak üzere sırası ile Şekil 3.2 ve Şekil 3.3'de verilmiştir. BFG algoritmasına ait regresyon eğrileri ise eğitim ve test verisi için sırası ile Şekil 3.6 ve Şekil 3.7 de verilmiştir.

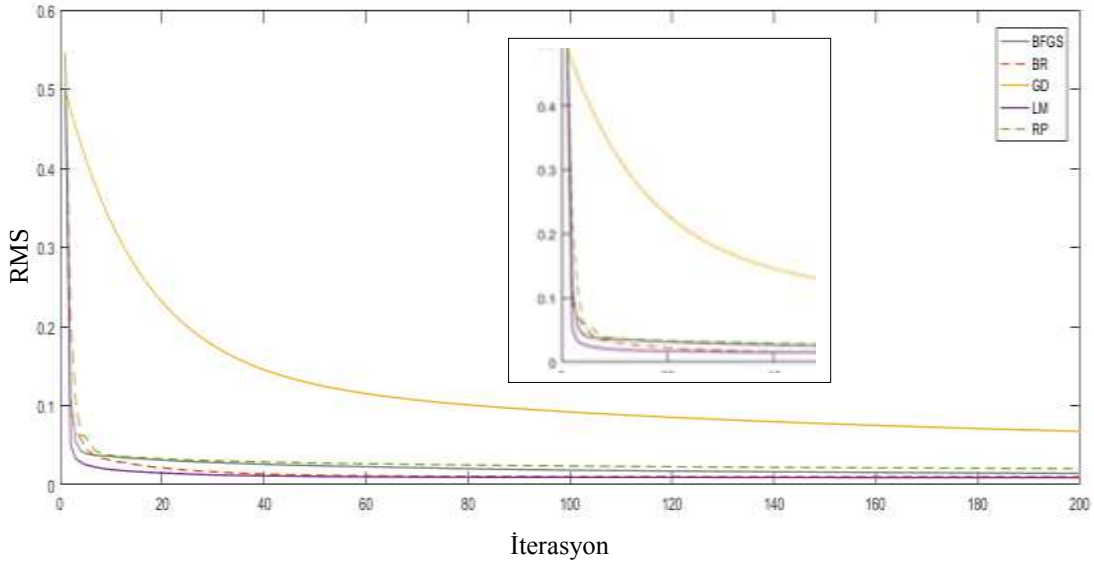



*Şekil 3.6.* Eğitim Verisi İçin BFG Algoritmasına Ait Regresyon Eğrisi




*Şekil 3.7.* Test Verisi İçin BFG Algoritmasına Ait Regresyon Eğrisi

Başarılı olan LM, BR, RP, BFGS ve en uzak sonuları üreten GD algoritmalarının yakınsama grafikleri Şekil 3.8'de verilmiştir. Grafik incelendiğinde GD algoritmasının yakınsama hızı olarak diğer algoritmaların ilk iterasyonlarda geldiği seviyelere 200. iterasyonda hala ulaşamamış olduğu görülmektedir. Diğer algoritmalara bakıldığında birbirlerine çok yakın olmakla birlikte en kısa sürede yakınsayan LM algoritması olmaktadır.




*Şekil 3.8.* Eğitim Algoritmaları İçin Yakınsama Grafikleri

Eğitim algoritmalarının karşılaştırılması amacıyla verilen Tablo 3.4, Şekil 3.5 ve 3.8 incelendiğinde bütün veriler göz önüne alınarak en kötü sonuçların GD algoritmalarından alındığı açık bir şekilde görülmektedir. Hata kriterleri açısından RP algoritması RMS için en iyi sonucu verse de SMAPE için en kötü sonucu üretmiştir. Determinasyon katsayılarına bakıldığında BFGS ve LM en iyi sonuçları üretmiştir. Ancak BFGS hata performansı açısından orta seviyelerde kalmıştır. Yakınsama hızı ise en iyi olan algoritma LM olmuştur. Bu nedenle bütün bu kriterler göz önüne alındığında internet trafiği veri setinde RNN ağını eğitmek için en optimal algoritmanın LM algoritması olduğu görülmektedir.

### 3.3. RNN ağının ABC Algoritması ile eğitilmesi ve Parametre Analizi

Bu bölümde RNN ağlarının eğitiminde ABC algoritmasının kullanımı gerçekleştirilmiştir. Algoritmanın performansını etkileyen parametreler Bölüm3.1 de anlatılan internet trafiği veri seti üzerinde analiz edilmiştir.

Aşağıda etkisi incelenen tüm parametreler için kullanılan RNN ağı tek gizli katmanında 10 nörona sahiptir ve aktivasyon fonksiyonu sigmoid fonksiyondur. Veri seti %70 eğitim, %30 test şeklinde ayrılmıştır. Algoritma için 1000 iterasyon kullanılmış, 30 defa çalıştırılmış ve elde edilen sonuçlar bu 30 koşmanın ortalaması alınarak verilmiştir.

#### 3.3.1. Parametre Aralığının Performansa Etkisi

Aralık etkisinin ağın performansına etkisini gözlemlemek amacıyla algoritmanın popülasyon büyüklüğü 50 olarak, limit değeri 100 olarak belirlenmiştir. Elde edilen RMS sonuçları Tablo 3.6'da görülmektedir.

*Tablo 3.6.* ABC Algoritmasına Aralık Parametresinin Etkisi

| Aralık | Min | Ort | Max | Std |
|---|---|---|---|---|
| (-1:1) | 0.151 | 0.161 | 0.167 | 0.004 |
| (-3:3) | 0.124 | **0.147** | 0.197 | 0.016 |
| (-5:5) | 0.136 | 0.169 | 0.254 | 0.003 |
| (-10:10) | 0.176 | 0.213 | 0.327 | 0.03 |

Tablo 3.6'den anlaşılacağı üzere ortalamada en başarılı sonuç 0.147 olarak [-3,3] aralığında elde edilmiştir.

### 3.3.2. Limit Değerinin Performansa Etkisi

Limit değerinin ağın performansına etkisini gözlemlemek için arama uzayı [-3,3] aralığında, popülasyon büyüklüğü 50 olarak belirlenmiş. Elde edilen RMS sonuçları Tablo 3.7'de görülmektedir.

*Tablo 3.7.* ABC Algoritmasına Limit Parametresinin Etkisi

| Limit | Min | Ort | Max | Std |
|---|---|---|---|---|
| 100 | 0.124 | **0.147** | 0.197 | 0.016 |
| 300 | 0.121 | 0.153 | 0.205 | 0.018 |
| 500 | 0.123 | 0.163 | 0.401 | 0.051 |
| 1000 | 0.124 | 0.150 | 0.177 | 0.126 |

Tablo 3.7'dan anlaşılacağı üzere ortalamada en başarılı sonuç 0.147 olarak limitin 100 değerinde elde edilmiştir.

### 3.3.3. Popülasyon Büyüklüğünün Performansa Etkisi

Popülasyon büyüklüğünün ağın performansına etkisini gözlemlemek için arama uzayı [-3,3] aralığında seçilmiştir, ancak algoritmaların eşit şartlarda çalışmasını sağlamak amacıyla popülasyon büyüklüğü ile iterasyon büyüklüğü çarpımı sabit olacak şekilde iterasyon büyüklüklerinde düzenlemeler yapılmış ve elde edilen RMS sonuçları Tablo 3.8'de gösterilmiştir.

*Tablo 3.8.* ABC Algoritmasına Popülasyon Büyüklüğü Parametresinin Etkisi

| Pop. Büyüklüğü | İterasyon Sayısı | Min | Ort | Max | Std |
|---|---|---|---|---|---|
| 10 | 3000 | 0.116 | 0.153 | 0.244 | 0.026 |
| 30 | 1000 | 0.124 | **0.147** | 0.197 | 0.016 |
| 50 | 600 | 0.138 | 0.156 | 0.245 | 0.019 |
| 100 | 300 | 0.145 | 0.161 | 0.203 | 0.015 |

Tablo 3.8'den anlaşılacağı üzere ortalamada en başarılı sonuç 0.147 olarak popülasyon büyüklüğü 30 değerinde elde edilmiştir.

Tablo 3.6, Tablo 3.7 ve Tablo 3.8 den yararlanılarak ABC algoritması için parametre aralık değeri (-3,3), limit değeri 100 ve popülasyon büyüklüğü değeri 30 en uygun

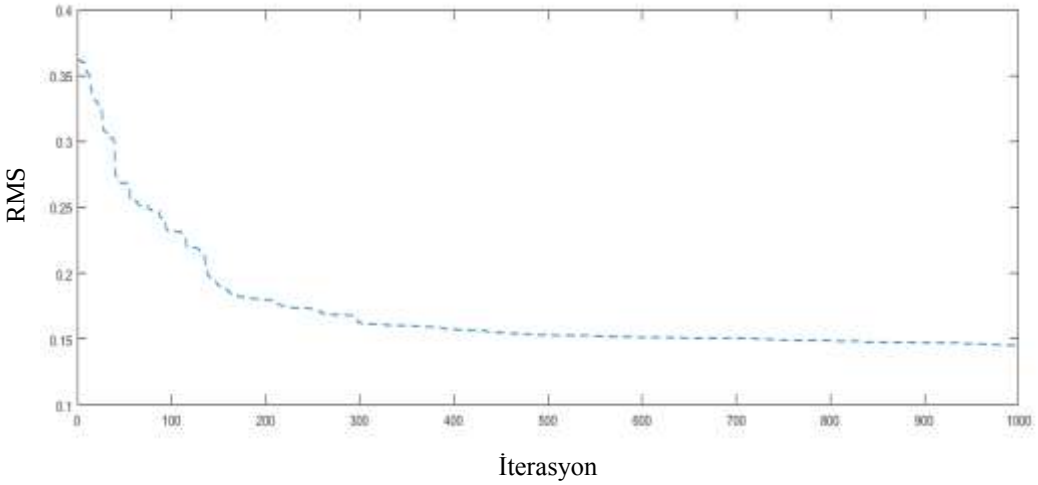
değerler olarak bulunmuştur. Bu değerler ilerleyen bölümlerde farklı problemlerin çözümlerinde kullanılmıştır.

### 3.4. RNN Ağ Eğitiminde ABC Ve LM Algoritmalarının Karşılaştırılması

*Tablo 3.9.* İnternet Trafiği Veri Seti İçin ABC İle LM Algoritmasının Karşılaştırılması

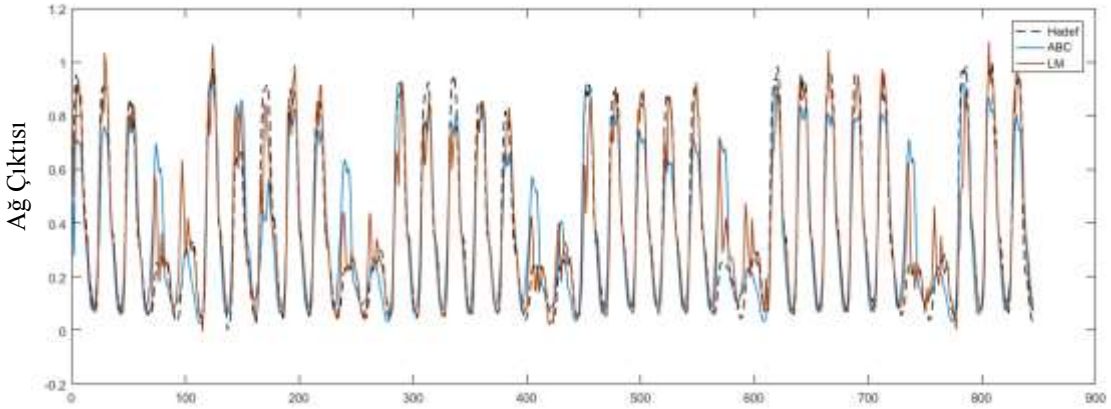
| | RMS | | | | SMAPE | | | |
|---|---|---|---|---|---|---|---|---|
| Alg. | Min | Ort | Max | Std | Min | Ort | Max | Std |
| ABC | 0.124 | **0.147** | 0.197 | 0.016 | $5.42.10^{-4}$ | **$6.36.10^{-4}$** | $8.39.10^{-4}$ | $6.75.10^{-5}$ |
| LM | 0.140 | 0.167 | 0.207 | 0.016 | 0.001 | 0.037 | 0.102 | 0.031 |

Tablo 3.9'daki sonuçlar incelendiğinde ABC algoritmasının LM algoritmasından RMS ve SMAPE kriterlerinin her ikisi için de daha iyi olduğu görülmektedir. Ayrıca Tablo 3.4'de karşılaştırılan tüm algoritmalara da bakılacak olursa her iki hata kriterinde de ABC algoritmasının tüm algoritmalardan daha iyi sonuç verdiği; ancak Şekil 3.9 daki yakınsama grafikleri incelendiğinde yakınsama hızı olarak LM algoritmasının gerisinde kaldığı görülmektedir.

