ANKARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

ÇEVRİM İÇİ SIRALI ÖĞRENME İÇİN YENİ BİR MODEL ÖNERİSİ: OS-CELM

Olcay TOSUN

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ANKARA
2020

Her hakkı saklıdır

ÖZET

Yüksek Lisans Tezi

ÇEVRİM İÇİ SIRALI ÖĞRENME İÇİN YENİ BİR MODEL ÖNERİSİ: OS-CELM

Olcay TOSUN

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Recep ERYİĞİT

Veri boyutlarının büyümesi ve verinin hızla yenilenmesi sınıflandırma problemlerinde çevrim içi öğrenmeyi daha fazla kullanır hale getirmiştir. Hızlı model oluşturma konusunda popülerliğini koruyan ELM tabanlı OS-ELM çevrim içi öğrenme alanında sıklıkla kullanılmaktadır. ELM tabanlı algoritmaların gizli katman parametrelerini rastgele ataması gizli katmanların etkili kullanımını engellemektedir. Bu tez çalışmasında gizli katman parametrelerinin rastgele atanması yerine sınıflar arası uzaklığı baz alan CELM yöntemini kullanarak daha iyi genelleme performansına sahip Çevrim İçi Sıralı Kısıtlı Aşırı Öğrenme Makinesi (OS-CELM) isminde yeni bir çevrim içi öğrenme modeli sunulmuştur. OS-CELM modelinin ve OS-ELM modelinin üç farklı veri seti üzerinde başarım oranları ve eğitim süreleri karşılaştırılmıştır. Karşılaştırmaların sonucunda önerilen modelin daha iyi başarım oranlarının olduğu görülmüştür.

Eylül 2020, 44 sayfa

Anahtar Kelimeler: Makine Öğrenmesi, Aşırı Öğrenme Makinesi (ELM), Kısıtlı Aşırı Öğrenme Makinesi (CELM), Çevrim İçi Sıralı Kısıtlı Aşırı Öğrenme Makinesi (OS-CELM)

ABSTRACT

M. Sc. Thesis

A NEW MODEL FOR ONLINE SEQUENTIAL LEARNING: OS-CELM

Olcay TOSUN

Ankara University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Recep ERYİĞİT

The growth of data sizes and the rapid renewal of data have made online learning more used in classification problems. Maintaining its popularity in model training speed, ELM-based OS-ELM is one of the most frequently used in online learning. Despite the fast training ability, random assignment of hidden layer parameters by ELM-based algorithms prevents the effective use of hidden layers. In this study, a new online learning model called Online Sequential Constrained Extreme Learning Machine (OS-CELM) with better generalization performance using the CELM method based on the distance between classes instead of random assignment of hidden layer parameters is presented. OS-CELM model and OS-ELM are compared on three different data sets, in terms of performance rates and training times. As a result of the comparisons, it is seen that the proposed model has better generalization and accuracy performance.

August 2020, 44 pages

Key Words: Machine Learning, Extreme Learning Machine (ELM), Constrained Extreme Learning Machine (CELM), Online Sequential CELM (OS-CELM)

TEŞEKKÜR

Rehberliğini ve desteklerini hiçbir zaman esirgemeyen, sadece eğitim ve akademik konularda değil aynı zamanda iş hayatı ile ilgili konularda da bilgisinden ve tecrübesinden ziyadesiyle faydalandığım, tanımaktan onur ve gurur duyduğum saygıdeğer danışman hocam Doç. Dr. Recep ERYİĞİT'e,

Yüksek lisansı eğitimim süresince gelişimime katkı sağlayan Bilgisayar Mühendisliğinde görev yapan bütün hocalarıma teşekkürü bir borç bilirim.

Varlığı ile hayatımın en önemli ve en güzel zamanlarını yaşatan, her başarımda en büyük desteği veren sevgili hayat arkadaşım Figen'e ve sevinç kaynağımız Deniz'e teşekkür ederim.

Olcay TOSUN
Sinop, Eylül 2020

İÇİNDEKİLER

TEZ ONAYI ..... **ii**
ETİK ..... **i**
ÖZET ..... **ii**
ABSTRACT ..... **iii**
TEŞEKKÜR ..... **iv**
KISALTMALAR DİZİNİ ..... **vi**
ŞEKİLLER DİZİNİ ..... **vii**
ÇİZELGELER DİZİNİ ..... **viii**
1. GİRİŞ **1**
2. KURAMSAL TEMELLER **4**
2.1 Makine Öğrenmesi **4**
2.2 Biyolojik Sinir Ağları **5**
2.3 Yapay Sinir Ağları **7**
2.4 İleri Beslemeli Yapay Sinir Ağları **10**
2.5 Geri Yayılım **11**
2.6 Aşırı Öğrenme Makinesi (ELM) **12**
2.7 Kısıtlı Aşırı Öğrenme Makinesi (CELM) **15**
2.8 Çevrim İçi Öğrenme (Online Learning) **17**
2.9 Çevrim İçi Sıralı Aşırı Öğrenme Makinesi (OS-ELM) **18**
3. YÖNTEM VE MATERYAL **21**
3.1 Önerilen Model **21**
3.2 Modelin Uygulanması **23**
3.3 Veri Setleri **25**
3.4 Veri Ön İşleme **29**
4. ARAŞTIRMA BULGULARI **30**
5. TARTIŞMA **37**
6. SONUÇ **40**
KAYNAKLAR **41**
ÖZGEÇMİŞ **44**

KISALTMALAR DİZİNİ

FNN	Feedforward Neural Network – İleri Beslemeli Sinir Ağları
BP	Back Propagation – Geri Yayılım
ELM	Extreme Learning Machine – Aşırı Öğrenme Makinesi
CELM	Constrained Extreme Learning Machine- Kısıtlı Aşırı Öğrenme Makinesi
OS-ELM	Online Sequential Extreme Learning Machine – Çevrim İçi Sıralı Aşırı Öğrenme Makinesi
OS-CELM	Online Sequential Constrained Extreme Learning Machine- Çevrim İçi Sıralı Kısıtlı Aşırı Öğrenme Makinesi
t-SNE	t-Distributed Stochastic Neighbor Embedding
ANFIS	Adaptive Neuro Fuzzy Inference Systems
GPU	Graphic Process Unit – Grafik İşlemci Ünitesi

ŞEKİLLER DİZİNİ

Şekil 2.1 Renklendirilmiş nöron dokusu (ACC,2005) 6
Şekil 2.2 Şematik nöron yapısı (Negnevitsky,2005) 6
Şekil 2.3 Yapay sinir ağı 7
Şekil 2.4 Perceptron (Nilsson ,2005) 8
Şekil 2.5 Geri Yayılım 11
Şekil 2.6 ELM 13
Şekil 3.1 Sınıflar arası vektörler 21
Şekil 3.2 Birbirine yakın veriler 22
Şekil 3.3 Akış diyagramı 24
Şekil 3.4 Wifi indoor localization veri seti 26
Şekil 3.5 Skin- no skin veri seti 27
Şekil 3.6 MNIST örnek rakamlar 28
Şekil 3.7 MNIST Veri Seti 28
Şekil 4.1 OS-ELM / OS-CELM (Wifi Localization) 31
Şekil 4.2 OS-ELM / OS-CELM (Skin/No Skin) 33
Şekil 4.3 OS-ELM / OS-CELM (MNIST) 35

ÇİZELGELER DİZİNİ

Çizelge 2.1 Biyolojik ve yapay sinir ağları 8
Çizelge 2.2 VE kapısı için perceptron eğitimi 9
Çizelge 3.1 Wifi indoor localization 25
Çizelge 3.2 Skin/No Skin 26
Çizelge 3.3 Skin-no skin ön işleme 29
Çizelge 4.1 Wifi indoor localization tur değerleri 30
Çizelge 4.2 Wifi Indoor Localization başarım oranları ve eğitim süreleri 31
Çizelge 4.3 Skin/no skin tur F_1 değerleri 32
Çizelge 4.4 Skin/no skin F_1 oranları ve eğitim süreleri 33
Çizelge 4.5 MNIST tur başarım oranları 34
Çizelge 4.6 MNIST başarım oranları ve eğitim süreleri 35
Çizelge 5.1 Wifi veri seti üzerinde model karşılaştırması 37
Çizelge 5.2 Skin veri seti üzerinde model karşılaştırması 38
Çizelge 5.3 MNIST veri seti üzerinde model karşılaştırması 39

1. GİRİŞ

Bilişim çağı olarak adlandırılan içinde yaşadığımız bu dönemde bilgisayar destekli otonom sistemler ve yapay zekâ uygulamaları hayatımızın her alanında kendini göstermektedir. Telefon bankacılığı işlemlerinde kullanılan otomatik yanıt sistemindeki ses ve konuşma tanımlamadan, e-posta uygulamalarında zararlı iletilerin ayıklanması ve engellenmesine, video görüntülerinden sıra dışı aktivitelerin tespit edilerek güvenlik birimlerine haber verilmesinden, binlerce haber kaynağından yayınlanan metinlerin sınıflarına göre ayrılmasına kadar hayatımızın kolaylaştıran her alanda yapay zekâ ve yapay zekanın alt alanı olan makine öğrenmesi metotları kullanılmaktadır. Verinin çok hızlı üretildiği ve aktarıldığı bu dönemde verilerin işlenmesi için daha hızlı ve karmaşık yapıların kullanılmasını zorunlu hale gelmiştir. 20. yüzyılın son çeyreğinden itibaren bilgisayar donanım üreticilerinin işlemci hızlarını ve çekirdek sayılarını artırmasıyla birlikte bilgisayar ve işlem gücü yüksek mobil cihazlar artan veri işleme ihtiyacına cevap verebilmeye çalışmaktadırlar.

Sıklıkla kullanılan uygulamalarda makine öğrenmesinin temel algoritmalarını görmek mümkündür. Plaka okuma sistemlerinde görüntü işleme ve sınıflandırma, hava durumu tahmininde ve market stok yönetiminde regresyon, sosyal medya kuruluşlarının yalan haber engelleme politikalarında kümeleme (clustering) metotları uç kullanıcılar tarafından farkında olmaksızın kullanılmaktadır. Sınıflandırma; eldeki verinin hangi kategoriye veya sınıfa ait olduğunu belirlemeye yarayan çalışmalardır. Mümkün olan en doğru şekilde yeni verileri doğru kategoriye atama veya sınıflandırma yapmak en temel amaçtır. Regresyon problemlerinde eldeki veriler matematiksel formüllere eşitlenerek yeni gelen verilerin sonuç değerlerini hesaplamayı amaçlamaktadır. Kümelemede ise amaç verilerdeki örüntüleri, benzerlikleri bularak verileri ayrıştırmaktır.

Problemlere çözüm arayışında insanoğlu doğadan esinlenmiştir. Makine öğrenmesinde de sınıflandırma ve regresyon problemlerinde sıklıkla kullanılan algoritmalardan bir tanesi olan yapay sinir ağları canlıların öğrenmesini model almıştır. Doğadaki canlıların özellikle insanın öğrenme yöntemi baz alınarak makinelere problem çözme kabiliyeti kazandırmayı amaçlayan bu yöntemde insan beyni makine ortamında canlandırılmıştır.

Rosenblatt'ın 1958 yılında bir yapay sinir ağının (perceptron) ilk eğitim algoritmasını yayınlamasından (Rosenblatt 1958) bu yana yapay sinir ağları araştırmacıların ilgi odağı olmuştur. Araştırmacılar yapay sinir ağlarının eğitimi için çok çeşitli yöntemler öne sürmüşlerdir. Bunların içinde en çok kullanılan yöntemler ileri beslemeli yapay sinir ağları (Feedforward Neural Networks) ve geri yayılım yapay sinir ağlarıdır (Back Propagation). İleri beslemeli yapay sinir ağı eğitiminde parametreler her bir iterasyonda hata oranında güncellenir. Geri yayılımda ise her bir iterasyon sonunda çıkış katmanından giriş katmanına doğru hata oranında parametreler güncellenerek bir sonraki iterasyona geçilir. Her iki yöntemin de eğitim hızları oldukça yavaştır. Bunun başlıca nedenleri yavaş gradyant tabanlı eğitim algoritmalarının yapay sinir ağlarında oldukça fazla kullanılması ve yapay sinir ağlarının parametrelerinin iteratif olarak bu algoritmaları kullanarak ayarlanmasıdır (Huang vd. 2004). Her ne kadar donanım üreticileri işlemci hızlarını ve çekirdek sayılarını artırmış olsalar da çevrim içi öğrenme gibi model eğitiminin veri akış hızına yakın gerçekleştirilmesi gerektiği veya verinin belleğe sığmadığı büyük veri uygulamaları gibi durumlarda ileri beslemeli ve geri yayılım yapay sinir ağlarında eğitim hızları yavaş kalmaktadır (Liang vd. 2006, Alpaydın 2010). Yapay sinir ağlarının eğitim metotlarının yavaş olmasının yanı sıra makine öğrenmesinin temel amaçlarından biri olan modellerin yüksek doğruluk oranına sahip olabilmesi için yapay sinir ağlarında nöron sayısı ve gizli katman sayısı artmaktadır. Bu durum ise eğitim hızının nöron sayısına bağlı olarak artışına sebep olmaktadır (Zhu, Miao ve Qing 2014).

Yapay sinir ağlarında eğitim süresini kısaltmak için 2004 yılında Huang ve arkadaşları Extreme Learning Machine (ELM) adını verdikleri ileri beslemeli yapay sinir ağları için yeni bir şema geliştirmişlerdir. ELM tek katmanlı ileri beslemeli bir yapay sinir ağıdır. ELM'de nöron parametrelerinin güncellenmesine gerek duyulmadığından eğitim süresi çok hızlıdır. Nöronların ağırlıkları ve diğer parametreleri rastsal olarak atanmaktadır. ELM'de lineer olmayan sistemler lineer sistemler gibi işlem görerek en küçük kareler yöntemi ile çözüme ulaşılmaktadır. Yapay sinir ağının hızlı eğitimine olanak sağlamasının yanı sıra diğer makine öğrenmesi algoritmalarıyla kıyaslanabilecek başarım oranına sahip olması ELM'in son yıllarda popülerliğini artırmaktadır (Huang,

Zhu ve Siew 2004, Huang, Ding ve Zhou 2010, Qiu, Gao ve Wang 2015, Chy ve Anisur Rahaman 2019).