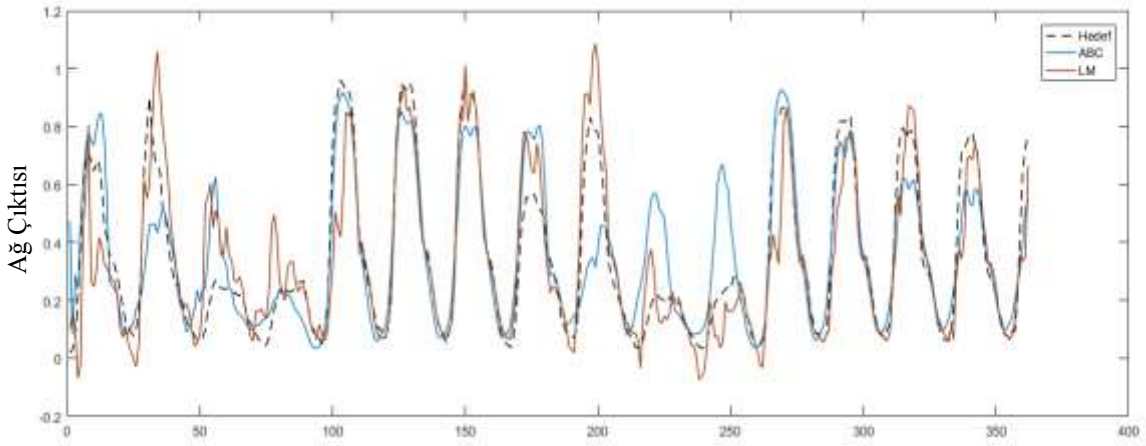



*Şekil 3.9.* ABC Algoritması Yakınsama Grafiği

Şekil 3.10 eğitim verileri için, Şekil 3.11 de ise test verileri için hedef veriler, LM ve ABC algoritmasıyla eğitilmiş RNN ağının çıktıları görülmektedir. Grafik incelendiğinde her iki algoritmanın da hedef verileri izlemekte genel anlamda başarılı olduğu söylenebilir.




*Şekil 3.10.* İnternet Trafiği Eğitim Veri Seti İçin Hedef Veri İle ABC, LM Algoritmalarıyla Eğitilmiş RNN Ağının Sonuçları




*Şekil 3.11.* İnternet Trafiği Test Veri Seti İçin Hedef Veri İle ABC,LM Algoritmalarıyla Eğitilmiş RNN Ağının Sonuçları

### 3.5. Gerçekleştirilen Modelin Temel Problemlere Uygulanması

ABC algoritmasının RNN ağlarındaki performansını belirlemek olarak yapay sinir ağları eğitiminde performans değerlendirme açısından sıkça kullanılan temel problemler olan Xor, 3-Bit Parity, Encoder Decoder problemlerinden yararlanılmıştır.

**XOR Problemi:** Yapay sinir ağlarının temelini oluşturan algılayıcı modelin And, Or problemlerini çözebildiği ancak Xor problemi çözümünde yetersiz kaldığı Bölüm 2.1.5 algılayıcı ağ yapılarında anlatılmıştır. Doğrusasl olmayan problemlerin temelini oluşturan ve çok katmanlı ağların eğitimi için önemli bir yapıtaşı olan Xor problemine ait doğruluk tablosu Tablo 3.10'da verilmiştir.

*Tablo 3.10.* Xor Doğruluk Tablosu

| 1.Giriş | 2.Giriş | Çıkış |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**3-Bit Parity Problemi:** 3 girişe karşılık tek bir çıktı üretilen bu problemde, eşlik (parity) biti olarak adlandırılan bit ikilik tabandaki bitlerin tek veya çift olması esasına göre kontrol amaçlı olarak kullanılan bittir. Bit değerlerinin toplamı tek ise 1 çift ise 0 bilgisini de bu 3 bitlik mesaja ekleyerek 4 bitlik bir mesaj elde edilir. Karşı taraf mesajı aldıktan sonra bit değerlerini toplayarak 4. bit olan eşlik biti ile karşılaştırır. Şayet eşlik biti tutuyorsa sorun yok demektir, şayet tutuşmuyorsa mesajda veya eşlik bitinde bir bozulma var demektir. Probleme ait doğruluk tablosu, Tablo 3.11'da görülmektedir.

*Tablo 3.11.* 3-Bit Parity Doğruluk Tablosu

| 1.Giriş | 2.Giriş | 3.Giriş | Çıkış |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**4-Bit Encoder Decoder:** 4 girişe karşılık 4 çıkışın üretildiği problemlerdir. Girişteki değişimlerin çıkışlara iletilmesini içeren problemlerde girişler aynı şekilde çıkışlara aktarılır. Problemin doğruluk tablosu Tablo 3.12'de görülmektedir.

*Tablo 3.12.* Encoder Decoder Doğruluk Tablosu

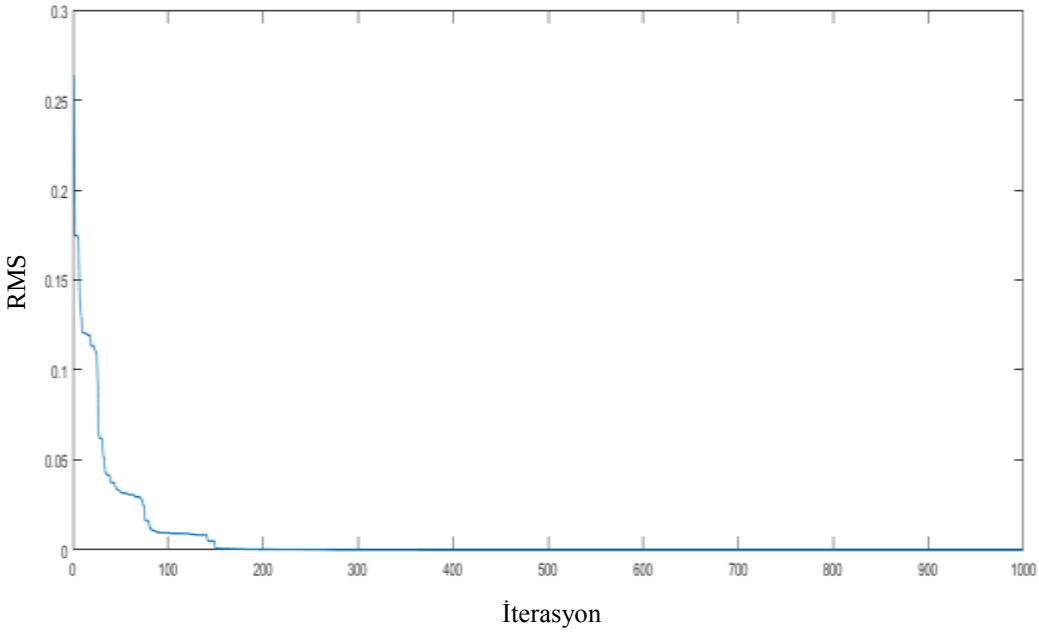
| 1.Giriş | 2.Giriş | 3.Giriş | 4.Giriş | 1.Çıkış | 2.Çıkış | 3.Çıkış | 4.Çıkış |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

Bu temel problemlerin eğitilmesinde ABC algoritmasıyla eğitilmiş RNN ağının yakınsama grafikleri RMS değerleri baz alınarak XOR problemi için Şekil 3.12'da 3bit Parity için Şekil 3.13'de ve encoder decoder problemi için Şekil 3.14'de verilmiştir.

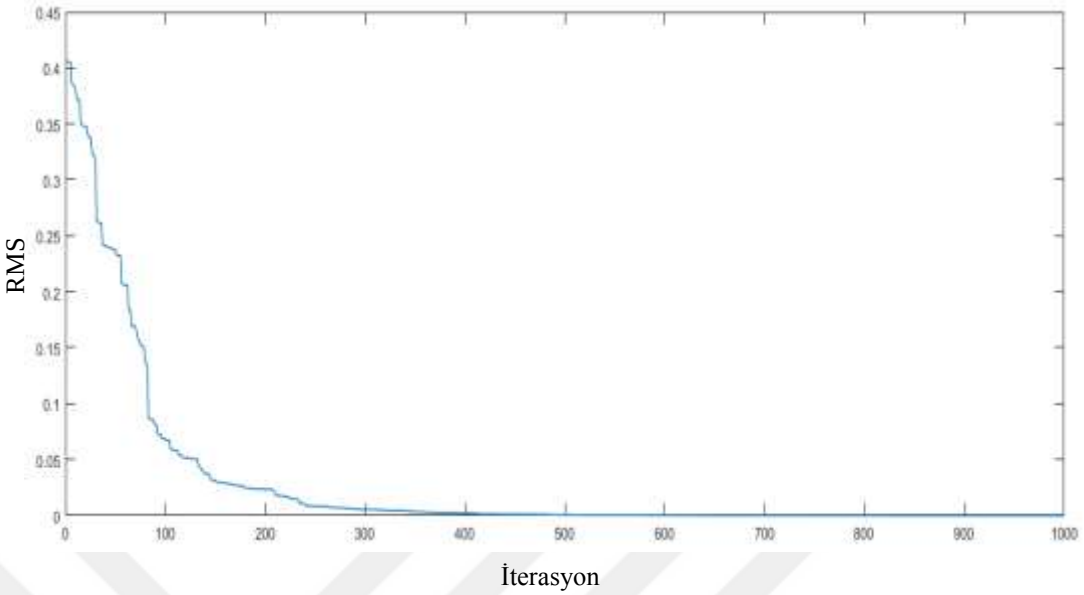
Ayrıca bu problemlerin 30 koşma sonucunda ulaştıkları en düşük,en yüksek ve ortalama RMS ve SMAPE değerleri Tablo 3.13'de görülmektedir. Her üç problem için de tablodan da anlaşılacağı üzere hataların yaklaşık olarak sıfır olduğu ve buna istinaden ağın %100 oranında başarı gösterdiği söylenebilir.

*Tablo 3.13.* Temel Problemler İçin ABC Algoritmasıyla Eğitilen RNN Ağının Performansı

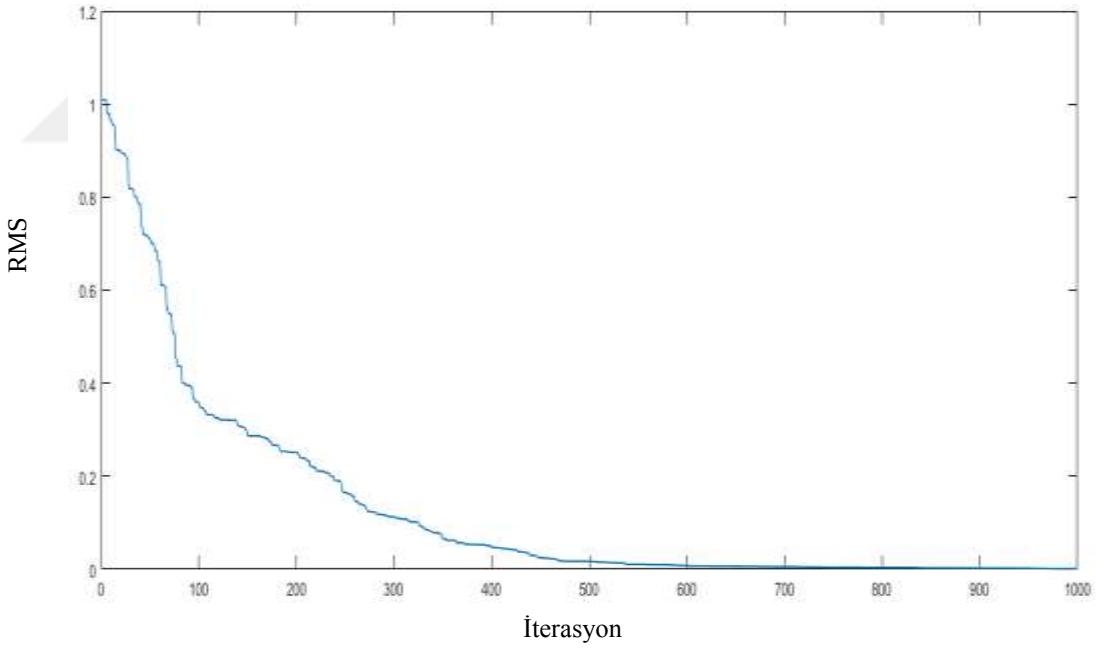
| | | XOR | 3-Bit Parity | Encoder Decoder |
|---|---|---|---|---|
| RMS | **Ort** | **$1.60*10^{-5}$** | **$4.30*10^{-5}$** | **$4.69*10^{-7}$** |
| | Min | $4.49\text{-}10^{-5}$ | $7.68*10^{-6}$ | $1.78*10^{-7}$ |
| | Max | 0.479 | $1.49*10^{-4}$ | $1.22*10^{-6}$ |
| | Std | $8.10*10^{-6}$ | $2.59*10^{-5}$ | $2.64*10^{-7}$ |
| SMAPE | **Ort** | **$9.14*10^{-4}$** | **0.0012** | **$8.90*10^{-7}$** |
| | Min | $4.38*10^{-4}$ | $6.59*10^{-4}$ | $8.06*10^{-4}$ |
| | Max | 0.002 | 0.0031 | 0.0013 |
| | Std | $3.08*10^{-4}$ | $5.02*10^{-4}$ | $3.01*10^{-4}$ |




*Şekil 3.12.* Xor Verisi İçin Yakınsama Grafiği




*Şekil 3.13.* 3-Bit Parity Verisi İçin Yakınsama Grafiği




*Şekil 3.14.* Encoder Decoder Verisi İçin Yakınsama Grafiği

### 3.6. Gerçekleştirilen Modelin Farklı Mühendislik Problemlerine Uygulanması

Modelin uygulanacağı problemlere ait özellikler Tablo 3.14'de görülmektedir.

*Tablo 3.14.* Geliştirilen Modelin Uygulanacağı Problemler

| | | | Probleme Ait Bilgiler | | | |
|---|---|---|---|---|---|---|
| | | | Giriş Sayısı | Çıkış Sayısı | Örnek Sayısı | Eğitim/Test |
| Zaman Serisi | Elektrik | VS1 | 4 | 1 | 476 | 333/143 |
| | Solar Güç | VS2 | 4 | 1 | 2534 | 1774/760 |
| IoT | Trafik | VS3 | 16 | 1 | 135 | 95/40 |
| Mekatronik | Duvar Takip Eden Robot | VS4 | 2 | 1 | 5456 | 3819/1637 |
| | | VS5 | 4 | 1 | 5456 | 3819/1637 |
| | | VS6 | 24 | 1 | 5456 | 3819/1637 |
| Sınıflandırma | Doluluk | VS7 | 5 | 1 | 20560 | 14392/6168 |

Tablo 3.13'de görülen, yararlanılan veri setlerine ait bilgiler ise şunlardır:

VS1: Datamarketten alınan veri setinde Avustralya da Ocak 1956 ile Ağustos 1965 yılları arasındaki elektrik üretim miktarları milyon kwH cinsinden ölçülmüştür. Alınan veri seti Bölüm3.1'de anlatılan Mackey Gauss zaman serisi formatına çevilerek 4 girşli tek çıkışlı bir problem haline getirilmiştir [115]. Veri setine ait girişlerin ve çıkışın bilgisi Tablo 3.15'de gösterilmiştir.

*Tablo 3.15.* Elektrik Üretim Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | (t-18) anında alınan elektrik üretim miktarı (kwH) |
| 2.Giriş | (t-12) anında alınan elektrik üretim miktarı (kwH) |
| 3. Giriş | (t-6) anında alınan elektrik üretim miktarı (kwH) |
| 4.Giriş | (t) anında alınan elektrik üretim miktarı (kwH) |
| Çıkış | (t+6) anı için tahmin edilen elektrik üretim miktarı (kwH) |

VS2: Kaggledan alınan veriseti [116] bir çatıya takılmış güneş paneli ile güç üretimi ölçümlerini 26 Ekim 2011 ve 26 Ekim 2018 tarihleri arasında içermektedir. Veri setinin amacı 1000 kwH üretimin ne zaman gerçekleşeceğini tahmin edebilmektir. Veri setine ait girişlerin ve çıkışın bilgisi Tablo 3.16'da gösterilmiştir.

*Tablo 3.16.* Güneş Paneli Güç Üretim Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | (t-18) anında güneş panelinden üretilen güç miktarı (kwH) |
| 2.Giriş | (t-12) anında güneş panelinden üretilen güç miktarı (kwH) |
| 3. Giriş | (t-6) anında güneş panelinden üretilen güç miktarı (kwH) |
| 4.Giriş | (t) anında güneş panelinden üretilen güç miktarı (kwH) |
| Çıkış | (t+6) anında üretilmesi tahmin edilen güç miktarı (kwH) |

VS3:UCI veri kütüphanesinden alınan veri seti ile IoT da akıllı trafik uygulamalarında da kullanılabilecek bir problem sembolize edilmiştir. Kentsel trafiğin modellenerek trafikteki yavaşlığın tahmin edilip olası rotalar oluşturulmasını öngören problemde 14-18 aralık 2009 tarihleri arasında San Paulo kentinden alınan verilerden yararlanılmıştır [117].

Giriş olarak sabit araç sayısı, arızalı araç sayısı, araç fazlalığı durumu, kaza,itfaiye araç sayısı, yük içeren olaylar, tehlikeli yük içeren araçlar, elektrik kesintisi durumu, yangın, taşkın noktalarının varlığı, troleybüs ağındaki arıza, yolda bulunan ağaç sayısı, semaförlerin kapalı olma durumu ve arızalı semaförler kullanılmıştır. Bu girişlerden yararlanılarak trafik akışındaki yavaşlık yüzde cinsinden tahmin edilmeye çalışılmıştır [118]. Veri setine ait girişlerin ve çıkışın bilgisi Tablo 3.17'de gösterilmiştir.

*Tablo 3.17.* Trafik Yavaşlığı Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | Hareketsiz taşıt sayısı |
| 2.Giriş | Arızalı kamyon sayısı |
| 3. Giriş | Araç fazlalığı durumu (1/0) |
| 4.Giriş | Kazazede sayısı |
| 5.Giriş | Ezilme olaylarının sayısı |
| 6.Giriş | Araç yangını sayısı |
| 7.Giriş | Yük taşıyan araç sayısı |
| 8.Giriş | Tehlikeli yük içeren araç sayısı |
| 9.Giriş | Elektrik kesintisi durumu (1/0) |
| 10.Giriş | Yangın durumu (1/0) |
| 11.Giriş | Taşkın noktaları sayısı |
| 12.Giriş | Yolda gösteri olma durumu (1/0) |
| 13.Giriş | Tramvay hattında arıza durumu |
| 14.Giriş | Yoldaki ağaç sayısı |
| 15.Giriş | Kapalı semafor sayısı |
| 16.Giriş | Aralıklı çalışan semafor sayısı |
| Çıkış | Trafikteki sıkışıklık oranı (%) |

VS 4/VS 5/VS 6: Odada robotun duvarı takip ederek ilerlemesi için kullanılan veri setinde 24 giriş için biri robotun önünde ve biri robotun arkasında, diğerleri de etrafında olmak üzere 24 adet ultrasonik sensör bulunmaktadır. VS4 için 60º açılarla yerleştirilmiş önünde ve solunda olan ikisi, VS5 için 60º açılarla yerleştirilmiş önünde, arkasında, sağında ve solunda olan dördü ve VS6 için farklı açılarla yerleştirilmiş olan bu sensörlerin tamamı kullanılmıştır. Ağın çıktısı tektir ve hafif sağa dön(1), keskin sağa dön(2), düz ilerle(3) ve hafif sola dön(4) şeklinde dört farklı yönlendirme komutundan oluşmaktadır. Çıktıları düşünüldüğünde problem, sınıflandırma problemi olarak ifade edilebilir [119].

VS4'e ait girişlerin ve çıkışın bilgisi Tablo 3.18'de, VS5'e ait girişlerin ve çıkışın bilgisi Tablo 3.19'da ve VS6'ya ait ait girişlerin ve çıkışın bilgisi Tablo 3.20'de gösterilmiştir.

*Tablo 3.18.* İki Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | Robotun önünün duvara olan mesafesi (1.sensör verisi) |
| 2.Giriş | Robotun önünün duvara olan mesafesi (2.sensör verisi) |
| Çıkış | Yönlendirme komutu (1/2/3/4) |

*Tablo 3.19.* Dört Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | Robotun önünün duvara olan mesafesi (1.sensör verisi) |
| 2.Giriş | Robotun arkasının duvara olan mesafesi (2.sensör verisi) |
| 3.Giriş | Robotun sağının duvara olan mesafesi (3.sensör verisi) |
| 4.Giriş | Robotun solunun duvara olan mesafesi (4.sensör verisi) |
| Çıkış | Yönlendirme komutu (1/2/3/4) |

*Tablo 3.20.* Yirmidört Sensörlü Duvar Takip Eden Robot Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| Girişler<br>(24 tane) | Sırasıyla 15° aralıklarla yerleştirilen sensörlerin etraflarındaki duvara göre mesafesi |
| Çıkış | Yönlendirme komutu (1/2/3/4) |

İki, dört ve yirmidört sensörlü robotun şematik gösterimi Şekil 3.15'de görülmektedir.



*Şekil 3.15.* Sırasıyla İki, Dört Ve Yirmidört Sensörlü Robotun Şematik Gösterimi

VS7: Enerji kullanımını optimize etmek amacıyla bir odadan alınan ışık, sıcaklık, $C0_2$, nem sensörleri ve odaya yerleştiren kameradan alınan verilere göre oda dolu (1) ya da boş (0) şeklinde sonuçlar üreten bir sınıflandırma problemidir [120] . Veri setine ait girişlerin ve çıkışın bilgisi Tablo 3.21'de gösterilmiştir.