ELM tabanlı birçok algoritma geliştirilmiştir. Deng vd. (2014) tarafından gömülü sistemlerde eğitim süresinin kısaltılması için geliştirdikleri RKELM, video görüntülerinin sınıflandırılması için Xu ve Wang (2016) tarafından geliştirilen IDS-ELM, büyük verilerde kullanılmak üzere Wang vd. (2017) tarafından geliştirilen DW-ELM bunlardan sadece bazılarıdır. Hızlı eğitim süresi çevrimiçi öğrenmede de avantaj yaratmaktadır. Liang ve arkadaşları (2006) tek gizli katmanlı ileri yönlü yapay sinir ağları için ELM tabanlı Online Sequential Extreme Learning Machine (OS-ELM) ismini koydukları bir yapı geliştirmişlerdir. OS-ELM sıralı bir şekilde, değişen veya sabit uzunluktaki verilerden hızlı eğitim gerçekleştirebilmektedir. Tüm veri setinin mevcut olduğu durumlarda ise ELM olarak model eğitimini yapmaktadır.

Hızlı eğitim süresine karşılık ELM ve ELM tabanlı diğer algoritmaların gizli katman parametrelerinin rastgele seçimi gizli katman nöronlarının etkili kullanımını engellemektedir (Zhu vd. 2005). Bu durum ise ELM ve ELM tabanlı algoritmaları performansını olumsuz etkilemektedir.

Yapılan bu tez çalışmasında çevrim içi ve sıralı öğrenim gerçekleştirebilen OS-ELM'in daha iyi bir sınıflandırma ve genelleme performansı sergilemesi için yeni bir model öne sürülmüştür. Önerilen bu yeni modele Online Sequential Constrained Extereme Learning (OS-CELM) ismi verilmiştir. Önerilen model 3. bölümde detaylı olarak anlatılmıştır. Önerilen modelin etkinliğini ölçmek için popüler MNIST, Skin Segmentation ve Wireless Indoor Localization veri setleri üzerinde başarım testleri yapılmıştır. Başarım test sonuçları OS-ELM ile karşılaştırılmıştır. Yapılan testler ve test sonuçları 4. bölümde gösterilmiştir. Önerilen model ve alanda yapılan diğer ELM tabanlı algoritmalarla ilgili değerlendirmeler 5. bölümde verilmiştir.

2. KURAMSAL TEMELLER

2.1 Makine Öğrenmesi

Karşılaşılan sorunlara doğadan çözümler aramak çok eski zamanlara dayanmaktadır. Alet kullanma yeteneği çok gelişmiş olan insanoğlu ilk çağlardan beri yaşam mücadelesinde karşılaştıkları sorunlar için çeşitli aletler tasarlamış ve kullanmıştır. Bu aletler zamanla basit el araçlarından daha karmaşık makinelere doğru bir gelişim göstermiştir. 2. Dünya Savaşında, Nazi Almanya'sı tarafından gizli mesajların şifrelenmesi ve tekrar çözülmesi amacı ile kullanılan Enigma şifreleme makinesinin kullanımı ve Enigma tarafından şifrelenen mesajların Alan Turing ve arkadaşları tarafından elektromanyetik bir makine ile çözülmesi bilgisayar bilimimin oluşmasındaki yapı taşları olmuştur. Turing (1950), "Hesaplama Makineleri ve Zekâ" adlı çalışmasında "Makineler düşünebilir mi?" sorusu ve tasarladığı taklit oyunu (imitation game) ile bilgisayar biliminin kurucusu ve yapay zekanın temellerini atan isim olarak literatürde yerini almıştır.

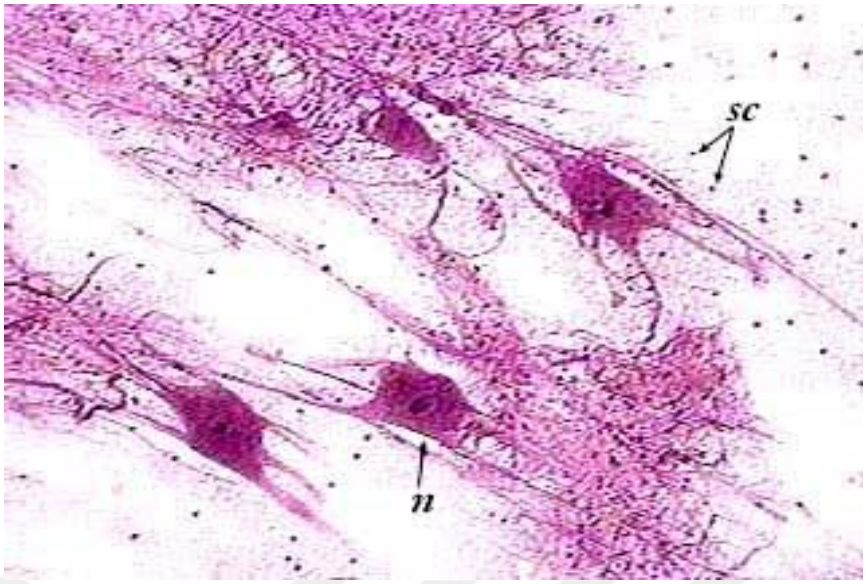
Yapay zekânın isim babası ve kurucusu olarak kabul edilen John McCarthy yapay zekayı biyolojik olarak gözlemlenebilir yöntemlerle kendisini sınırlamak zorunda olmayan, insan zekasının kullanımına benzer akıllı makineler özellikle de akıllı bilgisayar programları üretme bilimi ve mühendisliği olarak tanımlamıştır (*McCarthy, 2007*). Arthur Samuel 1959 yılında makine öğrenmesini geniş bir yapay zekâ alt alanı olarak ifade etmiştir. Makine öğrenmesi, örnek verileri veya geçmiş deneyimleri kullanarak problemlere çözüm bulmak için bilgisayarları programlamaktır (Alpaydın, 2010). Diğer bir değişle makine öğrenmesi, verideki anlamlı örüntülerin otomatik olarak algılanmasını sağlayan algoritmalar bütünüdür (Shai 2014).

Genel olarak makine öğrenmesi üç başlık altında toplanabilir. Bunlar denetimli öğrenme (supervised learning), denetimsiz öğrenme (unsupervised leaning) ve pekiştirmeli öğrenmedir (reinforcement learning). Denetimli öğrenme, etiketli verileri kullanarak karşılaşılan etiketsiz verilerin hangi kategoriye (sınıflandırma) ait olduğunu veya hangi değerleri içerebileceğini (regresyon) tahmin etmeye çalışmaktadır. Denetimli

öğrenmede mevcut olan etiketli veriler eğitim veri seti (training data set) olarak isimlendirilir. Eğitim veri setinden istatiksel yöntemler ve algoritmalar kullanılarak bir model oluşturulur. Oluşturulan bu model kullanılarak etiketsiz verilerin hangi kategoriye veya hangi değerlere sahip olacağı tahmin edilmeye çalışılır. Denetimli öğrenmede Destek Vektör Makineleri (SVM), k-Yakın Komşuluk (k-NN), Karar Ağaçları (Decision Trees), Doğrusal Regresyon (Linear Regression) ve Yapay Sinir Ağları (ANN) sıklıkla kullanılan yapılardır. Denetimsiz öğrenme, etiketsiz veriler arasındaki gizli örüntü, benzerlik veya ilişkiyi bulmaya çalışmaktadır. Denetimsiz öğrenmede eğitim veri seti etiketsiz olduğundan modelin performansını nicel olarak değerlendirmek zor olabilmektedir (Mohri vd. 2018). Denetimsiz öğrenmede k-Ortalama Kümesi (k-Means Clustering), Gizli Markov Model (Hidden Markov Model) ve Yapay Sinir Ağları sıklıkla kullanılan yapılardır. Pekiştirmeli öğrenmede ise eğitim ve test aşamaları aynı anda gerçekleştirilmektedir. Öğrenme algoritması veri toplamak için çevre ile aktif olarak iletişime geçerek, bazen de çevreyi etkileyerek, her bir eylem için anında ödül alır. Öğrenme algoritmasının amacı, çevreyle iletişimi boyunca ödülünü en üst düzeye çıkarmaktır. Bununla birlikte, çevre tarafından uzun vadeli bir ödül geribildirimi sağlanmaz. Bu yüzden öğrenme algoritması daha fazla bilgi edinmek için bilinmeyen eylemleri keşfetmek ile zaten toplanan bilgilerin daha fazla kullanılmasıyla ilgili bir seçim yapması gerekmektedir. Bu da keşif ve sömürü (exploration versus exploitation) ikilemini doğurur (Mohri vd. 2018). Pekiştirmeli öğrenmede Q Öğrenmesi (Q-Learning) ve TD Öğrenmesi (TD-Learning) sıklıkla kullanılan algoritmalardır.

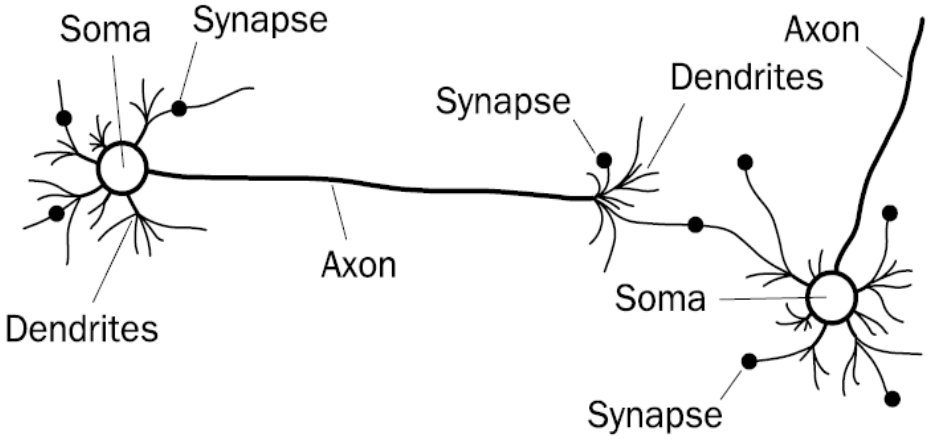
2.2 Biyolojik Sinir Ağları

Bilimin birçok alanında olduğu gibi makine öğrenmesinde de doğayı taklit ederek sorunlara çözüm aranmaktadır. Makine öğrenmesinde kullanılan yapay sinir ağları da insan öğrenmesini ve öğrenmeyi gerçekleştiren insan beynini model almaktadır. İnsan beyni oldukça sık bir şekilde birbiri ile bağlı "temel bilgi işleme birimleri" olan nöron hücrelerinden oluşmuştur. İnsan beyni yaklaşık olarak 86 milyar nöron ve bu nöronlar arasında 100 trilyon bağdan (synapses) oluşmaktadır (Herculano-Houzel 2009). Şekil 2.1'de bir nöron dokusu renklendirilmiş olarak görülmektedir.



Şekil 2.1 Renklendirilmiş nöron dokusu (ACC 2005)

Şekil 2.1'de görülen nöron dokusunda n ile gösterilen nöron, sc ile gösterilen ise yardımcı hücrelerdir (supporting cell). Nöronlar temel olarak dört bölümden oluşmaktadır. Çekirdek kısmı soma, çekirdek etrafındaki ipliksi fiber dallar dendrit (dendrite), çekirdekten çıkan tek fiber yapı akson (axon) ve diğer nöronlarla iletişim halindeki bağlar sinapsisdir (synapse). Şekil 2.2'de bir nöronun şematik yapısı gösterilmiştir.

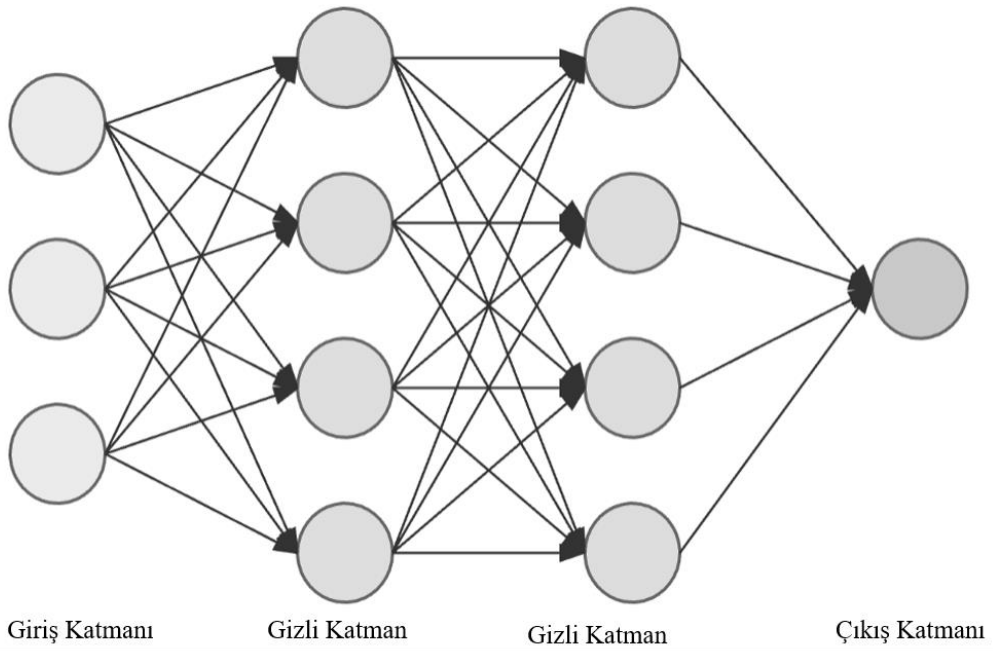


Şekil 2.2 Şematik nöron yapısı (Negnevitsky 2005)

Sinyaller bir nörondan diğerine karmaşık elektrokimyasal reaksiyonlar aracılığıyla yayılırlar. Sinapsisden yayılan kimyasal maddeler hücre gövdesinde elektriksel gerilim meydana getirmektedir. Bu elektriksel gerilim belirli bir değere ulaştığında bir elektriksel sinyal darbesi (action potential) akson boyunca iletilir. Bu darbe dağılarak sinapsislere ulaşır ve burada sinyal darbesinin değeri azalır veya artar. Bu mekanizma beyin içerisinde gerçekleşen temel öğrenme yoludur (Negnevitsky 2005).

2.3 Yapay Sinir Ağları

Denetimli ve denetimsiz öğrenmelerin her ikisinde de kullanılan yapay sinir ağları insan beyninin öğrenmesini model olarak almıştır. Yapay sinir ağları beynimizdeki biyolojik sinir ağı yapısını taklit eden oldukça basit ve çok sayıda birbiriyle bağlı, işlem yapan nöronlardan oluşmaktadır. Nöronlar birbirine ağırlıkları olan bağlarla bağlıdır ve bu bağlar ile birbirlerine sinyal gönderirler. Her bir nöron kendisine gelen bağ sayısı kadar sinyal alır fakat sadece bir çıkış sinyali üretir. Nörondan çıkan sinyal bölünmeksizin bağlantı sayısı kadar diğer nöronlara iletilir. Şekil 2.3'te yapay bir sinir ağı yapısı gösterilmektedir.



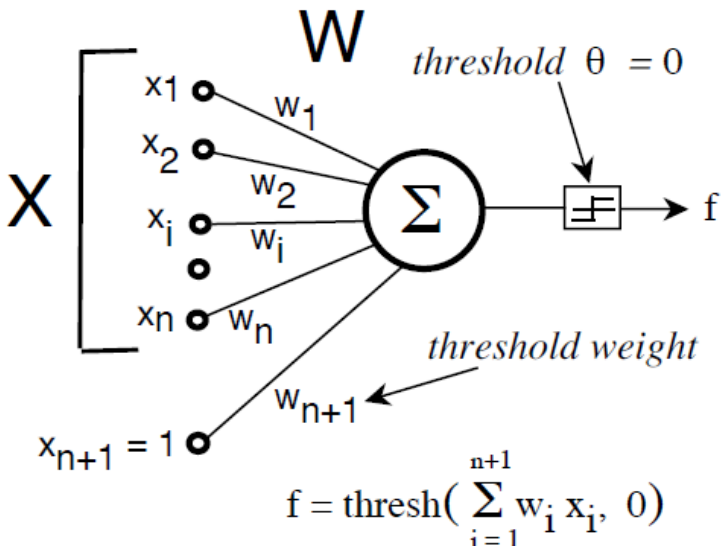
Şekil 2.3 Yapay sinir ağı

Şekil 2.3'te görüldüğü gibi nöron, gizli katman ve linkler arasındaki ağırlıklar istenildiği gibi ayarlanabilmektedir. Yapay sinir ağlarının biyolojik sinir ağlarındaki karşılıkları Çizelge 2.1'de verilmiştir.

Çizelge 2.1 Biyolojik ve yapay sinir ağları

Biyolojik Sinir Ağları	Yapay Sinir Ağları
Soma	Nöron
Dendrit	Giriş
Akson	Çıkış
Sinapsis	Ağırlık

İlk eğitilebilir yapay sinir ağı (perceptron) yapısı 1958 yılında Frank Rosenblatt tarafından oluşturulmuştur (Rosenblatt, 1958). Perceptron yapay sinir ağlarının en basit halidir. Tek bir nörondan ve ayarlanabilen ağırlıklardan ve bir adet sınırlayıcıdan oluşmaktadır. Girişlerin ağırlıklı toplamı sınırlayıcıya gelmektedir. Sınırlayıcı, eğer eşik değerinden büyük veya eşitse 1, aksi durumda 0 çıkışını vermektedir. Perceptronun amacı girişleri sınıflandırmaktır. Şekil 2.4'te perceptron yapısı görünmektedir.



Şekil 2.4 Perceptron (Nilsson ,2005)

Bir perceptronun eğitimi dört aşamadan oluşmaktadır.

A-1: İlklendirme (Initialization): Bütün ağırlıklara (w) ve threshold (θ) değerlerine [-0.5, 0.5] aralığında rastgele sayılar ata.

A-2: Aktivasyon (Activation): Bütün girişleri ağırlıklarıyla birlikte uygulayarak çıkış sinyalini (output) hesapla.

$$Y(p) = step\left[\sum_{i=1}^{n} x_i(p)w_i(p) - \theta\right]$$

A-3: Ağırlıkların Güncellenmesi (Weight Training): Bütün ağırlıklar güncelle.

$$w_i(p+1) = w_i(p) + \Delta w_i(p)\ ,\ \Delta w_i(p) = a * x_i * e_i(p)$$

A-4: Döngü (Iteration): p yi bir arttır ve A-2 ye git. Epoch sayısı kadar döngüyü sürdür.

Bu dört aşamanın uygulanması ile perceptron eğitimi yapılabilmektedir. A-1 belirtilen değerler ve A-2 belirtilen aktivasyon fonksiyonu problemin türüne ve araştırmacıya göre isteğe bağlı olarak seçilebilmektedir. Çizelge 2.2'de bir VE (AND) kapısının θ=0.2, a=0.1 parametreleri ile eğitimi verilmiştir.

Çizelge 2.2 VE kapısı için perceptron eğitimi

Epoch	Girişler		İstenen Çıkış	Giriş Ağırlıkları		Çıkış	Hata	Son Ağırlıklar	
	X_1	X_2	Y	W_1	W_2	Y_o	e	W_1	W_2
1	0	0	0	0.3	-0.1	0	0	0.3	-0.1
	0	1	0	0.3	-0.1	0	0	0.3	-0.1
	1	0	0	0.3	-0.1	1	-1	0.2	-0.1
	1	1	1	0.2	-0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	-1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	-1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	-1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

2.4 İleri Beslemeli Yapay Sinir Ağları

Paralel katmanlı yapıları ile ileri beslemeli yapay sinir ağları (feedforward neural networks) yapay sinir ağlarının popüler bir biçimidir (Ojha vd. 2017). Çoğu zaman çok katmanlı perceptron olarak da adlandırılan ileri beslemeli yapay sinir ağları nöronlar arasındaki bağların bir döngü yapmadığı ve bilginin ağ boyunca ileriye doğru tek yönlü olarak aktığı yapay sinir ağlarıdır. Şekil 2.3'te genel olarak iki gizli katmanlı bir ileri beslemeli yapay sinir ağı gösterilmiştir. İleri beslemeli yapay sinir ağlarında istenildiği kadar nöron ve gizli katman kullanılabilmektedir. Nöron ve gizli katman sayısına bağlı olarak sinir ağının eğitim süresi artmaktadır (Zhu, Miao, ve Qing 2014). İleri beslemeli yapay sinir ağının amacı bir *f** fonksiyon tahminidir. Örneğin bir sınıflandırıcı için, $y = f^*(x)$ x girişini y kategorisine eşleştirir. İleri beslemeli bir ağ, $y = f(x; \theta)$ eşlemesini tanımlar ve iyi bir fonksiyon tahmini için θ parametrelerini öğrenir (Goodfellow vd. 2016). *f* fonksiyonu aktivasyon fonksiyonu olarak adlandırılmaktadır. Aktivasyon fonksiyonu bir nöronun girişi ile çıkışı arasında matematiksel bir kapı görevi görmektedir. Yapay sinir ağlarında kullanılan belli başlı aktivasyon fonksiyonları şu şekildedir:

Sigmoid: [0,1] arasında çıkış üretmektedir. $\boldsymbol{y(x) = \frac{1}{1+e^{-x}}}$

Hiperbolik Tanjant: [-1,1] arasında çıkış üretmektedir. $\boldsymbol{y(x) = \frac{1-e^{-2x}}{1+e^{-2x}}}$

ReLu: [0,∞) arasında çıkış üretmektedir. $\boldsymbol{y(x) = \max(0, x)}$

Softmax: (0,1) arasında çıkış üretmektedir. $\boldsymbol{y(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_i}}}$

İleri beslemeli sinir ağlarının eğitiminde her bir iterasyonda giriş değerine bağlı olarak çıkış katmanından elde edilen değere göre hata/kayıp (error/lost) ortaya çıkmaktadır. Hata/kayıp hesaplaması için farklı yöntemler mevcuttur. Giriş sinyali $x_1(p)$, $x_2(p)$, ..., $x_n(p)$ için *p* iterasyonunda *y(p)* çıkış sinyali için ortaya çıkan bu hata/kayıp, *e(p)*, bir hata/kayıp fonksiyonu ile hesaplandığında;

$$\boldsymbol{w_{jk}(p+1) = w_{jk}(\mathrm{p}) + a \times x_{jk}(\mathrm{p}) \times e(p)} \quad (2.1)$$

eğitimin diğer iterasyonunda eşitlik 2.1 kullanılarak nöron ağırlıklarının güncellenmesi yapılmaktadır. Burada *a* öğrenme hızıdır. Genel olarak ağırlıklar:

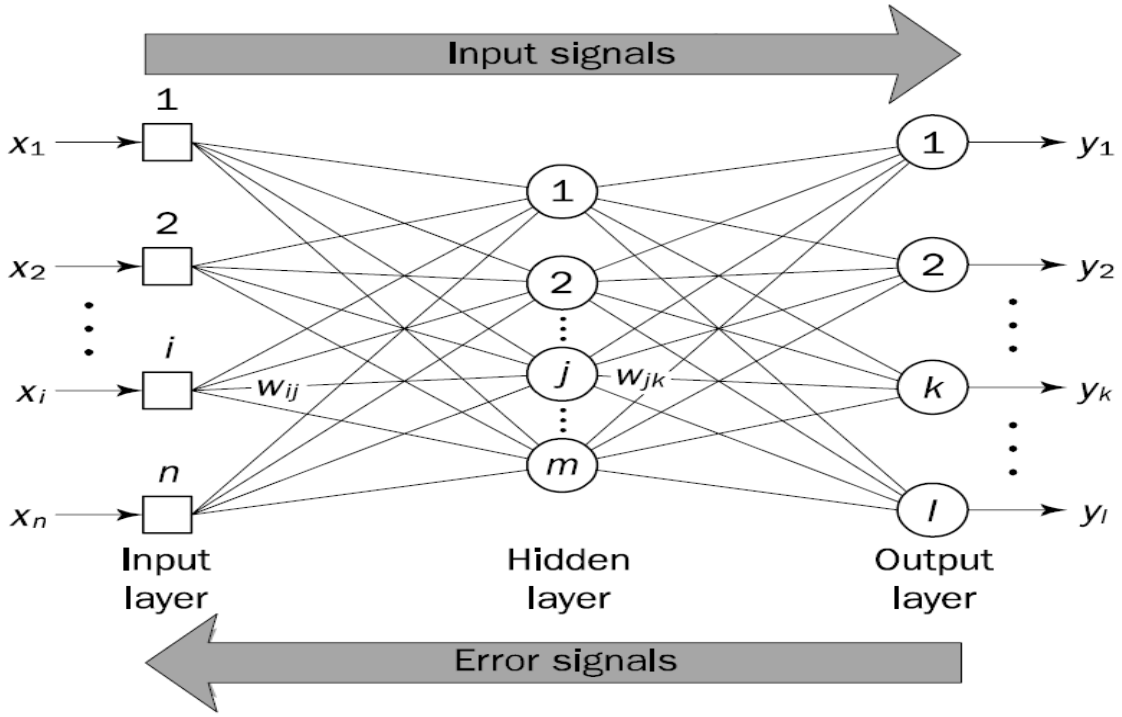
$$\Delta \boldsymbol{w_{jk}}(\mathbf{p}) = \boldsymbol{a} \times \boldsymbol{x_{jk}}(\mathbf{p}) \times \boldsymbol{e}(\boldsymbol{p}) \quad (2.2)$$

$$\boldsymbol{w_{jk}}(\boldsymbol{p}+\mathbf{1}) = \boldsymbol{w_{jk}}(\mathbf{p}) + \Delta \boldsymbol{w_{jk}}(\mathbf{p}) \quad (2.3)$$

eşitlik 2.3 ile hesaplanır. Hata/kayıp belirli bir eşik değerine ulaştığında eğitim süreci sonlandırılır.

2.5 Geri Yayılım

Yapay sinir ağlarının eğitimi için önerilen diğer bir popüler yöntem ise 1979 yılında (Bryson, Ho ve Siouris 1979) ortaya çıkan geri yayılım (back propagation) yöntemidir. Geri yayılım yöntemi iki aşamadan oluşmaktadır. İlk aşama ileriye doğru çıkış sinyalinin bulunmasıdır. Diğer aşama ise çıkış sinyali ile hata hesaplanıp, hataya bağlı olarak (error gradient) geriye doğru ağırlıkların ayarlanmasıdır. Şekil 2.5'te geri yayılım aşamaları görünmektedir. Giriş sinyali $x_1, x_2, x_3, \ldots, x_n$ ağ boyunca giriş katmanından çıkış katmanına doğru yayılırken, hata sinyali $e_1, e_2, e_3, \ldots, e_l$ çıkış katmanından giriş katmanına doğru yayılmaktadır.



Şekil 2.5 Geri Yayılım

Hata sinyalini yaymak için çıkış katmanından geriye doğru hesaplamalar yapılır. *p* iterasyonunda *k* nöronundaki hata sinyali:

$$\boldsymbol{e_k(p) = y_{d.k}(\mathbf{p}) - y_k(\mathbf{p})} \tag{2.4}$$

ile tanımlanmaktadır. Burada $\boldsymbol{y_{d.k}(\mathbf{p})}$, *p* iterasyonundaki *k* nöronunun istenen çıktı değeridir. Çıkış katmanındaki *k* nöronunun w_{jk} ağırlığının güncellenmesi ileri beslemeli ağların eğitimine benzer şekilde eşitlik 2.5'deki gibi hesaplanır.

$$\boldsymbol{w_{jk}(p+1) = w_{jk}(\mathbf{p}) - \Delta w_{jk}(\mathbf{p})} \tag{2.5}$$

Burada $\boldsymbol{\Delta w_{jk}(\mathbf{p})}$ ağırlık düzetmesidir. İleri beslemeli yapay sinir ağından farkı, giriş sinyali yerini burada hata gradyanı (error gradient) kullanılmasıdır.

$$\boldsymbol{\Delta w_{jk}(p) = a \times y_j(\mathbf{p}) \times \delta_k(\mathbf{p})} \tag{2.6}$$

$\boldsymbol{\delta_k(\mathbf{p})}$ ile gösterilen çıkış katmanındaki k nöronunun *p* iterasyon sununda oluşan hata gradyanıdır. Hata gradyanını hesaplanmasında aktivasyon fonksiyonunun türevi alınarak eşitlik 2.4 verilen hata sinyali ile çarpılır.