*Tablo 3.21.* Oda Doluluğu Veri Seti Girişleri Ve Çıkışı

| | |
|---|---|
| 1.Giriş | Odadaki ışık miktarı (lux) |
| 2.Giriş | Oda sıcaklığı (◦C) |
| 3.Giriş | Odadaki $C0_2$ oranı (%) |
| 4.Giriş | Odadaki nem oranı (ppm) |
| 5.Giriş | Kameradan gelen görüntü |
| Çıkış | Oda dolu ya da boş bilgisi (1/0) |

Bütün problemlerde toplam örnek sayısının %70'i ağın eğitimi, %30'u ağın testi amacıyla kullanılmıştır.

Deneyler bu bölümdeki tüm deneylerde olduğu gibi 1000 iterasyon boyunca 30 koşmada tekrarlanmıştır. ABC algoritması için Bölüm 3.4'de elde edilen optimum değerler kullanılmıştır.

Elde edilen sonuçlar eğitim verileri için Tablo 3.22'de, test verileri için 3.23'de verilmiştir.

*Tablo 3.22.* Çalışılan Veri Setleri İçin RNN Ağının Eğitim Performans Metrikleri

| | RMS | | | | SMAPE | | | |
|---|---|---|---|---|---|---|---|---|
| | Min | Ort | Max | Std | Min | Ort | Max | Std |
| VS1 | 0.014 | 0.016 | 0.019 | 0.001 | $9.46.10^{-5}$ | $1.05.10^{-4}$ | $1.18.10^{-4}$ | $6.92.10^{-6}$ |
| VS2 | 0.003 | 0.005 | 0.006 | $8.68.10^{-4}$ | $3.71.10^{-6}$ | $1.95.10^{-6}$ | $5.07.10^{-6}$ | $8.21.10^{-7}$ |
| VS3 | 0.025 | 0.031 | 0.038 | 0.003 | $2.17.10^{-4}$ | $1.42.10^{-4}$ | $2.52.10^{-4}$ | $2.85.10^{-5}$ |
| VS4 | 0.218 | 0.226 | 0.231 | 0.003 | $4.63.10^{-5}$ | $4.81.10^{-5}$ | $4.99.10^{-5}$ | $9.53.10^{-7}$ |
| VS5 | 0.207 | 0.213 | 0.217 | 0.003 | $4.55.10^{-5}$ | $4.98.10^{-5}$ | $5.35.10^{-5}$ | $1.45.10^{-6}$ |
| VS6 | 0.205 | 0.211 | 0.217 | 0.003 | $5.47.10^{-5}$ | $6.02.10^{-5}$ | $6.41.10^{-5}$ | $2.49.10^{-6}$ |
| VS7 | 0.109 | 0.112 | 0.113 | 0.024 | $2.01.10^{-6}$ | $2.11.10^{-6}$ | $2.15.10^{-6}$ | $5.58.10^{-8}$ |

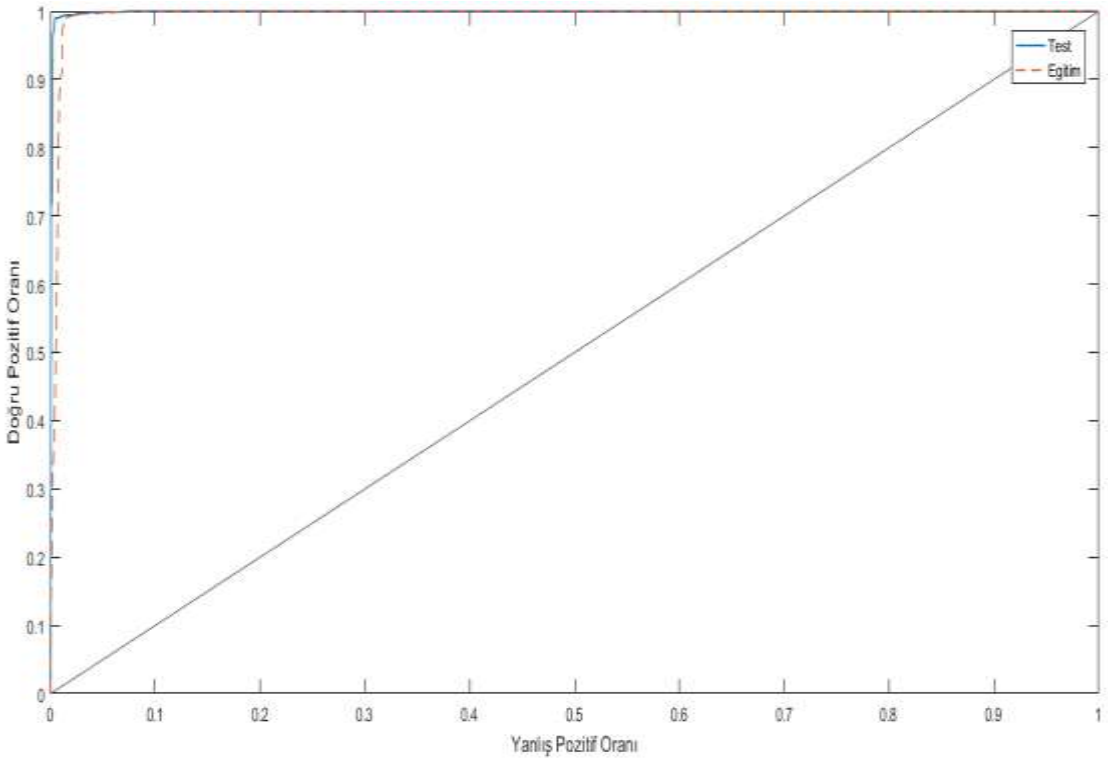
*Tablo 3.23.* Çalışılan Veri Setleri İçin RNN Ağının Test Performans Metrikleri

| | RMS | | | | SMAPE | | | |
|---|---|---|---|---|---|---|---|---|
| | Min | Ort | Max | Std | Min | Ort | Max | Std |
| VS1 | 0.059 | 0.139 | 0.343 | 0.072 | $3.68.10^{-4}$ | $9.26.10^{-4}$ | 0.005 | $8.81.10^{-4}$ |
| VS2 | 0.002 | 0.091 | 0.316 | 0.067 | $1.52.10^{-5}$ | $9.85.10^{-5}$ | $2.57.10^{-4}$ | $6.38.10^{-5}$ |
| VS3 | 0.051 | 0.072 | 0.112 | 0.014 | $8.51.10^{-4}$ | 0.001 | 0.002 | $1.92.10^{-4}$ |
| VS4 | 0.224 | 0.235 | 0.273 | 0.011 | $1.11.10^{-4}$ | $1.17.10^{-4}$ | $1.22.10^{-4}$ | $2.59.10^{-6}$ |
| VS5 | 0.201 | 0.217 | 0.233 | 0.006 | $1.09.10^{-4}$ | $1.22.10^{-4}$ | $1.34.10^{-4}$ | $5.69.10^{-6}$ |
| VS6 | 0.208 | 0.221 | 0.234 | 0.007 | $1.29.10^{-4}$ | $1.39.10^{-4}$ | $1.58.10^{-4}$ | $2.49.10^{-6}$ |
| VS7 | 0.068 | 0.088 | 0.120 | 0.024 | $3.67.10^{-6}$ | $7.19.10^{-6}$ | $2.15.10^{-5}$ | $4.72.10^{-6}$ |

Tablo 3. 22 ve Tablo 3.23 incelendiğinde hem eğitim hem test sonuçları için elektrik ve solar güç verilerini içeren zaman serileri için beklendiği gibi ağın performansının başarılı olduğu ortalama RMS değerlerinin sırasıyla 0.139 ve 0.091 olduğu görülmektedir. Trafik verilerini içeren IoT verilerinde de RMS değerlerinin problemde çok fazla giriş olmasına karşı ortalama 0.072 RMS değeri ile başarılı bir şekilde modellendiği söylenebilir.

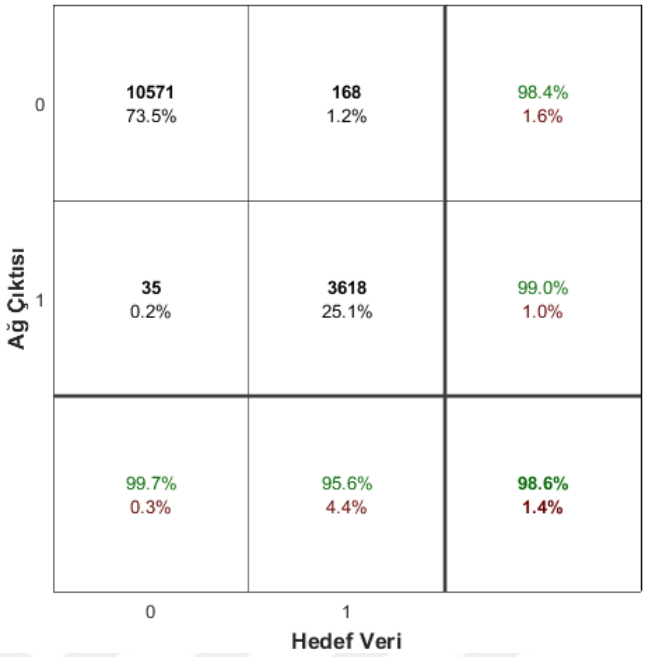
Ancak robot yönlendirme problemleri incelendiğinde ağın çok başarılı sonuçlar üretemediği giriş sayılarının değişmesine karşın ortalama RMS değerlerinin 0.21 değerlerinin üzerine çıkamadığı görülmüştür.

Doluluk algılama problemi incelendiğinde ise RMS ve SMAPE değerlerine bakılarak ağın başarılı olduğu söylenebilir ancak bir sınıflandırma problemi olduğundan ağ performansının değerlendirilmesinde Bölüm 2.3.2'de anlatıldığı şekilde ROC eğrilerinden ve karışıklık matrislerinden yararlanılabilir. VS7 için eğitim ve test verilerine ait ROC eğrileri Şekil 3.16'da, karışıklık matrisleri sırasıyla Şekil 3.17 ve Şekil 3.18'de görülmektedir.



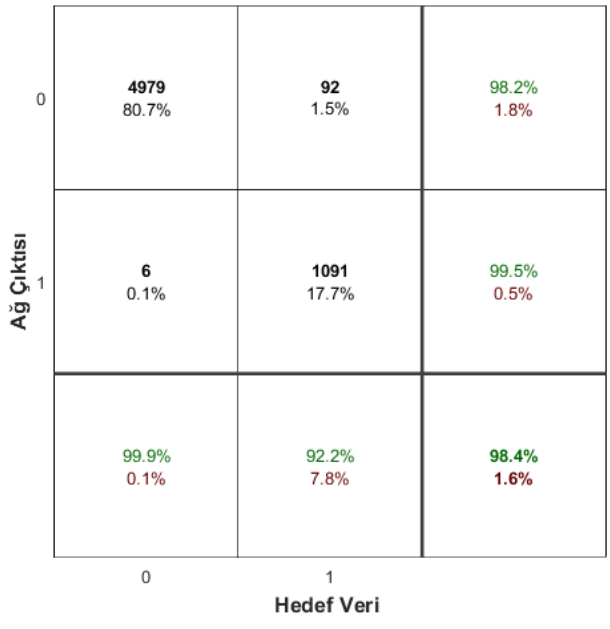

*Şekil 3.16.* Doluluk Algılama Problemi İçin Roc Eğrisi

Şekil 3.16 da gerekli hesaplamalar yapılarak altta kalan alan hesaplandığında eğitim verileri için eğitim verileri için 0.994 test verileri içinse 0.999 olarak bulunmuştur. Bölüm 2.3.2'de anlatılan alan ve ağ başarısı arasındaki ilişkiye bakıldığında 0.90 ile 1 arası ağın problemi mükemmel bir şekilde modellediğini göstermektedir.