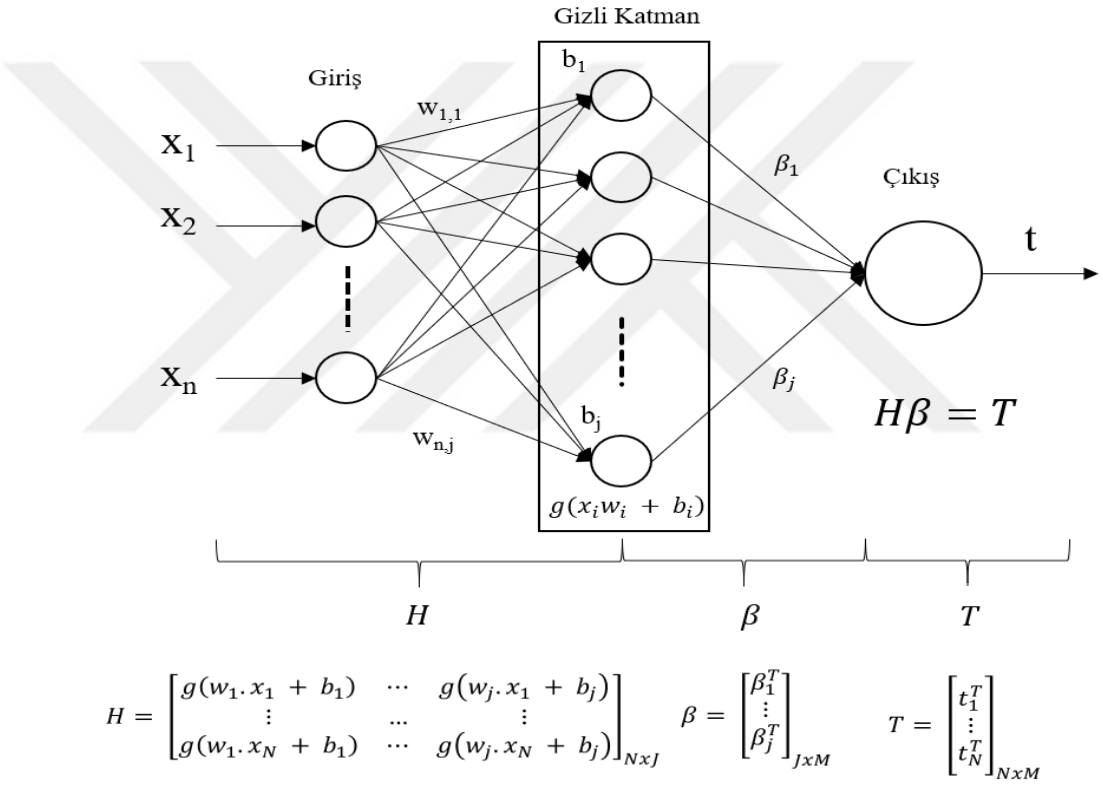
$$\boldsymbol{\delta_k(p) = \frac{\partial y_k(\mathbf{p})}{\partial x_k(\mathbf{p})} \times e_k(p)} \tag{2.7}$$

Aktivasyon fonksiyonlarının türevinin alınabilir olması burada oldukça önemlidir. Bu işlemler çıkış katmanında giriş katmanına doğru ilerleyerek ağırlıklar güncellenmektedir. Her bir iterasyonda girişten çıkışa kadar nöronlar aktivasyon fonksiyonuna tabi tutulup hata gradyanı hesaplanıp tekrardan geriye doğru güncelleme işleminin yapılması oldukça fazla işlem gücü ve eğitim zamanı harcamaktadır.

2.6 Aşırı Öğrenme Makinesi (ELM)

İleri beslemeli yapay sinir ağları ve geri yayılımda grandyant tabanlı fonksiyonların kullanılması ve ağırlıkların her iterasyonda ayarlanması model eğitim süresini artırmaktadır. Özellikle çok katmanlı ve çok nöronlu yapay sinir ağlarında model eğitim süresinin yanı sıra kaynak tüketimi de artmaktadır. Öğrenme hızının artırılması noktasında ELM kullanımı daha hızlı eğitilen yapay sinir ağları oluşturulmasına yardımcı olmaktadır. ELM, perceptron gibi tek katmanlı ileri beslemeli yapay sinir

ağlarının özel bir türüdür. 2004 yılında Huang Guangbin tarafından geliştirilmiştir. Geleneksel gradyant tabanlı yapay sinir ağı öğrenmesi yerine, nöronların giriş ağırlıkları ve bias değerlerini rastsal olarak atama yaparak çıkış ağırlıklarını analitik olarak ayarlamaktadır. Böylece gradyant tabanlı öğrenmelere kıyasla çok hızlı yapay sinir ağı öğrenmesi gerçekleştirilebilmektedir (Huang vd. 2004). ELM gizli katmandaki nöronların ağırlıklarını iteratif olarak ayarlamak yerine nöronlardaki parametreleri rastsal bir şekilde ayarlar. Tek iterasyonda çıkış nöron parametreleri hesaplanmaktadır. Şekil 2.6 da ELM yapısı gösterilmiştir.



Şekil 2.6 ELM

Aktivasyon fonksiyonu *g(x)*, N sayıda giriş değerleri $x_i = (x_1, x_2, \ldots x_n)^T \in R^n$, çıkışları $t_i = (t_1, t_2, \ldots t_n)^T \in R^m$ ve K sayıda gizli katman nöronu olan bir tek katmanlı ileri beslemeli yapay sinir ağı matematiksel olarak aşağıdaki gibi ifade edilir.

$$\sum_{i=1}^{K} \beta_i g(w_i.x_j + b_i) = o_j, j = 1,\ldots,\text{N} \quad (2.8)$$

Burada $w_i = (w_1, w_2, \ldots, w_n)^T$ giriş nöronları ile gizli katman nöronları arasındaki ağırlık vektörünü, $\beta_i = (\beta_1, \beta_2, \ldots \beta_n)^T$ ise gizli katman nöronları ile çıkış nöronları arasındaki ağırlık vektörüdür. b_i ise bias değeridir. Eşitlik 2.8 de K=N ise yani gizli katman nöron sayısı ile giriş sayısı birbirine eşit ise 0 hata ile çıkışları hesaplamak mümkündür Yani $\sum_{i=1}^{N} \|o_j - t_j\| = 0$ 'dır. Bu durumda eşitlik şu şekilde yazılabilir:

$$\sum_{i=1}^{N} \beta_i g(w_i . x_j + b_i) = t_j, j = 1, \ldots, \mathrm{N} \tag{2.9}$$

2.9'u daha sade bir şekilde yazılacak olursa

$$H\beta = T \tag{2.10}$$

Burada;

$$H(w_1, w_2, \ldots, w_n) = \begin{bmatrix} g(w_1 . x_1 + b_1) & \cdots & g(w_K . x_1 + b_K) \\ \vdots & \ldots & \vdots \\ g(w_1 . x_N + b_1) & \cdots & g(w_K . x_N + b_K) \end{bmatrix}_{NxK}$$

gizli katmanların çıkış matrisi β çıkış katmanın ağırlık matrisi ve T çıkışı temsil etmektedir.

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_K^T \end{bmatrix}_{Kxm} \text{ ve } T = \begin{bmatrix} t_1^T \\ \vdots \\ T_K^T \end{bmatrix}_{Nxm}$$

Eğer $\{(x_i, t_i) \mid x_i \in R^n, t_i \in R^m\}$ n≠m ise yani H matrisi kare matris değil veya tersi alınamıyorsa, yani tekil bir matris ise (singular) o zaman Moore-Penrose yöntemi kullanılarak eşitlik 2.10 düzenlenirse:

$$\beta = H^{\dagger} T \tag{2.11}$$

Burada $H^{\dagger} = H^T (HH^T)^{-1}$ eğer m<n, $H^{\dagger} = (H^T H)^{-1} H^T$ eğer n<m olarak denklem genelleştirilir.

ELM Özellikleri:

- Tek gizli katmanlı ileri beslemeli yapay sinir ağıdır.
- Giriş ağırlıkları ve gizli katman bias rastgele seçilir.

- Ağırlıklar ve bias tekrardan ayarlanmasına (tune) gerek yoktur.
- $H\beta=T$ Lineer bir sistem için, ağı en küçük kareler (β) çözümünü bularak eğitir.

ELM Algoritması:

$\aleph = \{(x_i, t_i) \mid x_i \in R^n, t_i \in R^m, i = 1,2, \dots, N\}$ şeklinde bir eğitim seti verilsin, g(x) aktivasyon fonksiyonu, K nöron sayısı olsun.

A-1: İlklendirme (Initialization)

Bütün ağırlıklara $\boldsymbol{w_i}$ ve bias $\boldsymbol{b_i}$ değerlerini i=1, 2,…,N için rastgele ata.

A-2: Çıkış Matrisini Hesapla

Atanan ağırlık ve bias değerleri ile gizli katman çıkış matrisi H'yi hesapla.

A-3: Çıkış Ağırlıklarını Hesapla

Çıkış ağırlıkları $\boldsymbol{\beta}$'yi hesapla.

$\beta = H^{\dagger}\, T\ ,\quad T = [t_1, t_2, \dots, t_N\ ]^T$

2.7 Kısıtlı Aşırı Öğrenme Makinesi (CELM)

Gizli katman parametrelerinin güncellenmesine gereksinim duyulmaması, hızlı eğitim süresi ve iyi genelleme (generalization) performansının olması ELM popülerliğini artırmıştır. Bunun yanı sıra ELM'de rastgele seçilen gizli katman parametreleri nöronlarının etkili kullanımını engellemektedir (Zhu vd. 2005). Bu sorunun üstesinden gelerek istenilen performans değerine ulaşabilmek için gizli katmandaki nöron sayısı arttırılır. Nöron sayısının artırılması ise eğitim süresini etkilediği gibi gerçek dünya uygulamaları için etkili bir yöntem değildir (Zhu vd. 2014). Rastgele atanan gizli katman parametrelerinin yarattığı sorunu ortadan kaldırmak için genel olarak üç yöntem vardır.

Bu yöntemlerden ilki artırımsal öğrenme metotları (online incremental learning) kullanarak gizli katmandaki nöron sayısının dinamik olarak artırımıdır. Diğer bir

yöntem gizli katmandaki nöron sayısının belirli istatiksel yöntemler kullanılarak nöron sayısının azaltılması yöntemidir. Son yöntem ise gradyan tabanlı yöntemler kullanılarak gizli katman parametrelerinin güncellenmesidir.

Gizli katman parametrelerinin ve nöron sayılarının hesaplanması için birçok yöntem öne sürülmüştür. Bunlardan birisi de Zhu ve arkadaşları (2014) tarafından geliştirilen Kısıtlı Aşırı Öğrenme Makinesi (Constrained Extreme Learning Machine-CELM) algoritmasıdır. CELM'de gizli katman parametrelerini rastgele atamak yerine sınıflar arasındaki uzaklıklardan yararlanarak gizli katman ağırlık ve bias değerini hesaplamaktadır. Bu algoritmaya göre farklı sınıflara ait rastgele vektörler alınarak sınıflar arasındaki uzaklık hesaplanır ve hesaplanan uzaklıklara bağlı olarak gizli katman nöronlarının ağırlık ve bias parametreleri atanır.

CELM'de sınıflar arasındaki farklılığı hesaplamak için sınıflardan birinin (c1) 1, diğerinin (c2) ise -1 ekseninde olduğu varsayılır. Bu varsayım ile 2.8'de verilen aktivasyon fonksiyon hesabı:

$$ax_{c1}(x_{c2} - x_{c1}) + b = -1$$
$$ax_{c2}(x_{c2} - x_{c1}) + b = 1$$

şeklinde yazılabilir. Bu denklemden normalizasyon faktörü;

$$a = \frac{2}{\|x_{c2} - x_{c1}\|^2}$$

şeklinde hesaplanabilir. *b* bias değeri ise;

$$b = \frac{(x_{c1}+x_{c2})(x_{c1}-x_{c2})}{\|x_{c2}-x_{c1}\|^2} \quad (2.12)$$

Burada $(x_{c2} - x_{c1})$ fark vektörüdür. Fark vektörünün normalize edilmesi sonucunda da ağırlık vektörü aşağıdaki 2.13 teki gibi hesaplanır.

$$w = \frac{2(x_{c2}-x_{c2})}{\|x_{c2}-x_{c1}\|^2} \quad (2.13)$$

Bu şekilde, sınıflar arasındaki farklar baz alınarak ağırlık ve bias değerlerinin rastgele atanması yerine 2.12 ve 2.13 kullanılarak hesaplanmaktadır.

2.8 Çevrim İçi Öğrenme (Online Learning)

Makine öğrenmesinde popülerliğini koruyan diğer bir konu ise çevrimiçi öğrenmedir (online learning). Çevrimiçi öğrenme, tüm veri seti üzerinde model eğitiminin mümkün olmadığı durumlarda kullanılan bir makine öğrenmesi yöntemidir. Çevrimiçi öğrenme, oluşturulacak modelin dinamik olarak verideki yeni örüntülere uyum sağlayacak veya verinin kendisinin zamanın bir fonksiyonu olarak üretildiği, örneğin hisse senedi fiyat tahmini gibi, durumlarda kullanılmaktadır (Pentreath 2015). Diğer öğrenme yöntemleri eski ve yeni veriyi birlikte kullanarak yeniden eğitim aşamasını tekrarlamaktadır. Bu ise oldukça zaman harcayan bir uygulamadır. Bu sebepten dolayı birçok endüstriyel uygulamam çevrimiçi öğrenmeyi kullanmaktadır çünkü yeni bir veri geldiğinde tekrardan modelin eğitime gerek yoktur (Liang vd. 2006).

Çevrimiçi öğrenmede eldeki verilerden model eğitilip, yeni veriler geldikçe model güncellenebileceği gibi elde mevcut veri olmaksızın ilk verinin elde edilmesiyle model oluşturulup devam eden verilerle güncelleme yapılarak da oluşturtulabilir. Veriler sıralı ve belirli boyutta gelebileceği gibi (sequential one-by-one) sırasız ve belirli boyutu olmayan (chunk-by-chunk) şekilde de gelebilir (Liang vd. 2006). Çevrimiçi öğrenmenin arkasındaki fikir modelin yeni veri geldiğinde yeniden eğitilmesi yerine kendisini sürekli güncellemesidir. Bunun yanında online öğrenmenin sadece çevrimiçi durumlarda kullanılması zorunluluğu yoktur. Sofistik gradyan indirme (stochastic gradient descent) optimizasyon yönteminde olduğu gibi model her eğitim aşaması tamamlandıktan sonra güncellenmektedir (Pentreath 2015). Çevrimiçi öğrenme aynı zamanda diğer öğrenme metotlarıyla birlikte de kullanılabilir. Diğer öğrenme yöntemleriyle bir model oluşturulur ve gerçek dünya uygulamasında kullanıma alındıktan sonra online öğrenme yöntemi ile isteğe bağlı olarak yeni veriler kullanılarak hızlı bir şekilde güncelleme yapılabilir. Çevrimiçi öğrenme ayrıca verilerimizin ve modelimizin belleğimize sığmayacak kadar büyük olduğu durumlarda verilerin parçalanarak modelin eğitilmesine olanak sağlamaktadır.

2.9 Çevrim İçi Sıralı Aşırı Öğrenme Makinesi (OS-ELM)

Çevrimiçi öğrenmede eğitim hızı önem teşkil etmektedir. Bazı durumlarda model güncellemesinin veri akış hızına yetişmesi gerekmektedir. Bu gibi durumlarda hızlı model eğitimi sağlayan ELM tabanlı çevrimiçi öğrenme modelleri kullanılabilir. Liang ve arkadaşları 2006 yılında çevrim içi öğrenmede ELM tabanlı OS-ELM algoritmasını geliştirmişlerdir. ELM algoritmasında eğitim verisinin eğitim zamanında hazır olduğu varsayılır ancak gerçek uygulamalarda bu her zaman mümkün olmayabilir. Bu yüzden ELM algoritmasında değişiklik yapılarak çevrimiçi öğrenme ve sıralı öğrenme algoritmasına dönüştürülebilir. OS-ELM iki kısımdan oluşmaktadır. İlk eğitimin yapıldığı kısım ve sıralı eğitimin yapıldı kısım.

2.11 de verilen β, ELM'de gizli katman ağırlık matrisi, problemin en küçük kareler yöntemiyle bulunan çözümüdür. $H^{\dagger} = (H^T H)^{-1} H^T$ çözümünde $(H^T H)$ genellikle tekil olma eğilimindedir. Tekil olan matrisin tersi alınamayacağından çözüm bulunamayabilir. Bu durumun önüne geçmek için gizli katman nöron sayısı azaltılabilir veya daha büyük başlangıç eğitim verisi seçilebilir. Bu durum OS-ELM ilk eğitim aşamasında önem arz etmektedir. β matrisini $\beta = (H^T H)^{-1} H^T T$ şekilde açık olarak yazabiliriz.

$\aleph_0 = \{(x_i, t_i)\}_{i=1}^{N_0}$ ve $N_0 \geq S$ şeklinde başlangıç eğitim seti verildiğinde (S gizli katman nöron sayısı) problemin çözümü $\|H_0\beta - T_0\|$ minimum eden değerdir (Liang vd. 2006). Bu eşitliğin çözümü ise $\beta^{0} = K_0^{-1} H_0^T T_0$'dır ve burada $K_0 = H_0^T H_0$ şeklinde tanımlanır. $\aleph_1 = \{(x_i, t_i)\}_{i=N_0+1}^{N_0+1}$ şeklinde diğer bir veri seti geldiğinde problemin çözümü;

$$\left\| \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \beta - \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \right\| \tag{2.14}$$

$\aleph_0$ve $\aleph_1$şeklinde verilen iki sıralı verinin eğitim ağırlıkları 2.14 ün çözümü $\beta^{(1)}$'dir.

$$\beta^{1} = K_1^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \text{ ve } K_1 = \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \tag{2.15}$$

Sıralı öğrenmeyi sağlamak için $\beta^{(1)}$'i $\beta^{(0)}$'ın fonksiyonu olarak yazmamız gerekmektedir. K_1'i bu şekilde düzenleyecek olursak:

$$\begin{aligned} K_1 &= [H_0^T \quad H_1^T]\begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \\ &= K_0 + H_1^T H_1 \\ \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} &= H_0^T T_0 + H_1^T T_1 \\ &= K_0 K_0^{-1} H_0^T T_0 + H_1^T H_1 \\ &= K_0 \beta^{(0)} + H_1^T H_1 \\ &= (K_1 - H_1^T H_1)\beta^{(0)} + H_1^T T_1 \\ &= K_1 \beta^{(0)} - H_1^T H_1 \beta^{(0)} + H_1^T T_1 \end{aligned} \tag{2.16}$$

2.15 ve 2.16 birleştirilirse 2.17 elde edilir.

$$\begin{aligned} \beta^1 &= K_1^{-1}\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \\ &= K_1^{-1}(K_1 \beta^{(0)} - H_1^T H_1 \beta^{(0)} + H_1^T T_1) \\ &= \beta^{(0)} + K_1^{-1} H_1^T \left(T_1 - H_1 \beta^{(0)}\right) \end{aligned} \tag{2.17}$$

Burada $K_1 = K_0 + H_1^T H_1$'dır. $\aleph_0$ve $\aleph_1$için bulunan 2.17 *k+1* verisi için genelleştirilirse;

$$\begin{aligned} K_{k+1} &= K_k + H_{k+1}^T H_{k+1} \\ \beta^{(k+1)} &= \beta^{(k)} + K_{k+1}^{-1} H_{k+1}^T \left(T_{k+1} - H_{k+1}\beta^{(k)}\right) \end{aligned}$$

K_{k+1}^{-1}'in Woodbury algoritmasına (Golub ve Loan 1996) göre düzenlenirse:

$$\begin{aligned} K_{k+1}^{-1} &= (K_k + H_{k+1}^T H_{k+1})^{-1} \\ &= K_k^{-1} - K_k^{-1} H_{k+1}^T (I + H_{k+1} K_k^{-1} H_{k+1}^T)^{-1} H_{k+1} K_k^{-1} \end{aligned}$$

$K_{k+1}^{-1} = P_{k+1}$ olarak ifade edersek $\beta^{(k+1)}$ aşağıdaki şekilde yazılır.

$$P_{k+1} = P_k - P_k H_{k+1}^T (I + H_{k+1} P_k H_{k+1}^T)^{-1} H_{k+1} P_k \tag{2.18}$$

$$\beta^{(k+1)} = \beta^{(k)} + P_{k+1} H_{k+1}^T \left(T_{k+1} - H_{k+1}\beta^{(k)}\right) \tag{2.19}$$

2.18 ve 2.19 sıralı öğrenmenin formülüdür. Eğer veriler sıralı bir şekilde değil de hepsi aynı anda girişe verilirse OS-ELM başlangıç eğitimi kısmını tamamlayacağından ELM ile aynı olacaktır. OS-ELM eğitim algoritması aşağıda verilmiştir.

OS-ELM:

A-1: a,β,b değerlerini rastgele olarak ata.

A-2: Sırasıyla aşağıdaki değerleri hesapla

$$H_0 = G(x_0.a + b) \ \ (k_0 \geq N)$$
$$P_0 = (H_0^T H_0)^{-1}$$
$$\beta_0 = P_0 H_0^T t_0$$

A-3: Yeni veri geldiği sürece aşağıdaki değerleri hesapla.

$$P_{k+1} = P_k - P_k H_{k+1}^T (I + H_{k+1} P_k H_{k+1}^T)^{-1} H_{k+1} P_k$$
$$\beta^{(k+1)} = \beta^{(k)} + P_{k+1} H_{k+1}^T \left(T_{k+1} - H_{k+1}\beta^{(k)}\right)$$

Bu bölümde makine öğrenmesi alanın denetimli ve denetimsiz öğrenme kollarında sıklıkla kullanılan yapay sinir ağlarından bahsedilmiştir. Yapay sinir ağlarının etkili bir algoritması olan ELM ve ELM algoritmasının iyileştirilmiş hali olan CELM algoritmaları önerilen modelin yapı taşlarını oluşturmaktadır. OS-ELM algoritmasında da görüldüğü gibi başlangıç gizli katman giriş ağırlıkları ve bias değerleri rastgele atanmaktadır. Rastgele atanan gizli katman parametreleri ELM de olduğu gibi gizli katmanın daha etkili kullanımını engellemektedir. Çevrim içi öğrenme problemindeki bu sorunun çözümüne yönelik bu çalışmada bir model önerilmiştir. Önerilen model 3. bölümde detaylıca anlatılmaktadır.

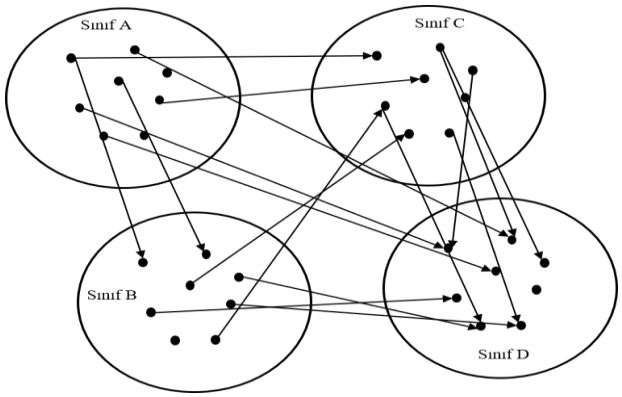
3. YÖNTEM VE MATERYAL

3.1 Önerilen Model

Yapay sinir ağlarının hızlı eğitimini sağlayan ELM, gizli katman nöron giriş parametrelerini rastgele ataması ile gizli katman nöronlarını etkin olarak kullanamamaktadır. ELM ile oluşturulacak modelin hata oranının azaltmak, başarım oranını yükseltmek için gizli katman nöron sayısının artırılması çözümlerden birisidir. Diğer bir çözüm yöntemi ise rastgele seçilen gizli katman parametrelerinin güncellenmesidir.

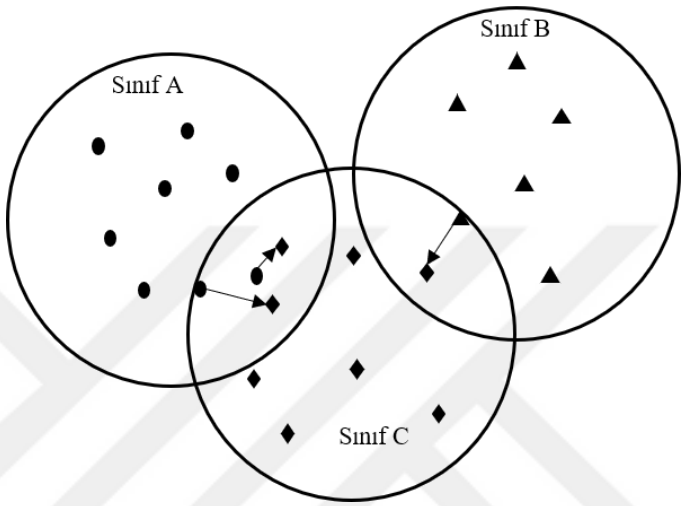
Sınıflar arasındaki farka bağlı olarak gizli katman nöron giriş parametrelerinin atanması gizli katman nöronlarının çıkış parametrelerinin sınıflar arasındaki farkı yansıtmasına yardımcı olacaktır. Böylece sınıflandırma problemlerinde daha başarılı modeller oluşturulabilecektir. Önerilen yöntemde çevrimiçi ve sıralı öğrenme yapabilen OS-ELM modelinin daha iyi bir sınıflandırma performansı sergilemesini sağlamak amacı ile başlangıç parametrelerinin atanmasında sınıflar arasındaki ayırıcı özellikler kullanılmış ve OS-CELM adında yeni bir model önerilmiştir.

Sınıflar arasındaki farklılıkları ortaya koymak için temel olarak sınıf üyeleri arasındaki uzaklıktan yararlanılmıştır. Bunu sağlayabilmek için sınıf üyeleri arasında rastgele vektörler çizilmiştir. Şekil 3.1 de önerilen modelde kullanılan sınıflar arasındaki rastgele çizilen vektörler gösterilmiştir.



Şekil 3.1 Sınıflar arası vektörler

Vektörler çizilmeden önce her sınıfa karşılık gelen artan bir şekilde numara verilmiştir. Çizilen vektörler rastgele seçildiğinden vektörler küçük numaralı sınıflardan yüksek numaralı sınıflara doğru çizilmiştir. Çizilen vektörlerden hangi sınıftan hangi sınıfa ne kadar vektör çizileceği yine rastgele olarak seçilmiştir. Gerçek verilerde sınıflar arasındaki farklar Şekil 3.1'deki gibi belirgin olmayacaktır. Şekil 3.2'de örnek bir durum gösterilmiştir.