*Şekil 3.17.* Doluluk Algılama Problemi Eğitim Verisi İçin Karışıklık Matrisi

Şekil 3.17'den görüldüğü üzere ağ toplam durumun %73.7 si olan boş bilgisini içeren negatiif durumların %73.5'ini, %26.3'ü olan dolu bilgisini içeren pozitif durumların %25.1'ini doğru tahmin etmiştir. Sonuç olarak toplam 14392 eğitim verisinin 14189'unu doğru tahmin edilmiştir yani ağın eğitim verilerini modellemekteki başarısı %98.6 olmuştur.




*Şekil 3.18.* Doluluk Algılama Problemi Test Verisi İçin Karışıklık Matrisi

Şekil 3.17'den görüldüğü üzere ağ toplam durumun %80.8 si olan boş bilgisini içeren negatif durumların %80.7'ünü, %19.2'si olan dolu bilgisini içeren pozitif durumların %17.7'sini doğru tahmin etmiştir. Sonuç olarak toplam 6168 test verisinin 6070'ini doğru tahmin edilmiştir yani ağın test verilerini modellemekteki başarısı %98.4 olmuştur.

Literatürde aynı veri setini kullanan farklı çalışmalarla kıyaslandığında; Ferraria kentsel trafik yoğunluğu için VS3 verisetini nörobulanık yöntemde 16 girişli tek çıkışlı 3 gizli katman ve her gizli katmanda 10 nöron olacak şekilde modellemiştir [118]. Elde ettiği sonuçları grafiksel olarak belirten çalışmaya göre daha başarılı sonuçlar elde eden ABC algoritmasıyla eğitilen RNN ağı 0.072 RMS değerine sahiptir ve ağı modellemede %95 oranında başarı göstermiştir.

Dash ve arkadaşları sezgisel algoritmalardan yerçekimsel arama yöntemini (Gravitational Search, GS) ve parçacık sürü optimizasyon algoritmasını içeren GS-PSO hibrid algoritması ile tek gizli katmanlı FFNN ağını eğiterek VS4/VS5/VS6 verisetlerinin analizinde kullanmışlardır [121]. Analizde popülasyon büyüklüğü değeri 30 ve 50 alınmış ve gizli katman nöron sayıları Tablo 3.13 de belirtilen giriş sayılarına eşit, giriş sayılarının iki ve üç katı kadar alınarak üç farklı yapı oluşturulmuştur. Sonuçlar her bir veri seti için ayrı ayrı incelenmiştir. VS4 için en iyi sonuç popülasyon büyüklüğü 30 ve 6 nörona sahip ağdan,VS5 için en iyi sonuç popülasyon büyüklüğü 50 ve 12 nörona sahip ağdan, VS6 için en iyi sonuç popülasyon büyüklüğü 50 ve 48 nörona sahip ağdan alınmıştır. Çalışmada farklı ağ yapıları ve farklı algoritmalarla alınan sonuçlar kıyaslanmıştır.

Tablo 3.16'da VS4, VS5 ve VS6 için kıyaslanan çalışmalar ve önerilen RNN- ABC yapısına ait doğruluk değerleri görülmektedir.

*Tablo 3.24.* Robot Navigasyon Verisinde Farklı Çalışmaların Başarı Değerleri

| Yöntem | Pop. | VS4 | VS5 | VS6 |
|---|---|---|---|---|
| RNN-ABC | 30 | 76.847 | 73.854 | **80.268** |
| RNN-ABC | 50 | 69.701 | **76.847** | 79.352 |
| GSPSO-ANN [122] | 30 | 79.670 | 73.663 | 71.886 |
| GSPSO-ANN [122] | 50 | 79.341 | 73.150 | 73.992 |
| GS-FNN [123] | 30 | 86.382 | 70.088 | 69.721 |
| GS-FNN [123] | 50 | 85.686 | 67.705 | 69.721 |
| GD-ANN [124] | - | **89.56** | 71.59 | 46.50 |

Tablo 3.16'da incelendiğinde 2 girişli olan VS4 için ağın giriş nöron sayısı azalmakta ve matematiksel işlem yükü azaldığı için GD algoritması daha başarılı sonuçlar üretebilmektedir; ancak iki sensörden alınan verileri gerçek ortamda tüm engellerden kaçmak çok olası görülmemektedir. Giriş sayısı arttıkça ağın karmaşıklığı artmakta ve bu noktada önerilen yöntem daha iyi sonuçlar üretmektedir. 4 sensörlü VS5 için en başarılı performans popülasyon büyüklüğü 50 olan RNN-ABC algoritmasından alınırken, VS6 için en başarılı sonuç popülasyon büyüklüğü 30 olan ABC algoritmasından alınmıştır.

VS7 temelde bir sınıflandırma problemi olduğundan karşılaştırma yöntemi olarak ROC eğrisi ve karışıklık matrisinden elde edilen sonuçlardan yararlanılmıştır. Sonuç kısaca şu şekilde özetlenebilir; Tutuncu ve arkadaşlarının yaptığı çalışmada [125] FFNN ağını geleneksel algoritmaların bütünüyle eğitmiş ve QN algoritmasından en başarılı sonucu almıştır. Sonuçlar ABC ile eğitilen RNN ağı ile karşılaştırıldığında, RNN ağının eğitim için başarı oranı %98.84 ve test için % 99.06 olan QN algoritmasıyla eğitilen FFNN ağının gerisinde kaldığı görülmektedir; ancak ROC eğrisine bakıldığında Tütüncü ve arkadaşlarının yapmış olduğu çalışmada eğitim verisi altında kalan 0.987 ve test için 0.995 olurken bu çalışmada sırasıyla 0.994 ve 0.999 olarak bulunmuştur. Sonuçlar Tablo 3.25 ve 3.26'da görülmektedir.

*Tablo 3.25.* Doluluk Algılama Problemi İçin Farklı Çalışmaların Doğruluk Oranları

| Karışıklık Matrisi | Eğitim (%) | Test (%) |
|---|---|---|
| ABC-RNN | 98.589 | 98.411 |
| Quick Propagation-FFNN | 98.895 | 96.033 |
| CGD-FFNN | **98.944** | 98.038 |
| QN-FFNN | 98.932 | 96.524 |
| Limited Memory QN-FFNN | 98.846 | **99.061** |
| LM-FFNN | 80.739 | 95.741 |
| Online BP-FFNN | 98.625 | 98.382 |
| Batch BP-FFNN | 78.701 | 80.324 |

Probleme ait ROC eğrisi Şekil 3.15'de görülmektedir. Bu ROC eğrisinden elde edilen verilerle Tütüncü ve arkadaşlarının yapmış olduğu çalışmanın karşılaştırılması Tablo 3.26'da verilmiştir.

*Tablo 3.26.* Doluluk Algılama Problemi İçin ROC Eğrileri Alanları

| Roc Eğrisi | Eğitim | Test |
|---|---|---|
| ABC-RNN | **0.994** | **0.999** |
| Limited Memory QN | 0.987 | 0.996 |

Tablo 3.25 incelendiğinde en başarılı perdormansu test aşamasında QN-FFNN yönteminin sağlamasının yanında ABC-RNN yöntemi de bu yöntemi de en yakın performansı sağlayan ikinci yöntem olduğu görülmektedir.

Tablo 3.26 incelendiğinde ise QN-FFNN ve ABC-RNN yöntemlerinin ROC eğrileri altında kalan alan açısından ABC-RNN yönteminin daha başarılı olduğu yani doğru pozitifleri ABC-RNN ağının daha başarılı modellediği söylenebilmektedir.

# 4. BÖLÜM

# TARTIŞMA – SONUÇ ve ÖNERİLER

## 4.1. Tartışma – Sonuç

Günümüzün ve geleceğimizin trend konularından olan IoT, halihazırda büyük veriler üretmektedir. Yapılan literatür taraması göstermiştir ki IoT kavramının giderek yaygınlaşan bir kavramdır ve veri analizi günümüzün ve geleceğin en önemli araştırma konularından biridir.

Giderek artan kullanımıyla birlikte Şekil 1.1'den de görüldüğü üzere 2020'li yıllarda internete bağlı cihaz sayısının 30 milyara yaklaşması beklenmektedir. Bu cihazlardan üretilen yüksek miktarda veriler genellikle zaman serisi formatında olup etkili analiz yöntemleriyle tahmin ve önlem uygulamalarında sıklıkla kullanılmaktadır.

Bu çalışmada zaman serilerinde veri analizi için lineer olmayan ilişkileri kolaylıkla çözebilen ANN modellerinden yararlanılmıştır. Öncelikle literatürde çoğunlukla karşımıza çıkan temel ağ yapılarından ileri beslemeli FFNN, Cascade ve geri beslemeli RNN ve Elman ağları birbirleriyle karşılaştırılmıştır. Önceki iterasyonda ürettiği değerleri gelecek iterasyonlarda kullanabilen geçmişten öğrenme yeteneğini sahip RNN ağları ileri beslemeli ağlardan FFNN, Cascade ağlarını geride bırakmasının yanısıra RNN lerin temeli kabul edilen Elman ağlarından da beklendiği üzere daha başarılı sonuçlar üretmiştir.

Daha sonraki aşamada RNN ağı kullanılarak geleneksel eğitim algoritmaları GD,GDA,GDM,GDX; RP; CGF,CGP,CGB,SCG;BFGS; OSS; LM ve BR algotritmaları birbiriyle karşılaştırılmıştır. Sonuçlar RMS, SMAPE hata kriterleri, determinasyon katsayısı ve yakınsama grafikleri açısından beraber değerlendirildiğinde en başarısız

algoritmatların GD algoritmaları olduğu açık bir şekilde görülmüştür.Hata kriterleri açısından en iyi sonuçlar LM ve RP algoritmalarından elde edilmekle birlikte RP algoritması RMS için en iyi sonucu vermesine karşın SMAPE için en kötü sonucu üretmiştir. Determinasyon katsayılarına bakıldığında BFGS ve LM en iyi sonuçları üretmiştir ancak BFGS hata performansı açısından orta seviyelerde kalmıştır. Yakınsama hızı ise en iyi olan algoritma yine LM olmuştur. Bu nedenle en optimal sonucun LM algoritmasından alındığı söylenmiştir.

IoT, zaman serisi veri analizinde LM algoritmasıyla eğitilen RNN ağı iyi bir seçenek olmakla birlikte literatürde yapılan çalışmalar incelendiğinde ABC algoritmasının ANN ağlarını eğitmede oldukça başarılı sonuçlar ürettiği görülmüştür. Bu nedenle literatürde farklı ağ yapılarının eğitiminde kullanılan ABC algoritması RNN ağının eğitiminde kullanılmıştır.

Sonuçlar incelendiğinde RMS ve SMAPE kriterlerinin her ikisi için de ABC algoritmasının LM algoritmasından da diğer tüm geleneksel algoritmalardan daha başarılı sonuçlar verdiği görülmüştür ve farklı veri setlerine uygulanarak sonuçlar incelenmiştir.

İncelenen sonuçlarda ABC algoritmasıyla eğitilen RNN ağı zaman serilerinde ve IoT verilerinde yüksek performansı sağlamıştır; ancak sınıflandırma problemleri ve çıktıları düşünüldüğünde sınıflandırma problemi olarak nitelendirilebilecek robot yönlendirme uygulamaları için aynı oranda performansı sağlayamamıştır.

### 4.2. Öneriler

Çalışmada istenen sonuçlar büyük oranda sağlanmasına karşın, incelenen sonuçlara bakıldığında ABC algoritması LM algoritmasından daha yüksek performansı sağlamıştır; ancak ABC algoritmasının geniş bir arama uzayında olması yakınsama hızının yavaş olmasına sebep olmaktadır. LM algoritmasının ise var olan çözümü iyileştirme yöntemine gitmesi algoritmanın daha yüksek yakınsama hızına sahip olmasını sağlamaktadır.

Zaman serileri ve IoT verileri için büyük oranda başarı sağlayan RNN ağı sınıflandırma problemlerinde aynı oranda başarıyı gösterememiştir.

Yukarıda anlatılan durumlar değerlendirildiğinde tez çalışmasının aşağıda belirtilen maddelerle geliştirilmesi düşünülmektedir:

- Yapılan çalışmada alınan sonuçlar literatürde kullanılan hazır veri setlerinden yararlanılarak elde edilmiştir. Bu veriler hazırlanacak IoT donanımından zaman serisi olarak alınarak, verilerin analizi tezde anlatılan ABC ile eğitilen RNN ağıyla yapılabilir.
- Sınıflandırma problemlerinde RNN ağının özelleştirilmiş bir türü olan NARX ağı ile daha başarılı sonuçlar elde edilebilir, bu amaçla çalışma NARX ağıyla genişletilebilir ve sonuçlar RNN ağı ile karşılaştırılabilir.
- ABC ve LM algoritmasının eksiklikleri beraber düşünüldüğünde, her iki algoritmanın da olumsuz etkilerinden kurtulmak amacıyla yüksek performans ve hızlı yakınsama özelliğini sağlayacak hibrid ABC-LM algoritması RNN ve NARX ağının eğitiminde kullanılabilir.

Tüm bu maddeler beraber değerlendirildiğinde RNN ağının tüm problemlerde başarıyla kullanılabilmesi açısında NARX ağıyla genişletilmesi, ABC algoritmasının yakınsama hızı konusunda yetersizliğini gidermek için ABC-LM hibrid algoritmasından yararlanılması ve IoT verilerinin hazır veri setleri yerine hazırlancak olan donanımsal yapıdan elde edilmesi planlanmaktadır.

# KAYNAKÇA

[1] Derya Şişmanyazıcı and Buket Doğan, "Nesnelerin internetinde veri madenciliği Data mining on internet of things," , Tekirdağ, 2016, pp. 420-425.

[2] David Floyer. **http://wikibon.org/wiki/v/Defining_and_Sizing_the_Industrial_Internet.** (Mayıs 2019)

[3] Prof. Dr. Haldun Akpınar, *DATA Veri Madenciliği Veri Analizi*.: PapatyaBilim, 2017.

[4] Kerim Kürşat Çevik and Emre Dandıl, "Yapay sinir ağları için.NET platformunda görsel bir eğitim yazılımının geliştirilmesi," **Bilim Teknolojileri Dergisi, vol. 5**, no. 1, pp. 19-28, Nisan 2012.

[5] Dervis Karaboga, "An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report," Erciyes University, Kayseri, 2005.

[6] Dervis Karaboga and Bahriye Akay, "A comparative study of artificial bee colony algorithm," **Applied Mathematics and Computation, 1**, no. 214, pp. 108-132, August 2009.

[7] Derviş Karaboğa and Bahriye Akay, "Artificial Bee Colony (ABC) Algorithm on Training Artificial Neural Networks," , Eskişehir, 2007.

[8] Celal Öztürk, Yapay sinir ağlarının yapay arı kolonisi algoritması ile eğitilmesi, 2011.

[9] Mohammad Saeid Mahdavinejad et al., "Machine learning for internet of things data analysis: a survey," **Digital Communications and Networks, 4**, no. 3, pp. 161-175, August 2018.

[10] Marco Conti, Sajal K. Das, Chatschik Bisdikian, and Mohan, "Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber-physical convergence," **Pervasive and Mobile Computing, 8**, no. 1, pp. 2-21, 2012.

[11] Kevin Ashton. (2009) That 'Internet of Things' Thing.

[12] P.P Ray, "A survey on Internet of Things architectures," *Journal of King Saud University – Computer and Information Sciences*, vol. 30, no. 3, July 2018.

[13] **https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html.** (Mayıs 2019)

[14] Muhammed Zekeriya Gündüz and Resul Dağ, "Nesnelerin interneti: GeliÇimi, bileşenleri ve uygulama alanları," vol. 24, no. 2, pp. 327-335, 2018.

[15] Rishika Mehtaa, Jyoti Sahni, and Kavata Khanna, "Internet of Things: Vision, Applications and Challenges," **Procedia Computer Science, 132**, no. 2018, pp. 1263-1269, 2018.

[16] Mustafa Çakır Okan Oral, "Nesnelerin İnterneti Kavramı ve Örnek Bir Prototipin Oluşturulması," **Mehmet Akif Ersoy Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 1**, pp. 172-177, Kasım 2017.

[17] Charalampos Konstantopoulos, Paolo Bellavista, Chi-Fu Huang, and Damla Turgut, "Reactive wireless sensor networks," **Computer Communications, 36**, no. 9, pp. 963-1100, 2013.

[18] Jiming Chen, Hannes Frey, and Xu Li, "Special Issue: Wireless Sensor and Robot Networks: Algorithms and Experiments," **Computer Communications, 35**, no. 9, pp. 1017-1164, 2012.

[19] Erwa Qin, Yoanna Long, Chenghong Zhang, and Lihua Huang, "Cloud Computing and the Internet of Things: Technology Innovation in Automobile Service," in *International Conference on Human Interface and the Management of Information*, Berlin, Heidelberg, 2013, pp. 173-180.

[20] Yonghui Zhang, Baodan Chen, and Xinning Lu, "Intelligent Monitoring System on Refrigerator Trucks Based on the Internet of Things," in *Wireless Communications and Applications*, Berlin, Heidelberg, 2012, pp. 201-206.

[21] S. Sofana Reka and Tomislav Dragicevic, "Future effectual role of energy Delivery: A comprehensive review of Internetof Things and smart grid," **Renewable and Sustainable Energy Reviews, 91**, pp. 90-108, 2018.

[22] Hemant Ghayvat, S.C. Mukhopadhyay, Arun Babu, and Md Eshrat E Alahi, "Internet of Things: Vision, Applications and Challenges," **Australian Journal of Telecommunications and the Digital Economy, 3**, no. 4, pp. 31-47, 2015.

[23] Biljana Stojkoska and Kire Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," **Journal of Cleaner Production, 140**, no. 3, pp. 1454-1464, 2017.

[24] Tolga Kılıç and Esra Bayır, "An Investigation on Internet of Things Technology (IoT) In Smart Houses," **Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi , 9**, no. 3, pp. 196-206, 2017.

[25] Selma Dilek, "Nesnelerin İnterneti Tabanlı Uzaktan Sağlık İzleme Uygulaması ," Gazi Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, 2017.

[26] Haruka Ishii, Keisuke Kimino, Maher Aljehani, Nobuhiro Ohe, and Masahiro Inoue, "An Early Detection System for Dementia using the M2M/IoT Platform," **Procedia Computer Science , 96**, pp. 1332 – 1340, 2016.

[27] Fadi M.Al-Turjmana, Hossam S.Hassaneina, and Mohamed A. Ibnkahla, "Efficient deployment of wireless sensor networks targeting environment monitoring applications," **Computer Communications, 36**, no. 2, pp. 135-148, 2013.

[28] WEI Qiuping, ZHU Shunbinga, and DU Chunquan, "Study on key technologies of Internet of Things perceiving mine," **Procedia Engineering, 26**, pp. 2326-2333, 2011.

[29] In Lee and Kyoochun Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," **Business Horizons, 58**, no. 4, pp. 431-440, 2015.

[30] Abdullah Baykal, "Veri Madenciliği Uygulama Alanları ,"**D.Ü.Ziya Gökalp Eğitim Fakültesi Dergisi, 7**, pp. 95-107, 2006.

[31] Serkan Kaya Ömer Akgöbek, "Veri madenciliği teknikleri ile veri kümelerinden bilgi keşfi," **e-Journal of New World Sciences Academy, 6**, no. 1, pp. 237-245, 2011.

[32] Sholom M. Weiss and Nitin Indurkhya, *Predictive Data Mining: (A Practical Guide The Morgan Kaufmann Series in Data Management Systems)*., 1998.

[33] Tom M. Mitchell, *Machine Learning*.: McGraw-Hill S cience / Engineering / Math, 1997.

[34] Feng Chen et al., "Data Mining for the Internet of Things: Literature Review and Challenges," **International Journal of Distributed Sensor Networks, 9**, pp. 1-14, 2015.

[35] Shweta Bhatia and Sweety Patel, "Analysis on different Data mining Techniques and algorithms used in IoT," **t. Journal of Engineering Research and Applications, 5**, no. 11, pp. 82-85, 2015.

[36] Shen Bin, Liu Yuan, and Wang Xiaoyi, "Research on data mining models for the internet of things," in *2010 International Conference on Image Analysis and Signal Processing*, 2010, pp. 127-132.

[37] Zhenjun Li, "A Data Classification Algorithm of Internet of Things Based on Neural Network," *International Journal of Online Engineering (iJOE)*, vol. 13, no. 9, pp. 28-37, 2017.

[38] S. M. Kamruzzaman and A. M. Jehad Sarkar, "A New Data Mining Scheme Using Artificial Neural Networks," **Sensors, 11**, pp. 4622-4647, 2011.

[39] Smita Nirkhi, "Potential use of Artificial Neural Network in Data Mining," , 2010.

[40] Furqan Alam, Rashid Mehmood, Iyad Katib, and Aiiad Albeshri, "Analysis of Eight Data Mining Algorithms for Smarter Internet of Things (IoT)," **Procedia Computer Science, 98**, pp. 437-442, 2016.

[41] Dr. Yashpal Singh and Alok Singh Chauhan, "Neural networks in data mining,"**Journal of Theoretical and Applied Information Technology, 5**, no. 6, pp.37-42, 2009.

[42] Zheng Liu and Clair J. Sullivan, "Prediction of weather induced background radiation fluctuation with recurrent neural network," **Radiation Physics and Chemistry, 155**, pp. 275-280, February 2019.

[43] Bahadır Catalbas, "Recurrent neural network learning with an application to the control of legged locomotion," Bilkent University Instutional Repository , Master Theses - Department of Electrical and Electronics Engineering 2015.

[44] Hisashi Shimodaira, "Time-Series Prediction," in *Expert Systems The Technology of Knowledge Management and Decision Making for the 21st Century*.: Academic Press, 2002, ch. 4, pp. 1295-1311.

[45] A.J.P. Samarawickrama and T.G.I. Fernando, "A recurrent neural network approach in predicting daily stock prices an application to the Sri Lankan stock market," in *2017 IEEE International Conference on Industrial and Information Systems (ICIIS)*, Peradeniya, Sri Lanka, 2017.

[46] Aini Hanifa and Saiful Akbar, "Detection of unstable approaches in flight track with recurrent neural network," in *2018 International Conference on Informationand Communications Technology (ICOIACT)*, Yogyakarta, Indonesia, 2018, pp. 735-740.

[47] Ruben Zazo, Phani Sankar Nidadavolu, Nanxin Chen, Joaquin Gonzalez- Rodriguez, and Najim Dehak, "Age Estimation in Short Speech Utterances based on LSTM Recurrent Neural Networks," *IEEE Access*, vol. 99, pp. 22524-22530, March 2018.

[48] Yinyan Zhang, Shuai Li, Bolin Liao, Long Jin, and Linsen Zheng, "A recurrent neural network approach for visual servoing of manipulators," in *IEEE International Conference on Information and Automation (ICIA)*, Macau, China, 2017, pp. 614-619.

[49] Yuxuan Zhang et al., "A recurrent neural network based method for predicting the state of aircraft air conditioning system," in *Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, USA, 2017, pp. 1-7.

[50] P. Abhigna, S. Jerritta, R. Srinivasan, and V. Rajendran, "Analysis of feed forward and recurrent neural networks in predicting the significant wave height at the moored buoys in Bay of Bengal," in *International Conference on*

*Communication and Signal Processing (ICCSP)*, Chennai, India, 2017, pp. 1856-1860.