Şekil 3.2 Birbirine yakın veriler

Şekil 3.2'deki veriler gürültü verileri de olabilmektedir. Bu veriler sınıflar arasındaki fakı ortaya çıkarmayı zorlaştırmaktadır. Veri setlerindeki gürültülerden arındırmak için çeşitli metotlar ve filtreler bulunmaktadır (Garcia vd. 2015). Önerilen yöntemde gürültülerden ve sınıflar arası farklılığı ortaya çıkaramayacak verilerin modeli en az etkilemesi için sınıflar arası çizilen vektörlerin uzunluklarından yararlanılmıştır. Çizilen her bir vektör arasındaki uzaklık hesaplanarak belirlenen bir eşik değerinin altında olanlar silinerek sınıflar arasında yeni rastgele vektörlerin çizimi sağlanmıştır. Sınıflar arasındaki uzaklığın yeterince ayırıcı olup olmadığını bulmak (eşik değer) için aşağıdaki formülden yararlanılmıştır (Zhu vd. 2014).

$$\|x_{c1} - x_{c2}\|^2 < \frac{1}{fT} \tag{3.1}$$

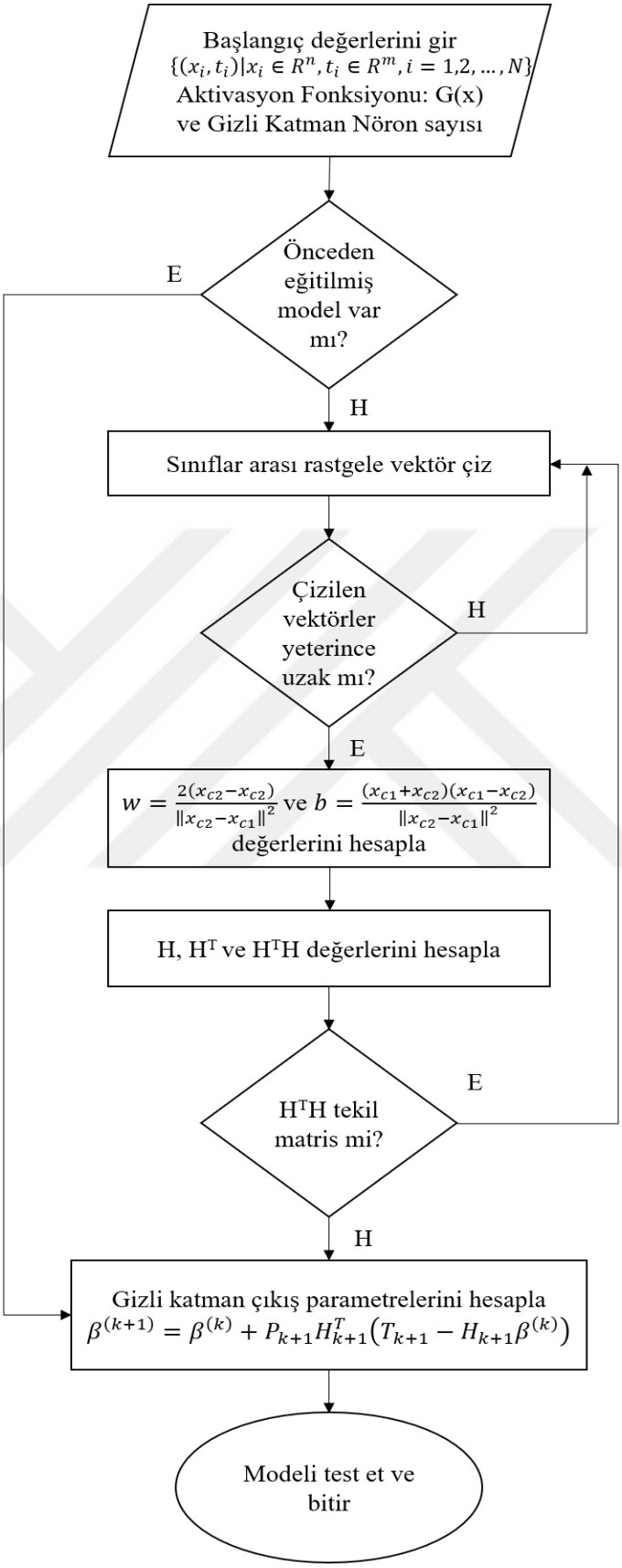
Formül 3.1'de f başlangıç değeri 1 olup birbirine yakın noktaların sıklığına göre ağırlık hesabı sırasında artmaktadır. T çıkış nöron sayısıdır yani çıkış sınıf sayısıdır. Eğitim aşamasının başında belirtilir. Farklı sınıflar arasındaki veriler arasındaki vektörün boyu

belirlenen eşikten küçükse vektör silinir ve yeni vektör oluşturulur. Sınıflar arasındaki vektörler çizildikten sonra 2.13 formülü kullanılarak ağırlık matrisi hesaplanır. Bias matrisi için 2.12 formülü kullanılarak gizli katmanın giriş parametreleri hesaplanmış olur. Veriler birer birer, sıralı veya tek seferde gelebilirler. Elimizde önceden oluşturulmuş bir model varsa sıralı eğitim safhasına geçiş yapılır. Eğer önceden oluşturulmuş bir model yoksa ilk eğitim safhasına uygulandıktan sonra sıralı eğitim safhasına geçilir. İlk eğitim safhasında ($H^T H$) işleminin tekil matris olup olmadığı kontrolü yapılır. Eğer tekil matris değilse eğitim işlemine devam edilir. Eğer tekil bir matrisse gizli katmanın giriş parametreleri tekrardan yapılmalıdır.

3.2 Modelin Uygulanması

Önerilen modelin uygulanması aşamasında algoritma ve model tasarımı için Python programlama dili kullanılmıştır. Şekil 3.3'de önerilen modelin akış diyagramı verilmiştir. Matris işlemlerinin yoğunlukta olduğu yapay sinir ağları eğitiminde matris üzerinde işlem yapmayı kolaylaştıran numpy kütüphanesi kullanılmıştır. Normalizasyon işlemleri, test ve eğitim verilerinin rastgele olarak seçilip ayrışması için python sklearn kütüphanesinden yararlanılmıştır. Normalizasyon için sklearn kütüphanesinin *minmax_scale* methodu, eğitim ve test verilerinin ayrılması için ise *train_test_split* metotlarından yararlanılmıştır. Kullanılan veri setleri %80 eğitim %20 test olarak ayrılmıştır.

Aktivasyon fonksiyonu için [-1,1] aralığında çıkış üreten hiperbolik tanjant fonksiyonu seçilmiştir. Çıkış katmanında sınıflandırma işlemini olasılık hesabını kullanarak saptayan softmax fonksiyonu kullanılmıştır. Modelin oluşturulması ve performanslarının test edilmesi için Intel Core i7 3.30GHz işlemcili 8 GB RAM özellikleri olan bilgisayar kullanılmıştır.



Şekil 3.3 OS-CELM Akış diyagramı

3.3 Veri Setleri

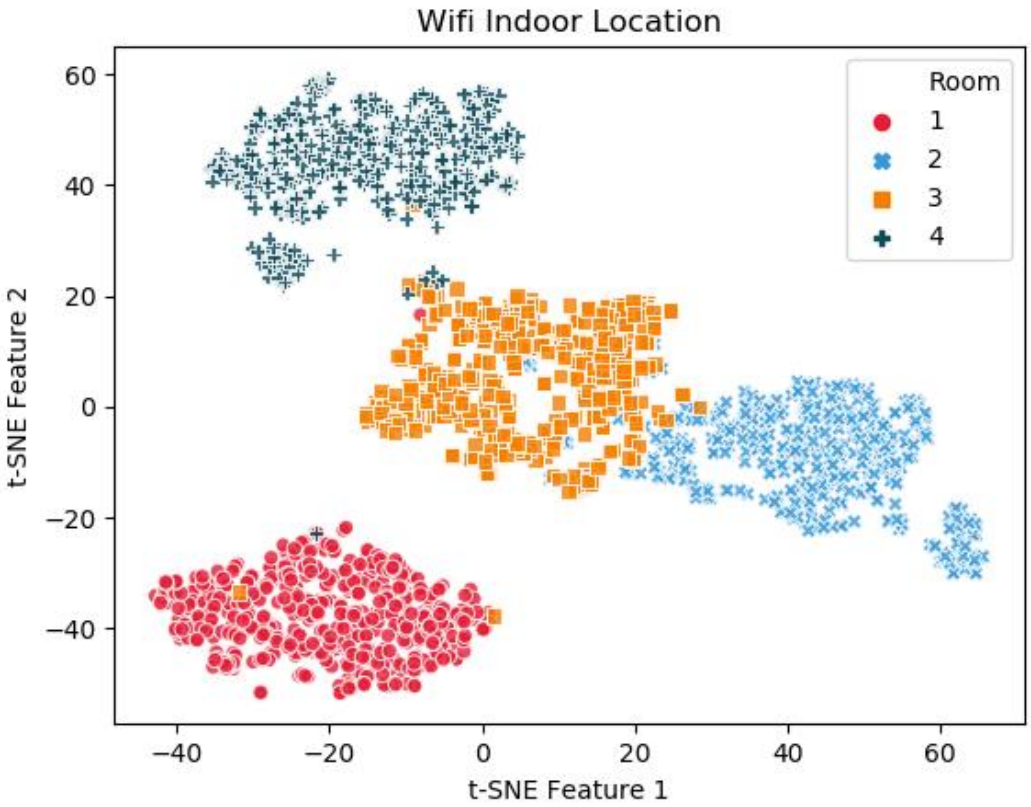
Önerilen modelin performansını değerlendirmek üzere UCI veri tabanından (Blake and Merz 1998) sınıflandırma kategorisine ait iki farklı veri seti seçilmiştir. Bunlar "Wireless Indoor Localization", "Skin Segmentation" veri setleridir. Diğer kullanılan veri seti ise popüler MNIST veri setidir (Lecun vd. 1998).

Wireless Indoor Localization veri seti (Rohra vd. 2017): 4 farklı odadan telefon ile 7 farklı sinyal seviyesi ölçülerek hangi odada bulunulduğunun tahmin edilmesini sağlayan veri setidir. Toplamda 2000 veri olup 8 farklı özelliğe sahiptir. Bunlar 4 ayrı sınıf bilgisine aittir. Her bir sınıf için 500 adet veri bulunmaktadır. Bu yüzden dengeli bir veri setidir. Veri setinin 7 özelliği ratio verilerden oluşmakta ve sinyal kalitesinin dB değerini ölçmektedir. Diğer özellik ise nominal veriler olup sınıf (bulunulan oda) değerinden oluşmaktadır. Kayıp veri bulunmamaktadır. Çizelge 3.1'de veri setine ait bilgiler verilmektedir.

Çizelge 3.1 Wifi indoor localization

Sınıflar	Ortalama	Min	Max	Veri Miktarı
Oda-1	-68.668	-97	-47	500
Oda-2	-57.251	-92	-10	500
Oda-3	-62.156	-93	-40	500
Oda-4	-64.403	-98	-36	500

Çok boyutlu veri setini görüntülemenin zorluk ve dezavantajlarından dolayı veri seti görüntülemek amacı ile 3 boyuta indirgenmiştir. Boyut indirgenmesi yöntemi olarak t-Distributed Stochastic Neighbor Embedding (t-SNE) metodundan yararlanılmıştır (Maaten ve Hinton 2008). Şekil 3.4'te Wifi Indoor Localization veri setinin boyut indirgenmiş hali görünmektedir. Her bir sınıf ayrı renklerde ve işaretlerle gösterilmiştir.



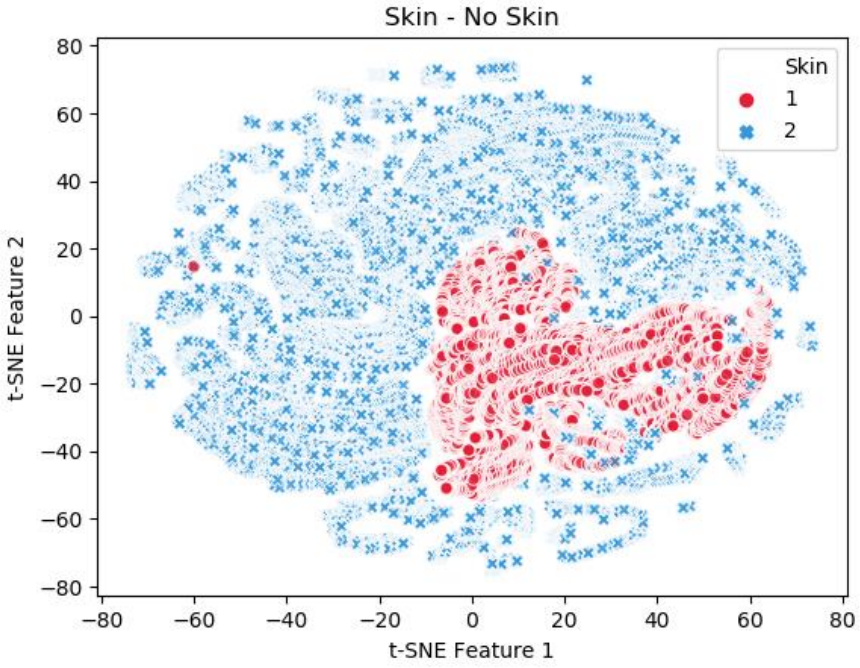
Şekil 3.4 Wifi indoor localization veri seti

Skin Segmentation veri seti (Rajen vd. 2009): R, G, B renk uzayından verilen bilginin ten (skin) olup olmadığının tahmin edilmesini sağlayan veri setidir. FERET ve PAL veri tabanlarından farklı renk, cinsiyet ve yaşlardaki kişilerin yüz resimlerinden toplanan 245057 adet veriden oluşmaktadır. Bu verilerden 50859 ten/cilt, 194198 adeti ten cilt olmayan verilerden oluşmaktadır. 3 özellik ratio verilerden oluşmakta ve dokunun R, G, B değerlerini 0-255 arasında vermektedir. Veri setinin diğer özelliği nominal veri olup sınıf (ten (skin), ten değil (no skin)) değerinden oluşmaktadır. Kayıp veri bulunmamaktadır. Çizelge 3.2'de veri setine ait bilgiler verilmektedir.

Çizelge 3.2 Skin/No Skin

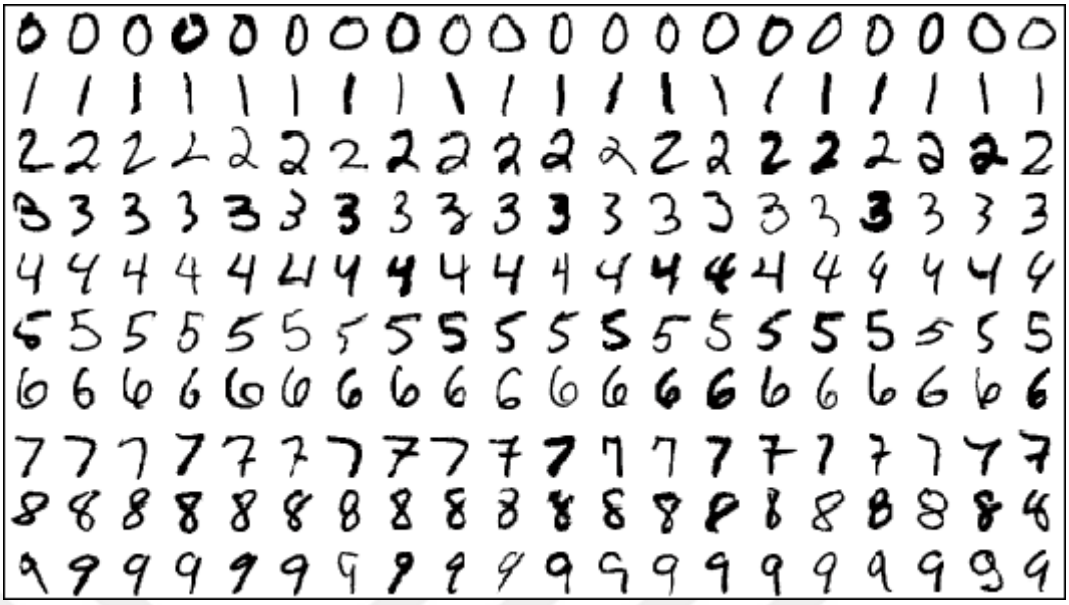
Sınıflar	Ortalama	Min	Max	Veri Miktarı
Ten/Cilt	154,821	26	255	50859
Ten/Cilt Değil	119,609	0	255	194198

t-SNE metodu kullanılarak Skin/No Skin veri setinin 3-boyutlu görüntüsü şekil 3.5'te verilmiştir. Her bir sınıf ayrı renklerde ve işaretlerle gösterilmiştir. Kırmızı ile gösterilen ten (skin), mavi ile gösterilen ten değil (no skin) verisidir.