[51] V.B. Nemirovskiy and A.K. Stoyanov, "Multi-step segmentation of images by means of a recurrent neural network," in *8th International Forum on Strategic Technology · IFOST*, Ulaanbaatar, Mongolia, 2013, pp. 221-224.

[52] Mingxin Yu, Xiaoying Tang, Yingzi Lin, and Xiangzhou Wang, "Diesel engine modeling based on recurrent neural networks for a hardware-in-the-loop simulation system of diesel generator sets," **Neurocomputing, 283**, pp. 9-19, March 2018.

[53] Yanjun Shen, Yu Yu, Peng Miao, Hui Yu, and Shaocheng Qu, "Energy efficiency optimization of belt conveyors based on finite-time recurrent neural networks," in *11th Asian Control Conference (ASCC)*, Gold Coast, QLD, Australia, 2017, pp. 1548-1553.

[54] Yangming Li et al., "Improving Control Precision and Motion Adaptiveness for Surgical Robot with Recurrent Neural Network," in *International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, 2017, pp. 3538-3343.

[55] Hayder Wafa, Abid Aicha, and Ben Hamed Mouna, "Modeling of a photovoltaic cell based on recurrent neural networks," in *International Conference on Green Energy Conversion Systems (GECS)*, Hammamet, Tunisia, 2017, pp. 1-5.

[56] Liu Yunpeng, Hou Di, Bao Junpeng, and Qi Yong, "Multi-step Ahead Time Series Forecasting for Different Data Patterns Based on LSTM Recurrent Neural Network," in *14th Web Information Systems and Applications Conference (WISA)*, Liuzhou, China, 2017, pp. 305-310.

[57] Ryan Amirkhan, Mark Hoogendoorn, Mattijs E. Numans, and Leon Moons, "Using recurrent neural networks to predict colorectal cancer among patients," in *Symposium Series on Computational Intelligence*, Honolulu, HI, USA, 2017, pp. Honolulu, HI, USA.

[58] Qiangqiang Cui, Zhiheng Li, Jun Yang, and Bin Liang, "Rolling bearing fault prognosis using recurrent neural network," in *29th Chinese Control And Decision Conference (CCDC)*, Chongqing, China, 2017, pp. 1196-1201.

[59] Yuan Lukito and Antonius Rachmat Chrismanto, "Recurrent Neural Networks Model for WiFi-based Indoor Positioning System," in *International Conference on Smart Cities, Automation & Intelligent Computing Systems* , Yogyakarta, Indonesia, 2017, pp. 121-125.

[60] Kalman Tornai et al., "Recurrent Neural Network Based User Classification for Smart Grids," in *IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Washington, DC, USA, 2017.

[61] Dervis Karaboga, Bahriye Akay, and Celal Ozturk, "Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks," in *International Conference on Modeling Decisions for Artificial Intelligence (MDAI)*, Berlin, Heidelberg, 2007, pp. 318-329.

[62] Dervis Karaboga and Celal Ozturk, "Neural networks training by artificial bee colony algorithm on pattern classification," *Neural Network World*, vol. 19, no. 3, pp. 279-292, January 2009.

[63] Tsung-Jung Hsieh, Hsiao-Fen Hsiao, and Wei-Chang Yeh, "Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm," *Applied Soft Computing (Appl Soft Comput)*, vol. 11, no. 2, pp. 2510–2525, March 2011.

[64] Derviş Karaboğa and Bahriye Baştürk, "On the performance of artificial bee colony (ABC) algorithm," **Applied Soft Computing, 8**, no. 1, pp. 687-697, January 2008.

[65] BahramHajimirzaei and Nima Jafar iNavimipour, "Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm," **ICT Express, 5**, no. 1, March 2018.

[66] Ozgur Kisi, Coskun Ozkan, and Bahriye Akay, "Modeling discharge–sediment relationship using neural networks with artificial bee colony algorithm," Journal of Hydrology, 428-429, pp. 94-103, March 2012.

[67] Erol Eğrioğlu Faruk Alpaslan, Çağdaş Hakan Aladağ, Damla İlter, and Ali Zafer Dalar, "Comparıson Of Sıngle Multıplıcatıve Neuron Artıfıcıal Neural Network Models Usıng Abc And Bp Traınıng Algorıthms," **Anadolu Üniversitesi Bilim ve Teknoloji Dergisi A-Uygulamalı Bilimler ve Mühendislik, 14,** no. 3, pp. 315-328, 2013.

[68] Rasoul Irani and Reza Nasimi, "Application of artificial bee colony-based neural network in bottom hole pressure," **Journal of Petroleum Science and Engineering, 78**, no. 1, pp. 6-12, July 2011.

[69] Celal Ozturk and Dervis Karaboga, "Hybrid Artificial Bee Colony algorithm for neural network training," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC*, New Orleans, LA, USA, 2011, pp. 84-88.

[70] Waheed Ghanem and Aman Jantan, "Using hybrid artificial bee colony algorithm and particle swarm optimization for training feed-forward neural networks," **Journal of Theoretical and Applied Information Technology, 67**, no. 3, pp. 664-674, September 2014.

[71] Habib Shah, Tutut Herawan, Rashid Naseem, and Rozaida Ghazali, "Hybrid Guided Artificial Bee Colony Algorithm for Numerical Function Optimization," in *ICSI 2014: Advances in Swarm Intelligence*, 2014, pp. 197-206.

[72] Priyanka Gaur, "Neural Networks in Data Mining," **International Journal of Electronics and Computer Science Engineering, 1**, no. 3, pp. 1449-1453, 2012.

[73] Peter Tino, Lubica Benuskova, and Alessandro Sperduti, "Artificial Neural Network Models," in *Handbook Of Computational Intelligence*.: Springer, 2015, pp. 455-471.

[74] Süleyman Canan, "Yapay Sinir Ağları İle GPS Destekli Navigasyon Sistemi," T.C Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya, Doktora Tezi 2006.

[75] M. L. Koç and A. Balas C. E. & Arslan, "Taş Dolgu Dalgakıranların Yapay Sinir Ağları İle Ön Tasarımı," **ĠMO Teknik Dergi, 15**, no. 74, pp. 3351-3375, 2004.

[76] Chandan Dasgupta, "Mathematical modelling of neurons and neural networks," **Current Science Association, 68**, no. 2, pp. 162-167, 1995.

[77] Juan Antonio Sánchez Mesa, Carmen Galán, and Cesar Hervás Martínez, "The use of discriminant analysis and neural networks to forecast with a typical Mediterranean climate," **International Journal of Biometeorology, 49**, no. 355-362, 2005.

[78] İbrahim Çayıroğlu. (2015) **http://www.ibrahimcayiroglu.com/Dokumanlar/IleriAlgoritmaAnalizi/Ileri AlgoritmaAnalizi-5.Hafta-YapaySinirAglari.pdf.** (Mayıs 2019)

[79] Adnan Fatih Kocamaz Zafer Cömert, "A study of artificial neural network training algorithms for classification of cardiotocography signals," **Bıtlıs Eren Unıversıty Journal Of Scıence And Technology, 7**, no. 2, pp. 93-103, 2017.

[80] Prof. Dr. Ercan Öztemel, *Yapay Sinir Ağları*. İstanbul, Ankara, İzmir: Papatya Yayıncılık, 2016.

[81] Darío Baptista and Fernando Morgado-Dias, "A survey of artificial neural network training tools," **Neural Computing and Applications, 23**, no. 3-4, pp. 609- 615, 2013.

[82] Jerome T. Connor, R. Douglas Martin, and L.E Atlas, "Recurrent Neural Networks and Robust Time Series Prediction ," *IEEE Transactions on Neural Networks* , vol. 5, no. 2, Mayıs 1994.

[83] Sumit Goyal and Gyandera Kumar Goyal, "Cascade and Feedforward Backpropagation Artificial Neural Network Models for Prediction of Sensory Quality Instant Coffee Flavoured Sterilized Drink," **Canadian Journal on Artificial Intelligence, Machine Learning and Pattern Recognition, 2**, no. 6, 2011.

[84] David Rumelhart, David Zipser, and James L. McClelland, "Parallel Distributed Processing," *MIT Press*, vol. 1, pp. 151–193, 1986.

[85] Mathworks Documentation. [Online]. **https://www.mathworks.com/help/deeplearning/ug/cluster-with-a-**

**competitive-neural-network.html** (Mayıs 2019)

[86] Priyanka Tyagi and Dr. Jayant Shekhar, "A Cpmparative Study of Elman Backpropagation Neural Network and Radial Basis Function Neural Network Models for Speech Recognition ," **International Journal of Engineering Applied Sciences and Technology, 1**, no. 9, pp. 64-72, 2016.

[87] "Modelling the backpropagation algorithm of the Elman neural network by a generalized net," in *13th Int. Workshop on Generalized Nets*, Londra, 2012.

[88] P K Sahoo, P K Satpathy, and M N Mohanty, "Sources, Elman Neural Network Backpropagation Based Evaluation of Critical Busbars in Power Systems with Renewable," **Internatıonal Journal Of Renewable Energy Research, 5**, no. 2, pp. 533-541, 2015.

[89] MathWorks Documentation. [Online]. **https://www.mathworks.com/help/deeplearning/ref/timedelaynet.html** (Mayıs 2019)

[90] Hisashi Shimodaira, "Time-Series Prediction," in *Expert Systems.*, 2002, pp. 1259- 1313.

[91] Hava T. Siegelmann, Bill G. Horne, and C. Lee Giles, "Computational Capabilities of Recurrent NARX Neural Network," **IEEE Transactıons On Systems, Man, And Cybernetıcs, 27**, no. 2, pp. 208-215, 1997.

[92] Jose Maria P., Menezes Jr, and Guilherme A. Barreto, "Long-term time series prediction with the NARX network: An empirical evaluation," **Neurocomputing, 71**, no. 16-18, pp. 3335-3343, 2008.

[93] Sixte De Maupeou. Introduction To Hopfield Neural Networks. [Online]. **https://www.doc.ic.ac.uk/~sd4215/hopfield.html** (Mayıs 2019)

[94] Guoqiang Zhang, B. Eddy Patuwo, and Michael Y. Hu, "Forecasting with artificial neural networks:The state of the art," **International Journal of Forecasting, 14**, no. 1, pp. 35-62, 1998.

[95] Donald F.Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, no. 1, pp. 109-118.

[96] Hikmet Kerem Cigizoglua and Murat Alp, "Generalized regression neural network in modelling river sediment yield," **Advances in Engineering Software, 37, 2**, pp. 63-68, 2006.

[97] Selahattin Yavuz and Muhammet Deveci, "İstatiksel Normalizasyon Tekniklerinin Yapay Sinir Ağın Performansına Etkisi,"**Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 1**, no. 40, pp. 167-187, 2012.

[98] Derviş Karaboğa, *Yapay Zeka Optimizasyon Algoritmaları*.: Nobel, 2017.

[99] Engin Taş, "Geliştirilmiş Yapay Sinir Ağı Algoritmaları ve Uygulamaları," Anadolu Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı, Yüksek Lisans Tezi 2008.

[100] Mammadagha Mammadov, Berna Yazici, Şenay Yolaçan, Atilla Aslanargun, and Ali Fuat Yüzer, "Statistical Methods and Artificial Neural Networks Methods," **Journal of Modern Applied Statistical, 5**, no. 2, pp. 495-512, 2005.

[101] I.E. Livieris and P. Pintelas, "A survey on algorithms for training artificial neural networks," Department of Mathematics, University of Patras, Teknik Rapor 2008.