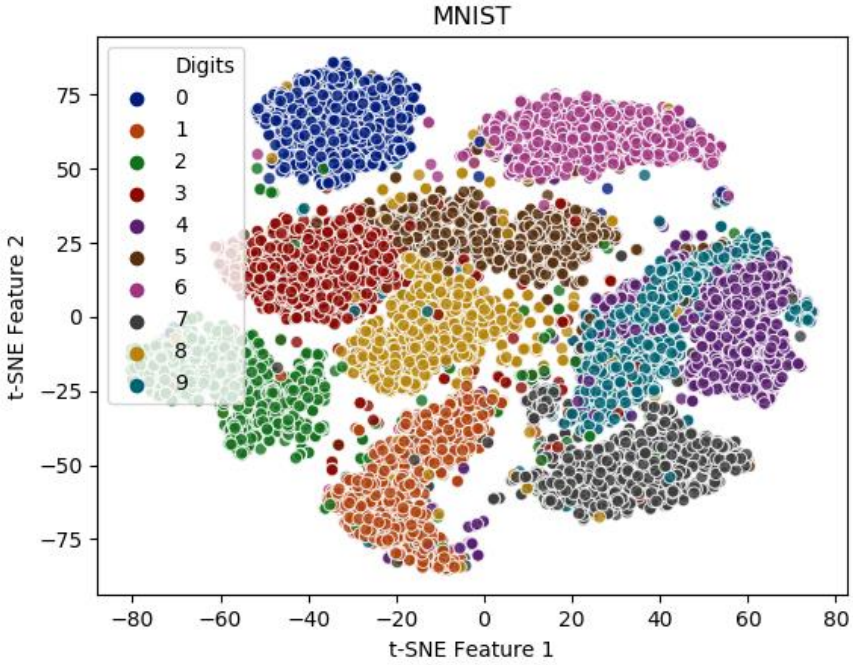
Şekil 3.5 Skin- no skin veri seti

MNIST veri seti (Lecun vd. 1998): El yazısı ile yazılmış 0-9 arasındaki rakamların toplandığı veri setidir. NIST veri setinden oluşturulmuş bir alt veri setidir. 60000 eğitim verisi 10000 test verisi olmak üzere toplamda 70000 veriden oluşmaktadır. Her bir rakam 28x28 boyutunda piksel yoğunluklarına göre merkeze alınmış gri resimlerden oluşmaktadır. Her bir el yazısı rakamı içeren resimler 0 – 255 aralığında değerlere sahiptir. 0 beyaz arka planı, 255 siyah yazının olduğu kısımları ifade etmektedir. Şekil 3.6'da veri setinden örnek rakam resimleri görünmektedir.



Şekil 3.6 MNIST örnek rakamlar

Çok boyutlu bir veri seti olduğundan gösterim yapmak için boyut düşürme yönteminden t-SNE metodu kullanılmıştır. t-SEN modelini oluşturmak için rastgele 1000 eğitim verisinden yararlanılmıştır. Oluşturulan model sonucunda veri setinin 2-boyutlu görüntüsü şekil 3.7'de verilmiştir. Her bir rakamın için farklı renk kullanılmıştır.



Şekil 3.7 MNIST veri seti

3.4 Veri Ön İşleme

Verilerin bir düzen içerisinde gösterimini sağlamak için veri setlerinde normalizasyon işlemi uygulanmıştır. Özellik ölçekleme (min-max scaling) veri normalizasyon yöntemi kullanılmıştır. Formül 3.2'de matematiksel ifadesi verilmiştir. Normalizasyon işlemi ile veriler [0-1] aralığına çekilmiştir. Böylece işlem kolaylığı sağlanmıştır.

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.2}$$

Skin segmentation veri seti tekrarlı verileri içerdiğinden tekrarlı veriler veri setinden çıkarılmıştır. Tekrarlı veriler çıkarıldıktan sonra veri setinde 51444 veri kalmıştır. Tekrarlı veriler çıkarıldıktan sonra 14654 tanesi ten (skin) 36790 adedi ten değil (no skin) verisi kalmıştır. Çizelge 3.3'te tekrarlı veriler ve tekrarlı veriler çıkarıldıktan sonraki veri sayısı özetlenmiştir.

Çizelge 3.3 Skin-no skin ön işleme

Sınıflar	Orijinal Veri seti	Tekrarsız Veri seti	Fark
Ten/Cilt	50859	14654	36205
Ten/Cilt Değil	194198	36790	157408

MNIST veri setinde veriler 28x28 resim formatında olduğu için her bir resim tek bir boyuta indirgenmiştir. Bunun için her bir 28'lik satır ilk satırın sonuna eklenerek her resim için 1x784'lik bir vektör elde edilmiştir. Daha sonrasında veriler 3.2'deki formül kullanılarak normalizasyon işlemine tabi tutulmuştur. Böylece hem boyut düşürme hem de normalizasyon işlemi bu veri setinde uygulanmıştır.

4. ARAŞTIRMA BULGULARI

Yapılan çalışmada OS-ELM'ın daha iyi bir genelleme ve sınıflandırma performansına sahip olabilmesi için OS-CELM adında yeni bir model önerilmiştir. OS-ELM ile önerilen modelin, OS-CELM, karşılaştırılması için her iki modele aynı eğitim ve test verisi uygulanmıştır. Her iki yöntem için model eğitilmiş ve performansları ölçülmüştür. Aktivasyon fonksiyonu olarak hiperbolik tanjant kullanılmıştır. Sıralı olarak nöron sayıları artırılmıştır ve her nöron sayısına göre 10 adet tur yapılmış ve bu turların sonunda modelin başarı değerlerinin ortalaması alınmıştır. Her test sonunca modelin başarım oranı ve ağırlıklı F_1 değerleri kaydedilmiştir. Ayrıca her tur için geçen zaman kaydedilmiş ve ortalama model eğitim süresi hesaplanmıştır.

Wireless Indoor Localization veri setinde 7 adet sinyal seviyesi özelliğinin ve bu verilerin alındığı 4 adet farklı oda özelliğinin bulunduğu toplam 2000 veri bulunmaktadır. Başlangıç verisi 200, diğer her sıralı eğitim için veriler 100'lük paketler halinde eğitime gönderilmiştir. Gizli katman sayısı sırası ile 25, 50, 75, 100, 125, 150 olarak ayarlanmıştır. Veri seti dengeli olduğu için F_1 değeri yerine model başarım oranları verilmiştir. Çizelge 4.1'de her bir turdaki modelin başarım oranları gösterilmiştir.

Çizelge 4.1 Wifi indoor localization tur değerleri

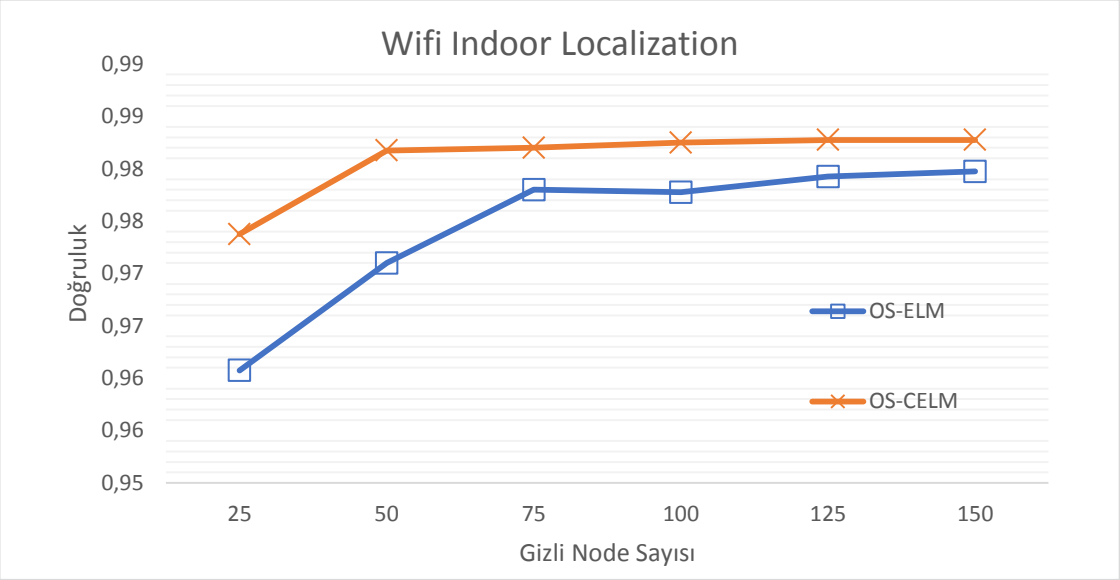
Nöron Sayısı		Turlar									
		1	2	3	4	5	6	7	8	9	10
25	OS-ELM	0.965	0.963	0.965	0.960	0.963	0.948	0.965	0.960	0.963	0.958
	OS-CELM	0.975	0.968	0.975	0.975	0.973	0.980	0.980	0.968	0.973	0.973
50	OS-ELM	0.975	0.973	0.968	0.970	0.970	0.973	0.970	0.976	0.970	0.973
	OS-CELM	0.983	0.983	0.980	0.980	0.983	0.980	0.983	0.983	0.983	0.983
75	OS-ELM	0.978	0.975	0.970	0.978	0.975	0.975	0.985	0.983	0.983	0.980
	OS-CELM	0.983	0.975	0.983	0.983	0.983	0.983	0.983	0.983	0.983	0.985
100	OS-ELM	0.978	0.978	0.983	0.980	0.978	0.983	0.983	0.968	0.983	0.968
	OS-CELM	0.980	0.983	0.983	0.978	0.983	0.985	0.983	0.983	0.985	0.985
125	OS-ELM	0.983	0.988	0.983	0.978	0.973	0.975	0.970	0.980	0.980	0.985
	OS-CELM	0.985	0.985	0.983	0.985	0.983	0.985	0.980	0.980	0.980	0.983
150	OS-ELM	0.975	0.985	0.983	0.980	0.980	0.985	0.975	0.978	0.978	0.980
	OS-CELM	0.983	0.980	0.985	0.983	0.980	0.980	0.985	0.985	0.983	0.983

Modellerin 10 tur sonunda ortalama başarım yüzdeleri ve ortalama model eğitim süreleri çizelge 4.2'de verilmiştir.

Çizelge 4.2 Wifi indoor localization başarım oranları ve eğitim süreleri

Wifi Indoor Localization				
Nöron Sayısı	OS-ELM		OS-CELM	
	Başarım Oranı	Eğitim Süresi (sn)	Başarım Oranı	Eğitim Süresi (sn)
25	0.960	0.0047	0.974	0.0079
50	0.971	0.0058	0.982	0.0098
75	0.978	0.0078	0.982	0.0114
100	0.978	0.0157	0.983	0.0220
125	0.979	0.0156	0.983	0.0281
150	0.980	0.0151	0.983	0.0413

Şekil 4.1'de başarım ortalaması grafik olarak verilmiştir. Her iki modelin başarım oranı oldukça yüksektir. Veri setindeki veri miktarı da dikkate alındığında OS-CELM'in daha iyi performans sergilediği görülmektedir. 150 nöron eğitimi 41.3 milisaniyede gerçekleşmektedir.



Şekil 4.1 OS-ELM / OS-CELM (Wifi localization)

Skin Segmentation veri setinde tekrarlı veriler arındırıldıktan sonra 51444 veri üzerinden işlemler yapılmıştır. Başlangıç verisi 400, diğer her sıralı eğitim için veriler 400'lük paketler halinde eğitime gönderilmiştir. Gizli katman sayısı sırası ile 25, 50, 75, 100, 125, 150 olarak ayarlanmıştır. Veri seti dengeli olmadığı için sınıflandırma başarım oranı yerine F_1 değerleri verilmiştir. Çizelge 4.3'de her bir turdaki modelin F_1 değerleri gösterilmiştir.