[102] Mehmet Fatih Amasyalı, "Yapay Sinir Ağlarında Parçalı Eğitim Mini-Batching for Artificial Neural Network Training," **Türkiye Bilişim Vakfı Bilgisayar Bilimleri Ve Mühendisliği Dergisi, 8**, no. 1, pp. 25-33, 2015.

[103] Howard Demuth, Mark Beale, and Martin Hagan. (2009) Neural Network Toolbox™ 6 User‟s Guide.

[104] Y. Müh. Murat Ermiş, "Lojistik Sistemlerin Yapay Sinir Ağları İle Modellenmesi, Gerçeklenmesi ve Kontrolü," İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Doktora Tezi 2005.

[105] Alan Mosca and GeorgeD. Magoulas. (2015, Ekim) **https://arxiv.org/abs/1509.04612.** (Mayıs 2019)

[106] Rodica Constantinescu, Vasile Lazararescu, and Radwan Tahboub, "Geometrıcal Form Recognıtıon Usıng "One-Stepsecant" Algorıthm In Case Of Neural Network," **U.P.B. Sci. Bull., Series C, 70**, no. 2, pp. 15-28, 2008.

[107] Christopher M. Bishop, *Neural Networks for Pattern Recognition.*: Oxford Press, 1995.

[108] Maja Lunde, *The History of Bees: A Novel (English Edition).*, 1927.

[109] Bahriye Akay and Derviş Karaboğa, "Artificial bee colony algorithm for large-scale problems and engineering design optimization," **Journal of Intelligent Manufacturing, 23**, no. 4, pp. 1001-1014, August 2012.

[110] Maxim Vladimirovich Shcherbakov et al., "A Survey of Forecast Error Measures," **World Applied Sciences Journal, 24**, no. 24, pp. 171-176, 2013.

[111] M. Hashem Pesaran, "Regression and correlation analysis. New Palgrave: A Dictionary of Economics," **A Dictionary of Economics, 4**, no. 1, pp. 120-123, 1987.

[112] Üstün Özen Fulya Aslay, "Meteorolojik Parametreler Kullanılarak Yapay Sinir Yapay Sinir Ağları ile Toprak Sıcaklığının Tahmini," **Politeknik Dergisi, 16**, no. 4, pp. 139-144, 2013.

[113] Yapay Sinir Ağları ve Destek Vektör Makineleri Yöntemleriyle Borsa Endeksi Tahmini, "Yrd. Doç. Dr. Emre Yakut; Doç. Dr. Bekir Elmas; Yrd. Doç. Dr. Selahattin Yavuz," **Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Dergisi, 19**, no. 1, pp. 139-157, 2014.

[114] **https://www.qlik.com/us/products/qlik-data-market?q=internet+traffic+data+price%3Afree.** (Nisan 2019)

[115] Time Series Data Library. **https://datamarket.com/data/set/22l0/monthly-electricity-production-in-australia-million-kilowatt-hours-jan-1956-aug-1995.** (Mayıs 2019)

[116] **https://www.kaggle.com/fvcoppen/solarpanelspower.** (Mayıs 2019)

[117] **https://archive.ics.uci.edu/ml/machine-learning-databases.** (Mayıs 2019)

[118] Ricardo Pinto Ferreira, Carlos Affonso, and Renato José Sassi, "Combination of artificial intelligence techniques for prediction the behavior of urban vehicular traffic in the city of sao paulo," , Fortaleza, Ceará Brazil, 2011.

[119] Ananda Freire, Marcus Veloso, and Guilherme Barreto. **https://archive.ics.uci.edu/ml/datasets/Wall-Following+Robot+Navigation+Data.** (Mayıs 2019)

[120] Luis Miguel Candanedo Ibarra and Veronique Feldheim, "Accurate occupancy detection of and office room from light, temperature, humidity and CO2 measurements using statistical learning models," **Energy and Buildings, 112**, no. 2016, pp. 28-39, December 2016.

[121] Rakesh Ranjan Swain, Tanistha Nayak Tirtharaj Dash, "Automatic navigation of wall-following mobile robot using a hybrid metaheuristic assisted neural network," , 2017, pp. 1-17.

[122] Tirtharaj Dash, Rakesh Ranjan Swain, and Tanistha Nayak, "Automatic navigation of wall-following mobile robot using a hybrid metaheuristic assisted neural network," **Data Science, 1**, pp. 1-17, 2017.

[123] Tirtharaj Dash, Tanistha Nayak, and Rakesh Ranjan Swain, "Controlling Wall Following Robot Navigation Based on Gravitational Search and Feed Forward Neural Network," in *Proceedings of the 2nd International Conference on Perception and Machine Intelligence*, New York, 2015, pp. 196-200.

[124] Tirtharaj Dash, Tanistha Nayak, Rakesh Ranjan Sahu, and Mishra, "Neural network approach to control wall-following robot navigation," in *Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference*, Ramanathapuram, India, 2014, pp. 1072-1076.

[125] Kemal Tütüncü, Özcan Çataltaş, and Murat Köklü, "Occupancy Detection Through Light, Temperature, Humidity and CO2 Sensors Using ANN," , Rabat, Morocco, 2016, pp. 16-20.

# EKLER

## EK 1. Algoritmanın Matlab Kodu

**ABC Algoritmasının RNN Ağının Eğitimi İçin Kullanılması**

```
clear all
rand('state',sum(100*clock));
PID=13;
```

**Problem Fonksiyonundan Değerlerin Alınması**

```
fhd=@Problem;
```

**Problemin Kayıtlı Olduğu Şekilde Çağrılması**

```
if (PID==1)
  load data1;
  num_input = 4;
  num_hidden = 10;
  num_output = 1;
elseif (PID==2)
  load data2;
  num_input = 4;
  num_hidden =10;
  num_output = 1;
elseif (PID==3)
  load data3;
  num_input = 2;
  num_hidden = 10;
  num_output = 1;
elseif (PID==4)
  load data4;
  num_input = 16;
  num_hidden =10 ;
  num_output = 1;
elseif (PID==5)
```

```
 load data5;
 num_input = 4;
 num_hidden = 10;
 num_output = 1;
end;
```

**Optimize Edilecek Parametre Sayısının Belirlenmesi**

```
D=num_input*num_hidden+num_hidden+num_hidden*num_hidden+num_hidden*nu
m_output+num_output;
%D=2-20-1:481
```

**ABC Algoritmasının Parametrelerinin Girilmesi**

```
Xmin=-3;
Xmax=3;
NP=50;
```

**İterasyon ve Koşma Sayısının Girilmesi**

```
maxCycle=1000;
runs=30;
for j=1:runs
 Karistir_Train_Test_Ayir;
 ABC_func;
```

**Belirlenen Hata Kriteri İçin Son İterasyondan Sonra Üretilen Eğitim, Test Çıktı ve Hata Değerlerinin Alınması**

```
 [fbest_train(j),sondeger_train]=feval(fhd,gbest,num_input,
num_hidden,num_output,0,0,train_input_data, train_output_data, test_input_data,
test_output_data);

 %Test Error
[fbest_test(j),sondeger_test]=feval(fhd,gbest,num_input,
num_hidden,num_output,0,1,train_input_data, train_output_data, test_input_data,
test_output_data);
end
```

```
%Train Max, Min, Std Error
f_mean_train=mean(fbest_train);
f_min_train=min(fbest_train);
f_max_train=max(fbest_train);
f_std_train=std(fbest_train);

%Test Max, Min, Std Error
f_mean_test=mean(fbest_test);
f_min_test=min(fbest_test);
f_max_test=max(fbest_test);
f_std_test=std(fbest_test);

save (['VS4_smape',  num2str(PID), '.mat']);
```

**Problem Fonksiyonu**

```
function [sonuc,y_k]= Problem(p, num_input,
num_hidden,num_output,wyaz,ttip,train_input_data, train_output_data, test_input_data,
test_output_data)

 np=size(p,1);
```

**Ağın Eğitim ve Test Verilerinin Ayrılması**

```
  if (ttip==0)  %train
    input_data = train_input_data;
    output_data = train_output_data;
else       %test
    input_data = test_input_data;
    output_data = test_output_data;
  end

  for i=1:np
    x=p(i,:);
```

**RNN Ağı İçin Ağırlıkların Hesaplanması**

```
    wbas = 1;
    wbit = num_input*num_hidden;
    w_ij=reshape(x(wbas:wbit),num_input,num_hidden);

    wbas = wbit+1;
    wbit = wbit+num_hidden*1;
    b_j =reshape(x(wbas:wbit),num_hidden,1);

    wbas = wbit+1;
    wbit = wbit+num_hidden*num_hidden;
    w_hj=reshape(x(wbas:wbit),num_hidden,num_hidden);

    wbas = wbit+1;
    wbit = wbit+num_hidden*num_output;
    w_jk=reshape(x(wbas:wbit),num_hidden,num_output);

    wbas = wbit+1;
    wbit = wbit+num_output*1;
    b_k =reshape(x(wbas:wbit),num_output,1);

s_j     = zeros(num_hidden, 1);
h_prev  = zeros(num_hidden, 1);
s_k     = zeros(num_output, 1);
y_k     = zeros(num_output, 1);
    e_k     = zeros(size(output_data));

    for data=1:size(input_data,1)
        % Forward propagation
        x_data  = input_data(data,:)'; % 2x1
        s_j     = (x_data'*w_ij)' + b_j + (h_prev'*w_hj)';
h_j     = sigmoid(s_j, num_hidden);
```

```
h_prev= h_j;
s_k     = (h_j'*w_jk)' + b_k;
       y_k(data)  = sigmoid(s_k, num_output);
       e_k(data)= abs(output_data(data) - y_k(data));
      p_k(data)=(output_data(data) + y_k(data));
      if (p_k(data)==0)
        p_k(data)=0.0000000000000001
      end
     % enc_dec
      %  y_k     = sigmoid(s_k, num_output);
      %e_k(data,:)= output_data(data,:) - y_k';
      %p_k(data,:)=(output_data(data,:) + y_k');
      if (wyaz==1)
         fprintf('Input: %.1fl, %.1f, Output: %.2f\n',x_data(1), x_data(2), y_k);
       end

   end
```

**İlgili Hata Değerlerinin Hesaplanması**

```
   %RMS hesaplama
    %J = (sum(e_k'*e_k))/length(input_data);
   %sonuc(i)=sqrt (J);

  %SMAPE hesaplama
    J = (2/length(input_data))*sum((abs(e_k))'/p_k);
   sonuc(i)=J;
  %J = (2/length(input_data))*abs(sum(sum((abs(e_k))/p_k)));
   %sonuc(i)=J;

  %MAE hesaplama
  %J = sum(sum(abs(e_k.*ones(size(e_k)))/size(input_data,1)));
 % sonuc(i)=J;
 end

end
```

# ÖZGEÇMİŞ

## KİŞİSEL BİLGİLER

**Adı Soyadı:** Ebru YÖNEM
**Uyruğu:** Türkiye (T.C)
**Doğum Tarihi ve Yeri:** 04.11.1994 -Kayseri
**Medeni Durum:** Bekar
**e-mail:** yonemebru@gmail.com
**Yazışma Adresi:** Devlet Mahallesi Süleyman Emin Caddesi
No:6-7 06420 Çankaya/Ankara

## EĞİTİM

| Derece | Kurum | Mezuniyet Tarihi |
|---|---|---|
| Lisans | Erciyes Üniversitesi Mekatronik Mühendisliği | 2016 |
| Lise | Ahmet Eren Anadolu Lisesi Kayseri | 2012 |

## İŞ DENEYİMLERİ

| Yıl | Kurum | Görev |
|---|---|---|
| 2019-Halen | T.C. Cumhurbaşkanlığı Savunma Sanayii Başkanlığı | Proje Mühendisi |
| 2018-2019 | Ziraat Teknoloji A.Ş | İş Analisti |

## YABANCI DİL

İngilizce