Çizelge 4.3 Skin/no skin tur F_1 değerleri

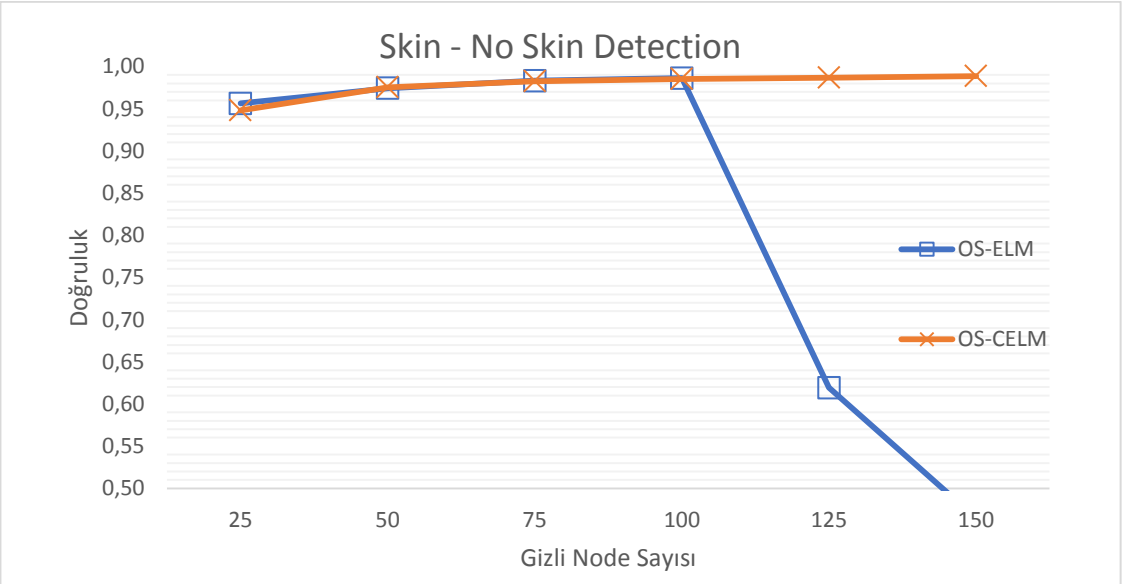
Nöron Sayısı		Turlar									
		1	2	3	4	5	6	7	8	9	10
25	OS-ELM	0.959	0.948	0.953	0.959	0.960	0.962	0.956	0.952	0.963	0.953
	OS-CELM	0.954	0.957	0.940	0.959	0.949	0.958	0.940	0.946	0.947	0.933
50	OS-ELM	0.973	0.976	0.975	0.971	0.976	0.976	0.976	0.975	0.978	0.967
	OS-CELM	0.976	0.973	0.979	0.979	0.980	0.974	0.975	0.972	0.975	0.971
75	OS-ELM	0.980	0.985	0.985	0.983	0.983	0.983	0.982	0.982	0.985	0.984
	OS-CELM	0.980	0.981	0.985	0.985	0.982	0.986	0.984	0.981	0.983	0.981
100	OS-ELM	0.986	0.987	0.987	0.987	0.986	0.982	0.987	0.986	0.987	0.987
	OS-CELM	0.984	0.984	0.985	0.987	0.983	0.985	0.986	0.986	0.986	0.986
125	OS-ELM	0.122	0.265	0.987	0.323	0.853	0.508	0.656	0.948	0.639	0.892
	OS-CELM	0.987	0.987	0.987	0.987	0.986	0.987	0.985	0.987	0.988	0.987
150	OS-ELM	0.408	0.608	0.604	0.121	0.588	0.203	0.575	0.649	0.605	0.283
	OS-CELM	0.986	0.990	0.989	0.988	0.991	0.990	0.988	0.987	0.991	0.987

Modellerin 10 tur sonunda ortalama F_1 ve ortalama model eğitim süreleri çizelge 4.4'de verilmiştir. 150 nöron kullanılarak oluşturulan modelin 10 tur sonunda ortalama F_1 değeri 0,9887'dir. Eğitim süresi ise ortalama 1.347 sn'dir.

Çizelge 4.4 Skin/no skin F_1 oranları ve eğitim süreleri

Skin/no skin				
Nöron Sayısı	OS-ELM		OS-CELM	
	Başarım Oranı	Eğitim Süresi (sn)	Başarım Oranı	Eğitim Süresi (sn)
25	0.956	0.5281	0.947	0.5313
50	0.974	0.5760	0.975	0.5680
75	0.983	0.6243	0.982	0.6133
100	0.986	0.6611	0.985	0.6716
125	0.654	0.7250	0.987	0.7601
150	0.501	0.7904	0.989	1.347

Şekil 4.2'de başarım ortalaması grafik olarak verilmiştir. OS-ELM nöron sayısı 100 üzerine çıktığında aşırı öğrenme (overfitting) gerçekleşip yeni verilere karşı duyarsızlaşmaya başlamıştır. Bu ise başarım oranını oldukça fazla etkilemiştir. Önerilen model olan OS-CELM'de aşırı öğrenme gerçekleşmediği gibi nöron sayısına bağlı olarak başarım performansının arttığı görülmüştür. OS-ELM 0.79 sn. eğitim gerçekleştirmiş buna karşılık OS-CELM 1.35 sn. model eğitimini gerçekleştirmiştir.



Şekil 4.2 OS-ELM / OS-CELM (Skin/No Skin)

MNIST veri setinde 70000 veri üzerinden işlemler yapılmıştır. Nöron sayıları sırası 256, 512, 1024, 2048, 4096'dır. İlk başlangıç eğitiminin yapılabilmesi için nöron sayısından fazla giriş verisine ihtiyaç vardır. Bu yüzden gizli katman nöron sayısı 256-1024 için 2000, 2048 nöron için 3000, 4096 nöron için 5000 başlangıç verisi kullanılmıştır. Sıralı eğitim için veri paket boyutu 2000'dir. Veri seti dengeli olduğu için F_1 değeri yerine model başarım oranları verilmiştir. Son olarak 10000 nöron kullanılarak tek tur eğitim yapılarak modelin başarım oranına ve eğitim süresine bakılmıştır çizelge 4.5'de her bir turdaki modelin başarım oranı verilmiştir.

Çizelge 4.5 MNIST tur başarım oranları

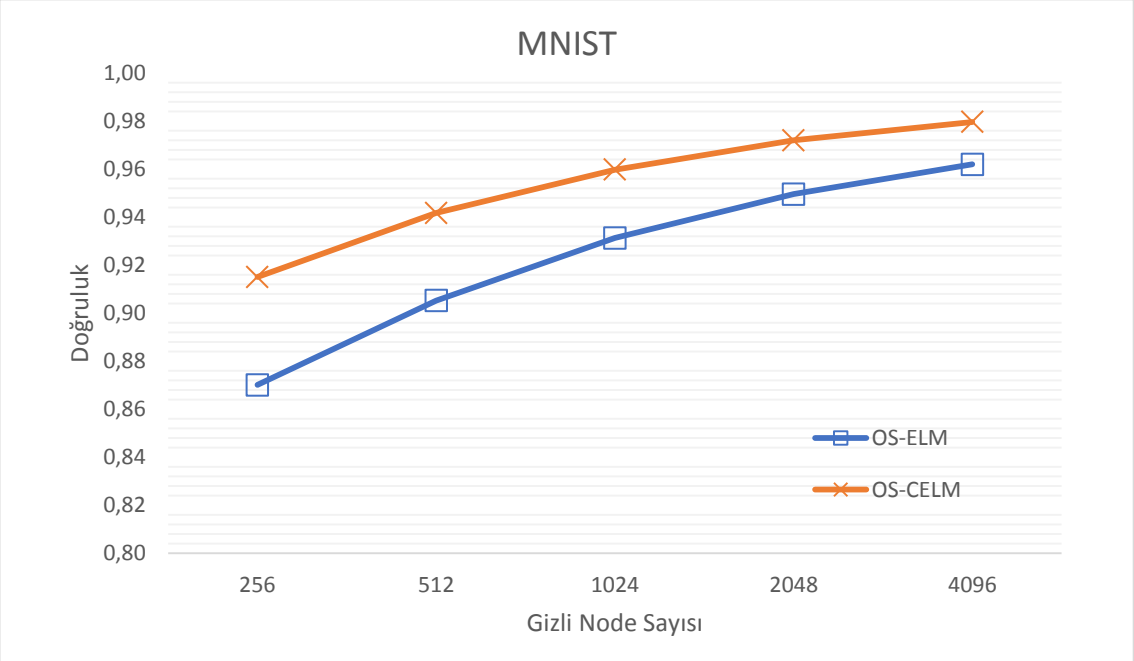
Nöron Sayısı		Turlar									
		1	2	3	4	5	6	7	8	9	10
256	OS-ELM	0.871	0.871	0.867	0.867	0.870	0.871	0.873	0.871	0.873	0.867
	OS-CELM	0.915	0.916	0.917	0.917	0.913	0.918	0.915	0.911	0.915	0.915
512	OS-ELM	0.907	0.905	0.904	0.905	0.907	0.908	0.904	0.908	0.902	0.903
	OS-CELM	0.944	0.941	0.941	0.943	0.940	0.941	0.941	0.941	0.941	0.943
1024	OS-ELM	0.931	0.933	0.932	0.931	0.932	0.931	0.929	0.929	0.934	0.931
	OS-CELM	0.960	0.960	0.961	0.960	0.959	0.959	0.960	0.959	0.960	0.959
2048	OS-ELM	0.951	0.950	0.951	0.950	0.947	0.948	0.950	0.950	0.948	0.951
	OS-CELM	0.972	0.972	0.972	0.973	0.971	0.971	0.972	0.972	0.971	0.973
4096	OS-ELM	0.962	0.964	0.962	0.963	0.961	0.962	0.960	0.961	0.961	0.964
	OS-CELM	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.980

Oluşturulan modellerin 10 tur sonunda ortalama başarım yüzdeleri ve ortalama model eğitim süreleri çizelge 4.6'de verilmiştir.

Çizelge 4.6 MNIST başarım oranları ve eğitim süreleri

MNIST				
Nöron Sayısı	OS-ELM		OS-CELM	
	Başarım Oranı	Eğitim Süresi (sn)	Başarım Oranı	Eğitim Süresi (sn)
256	0.870	11.4550	0.915	10.0764
512	0.905	14.6153	0.942	11.4921
1024	0.931	21.5518	0.960	17.0360
2048	0.950	38.1908	0.972	36.2503
4098	0.962	88.8612	0.980	88.0966
10000	0.9728	376.7580	0.986	374.7992

Şekil 4.3'de başarım ortalaması grafik olarak verilmiştir. Şekilden de görüleceği gibi OS-CELM ile oluşturulan model düşük nöron sayısında bile daha iyi bir genelleştirme ve sınıflandırma performansına sahiptir. Veri sayısının ve özelliklerin artmasıyla OS-CELM nispeten daha hızlı eğitim süresine sahip olmuştur.



Şekil 4.3 OS-ELM / OS-CELM (MNIST)

Önerilen OS-CELM modeli uygulanan veri setlerinde daha iyi sınıflandırma performansı sergilemiştir. ELM'in hız özelliklerini ve OS-ELM in çevrim içi ve sıralı öğrenme özelliklerini kaybettirmeden daha iyi bir genelleştirme performansına sahip olduğu görülmektedir.

5. TARTIŞMA

Yapılan bu çalışmada çevrim içi ve sıralı eğitilebilen OS-ELM modelinin genelleme ve performansını artırmak için rastgele atanan giriş parametreleri yerine sınıflar arasındaki farkları ortaya çıkarmaya yönelik ayarlanmış giriş parametreleri atayan OS-CELM modeli öne sürülmüştür. Üç ayrı veri seti üzerinde OS-CELM modeli ile OS-ELM modelinin performansları kıyaslanmıştır.

Wifi Indoor Localization veri setinin kullanıldığı bir çalışmada (Rohra vd. 2017) bulanık mantık kullanan hibrit bir yapay sinir ağı modeli öne sürülmüştür. Rohra vd. (2017) yaptıkları çalışmada yapay sinir ağı parametrelerini önerdikleri bulanık mantık kullanan algoritma ile ayarlamışlardır. Aynı veri seti üzerinde Sabancı vd. (2018) yaptıkları çalışmada nöron sayısı 1000-4000 arasında seçilen ELM modeli kullanılmıştır. Chen vd. (2019) yaptıkları çalışmada geliştirilmiş BP yöntemi öne sürülmüşlerdir. Bu yöntemde 7-10-10-4 şeklinde çift gizli katmana sahip bir yapay sinir ağı eğitilmiştir. Bu çalışmada önerilen model, OS-CELM, ile diğer çalışmalarda kullanılan modellerin karşılaştırması çizelge 5.1'de verilmiştir.

Çizelge 5.1 Wifi veri seti üzerinde model karşılaştırması

Kaynak	Kullanılan Metot	Model Doğruluk Oranı (%)
Rohra vd. (2017)	FPSOGSA	95.16
Sabancı vd. (2018)	ELM	96,27
Chen vd. (2019)	BP	91,48
Önerilen Model	OS-CELM	98,3

Skin/No-skin veri setinde Bhatt vd. 2009 yılında yaptıkları çalışmada bulanık mantık kullanarak karar ağacı modeli öne sürmüşlerdir. ELM tabanlı başka bir çalışmada (Çatak 2017) AdaBoost destekli ELM modeli kullanmıştır. Idoko vd. (2018), yapay sinir ağlarının bulanık mantık ile birlikte oluşturulduğu hibrit bir model olan ANFIS (Adaptive Neuro Fuzzy Inference Systems) modelini önermişlerdir. Ten alanı

bölümleme (skin segmentation) alanıyla ilgi yapay sinir ağlarının kullanıldığı Ma ve Shih (2018) tarafından yapılan çalışmada model olarak CNN kullanılmıştır. Üç farklı veri tabanından toplamda 5080 resmin kullanıldığı çalışmada %94,99 başarım oranı yakalanmıştır. Bu çalışmada dikkat çekici nokta ise çalışmanın 11GB belleğe sahip NVIDIA GeForce GTX 1080 Ti grafik kartının kullanıldığı grafik işlemcisi destekli bir ortamda yürütüldüğüdür. Bu çalışma 20 iterasyonda (epoch) yaklaşık iki saat sürmüştür. OS-CELM modelinin Skin segmentation veri setindeki eğitimlerinin hiçbiri üç saniyenin üzerinde olmamakla birlikte grafik işlemci desteği de kullanılmamıştır. OS-CELM ile diğer modellerin kıyaslaması çizelge 5.2 de verilmiştir.

Çizelge 5.2 Skin veri seti üzerinde model karşılaştırması

Kaynak	Kullanılan Metot	Model Doğruluk Oranı (%)
Bhatt vd. (2009)	Bulanık Karar Ağacı	94,1
Çatak (2017)	AdaBoost destekli ELM	98,92
Idoko vd. (2018)	ANFIS	90,1
Ma ve Shih (2018)	CNN	94,99
Önerilen Model	OS-CELM	98.9

MNIST veri setinde literatürde benzer çalışmalara bakıldığında Zhu vd. (2014) tarafından CELM modeli öne sürülmüştür. Yapılan çalışmada ELM ve CELM karşılaştırması yapılmıştır ve 200 nöron kullanılarak %85'lerde bir başarı elde edildiği belirtilmiştir. Bu tezde önerilen model 256 nöron ile %91,6'lık bir performans sergilemektedir. Bu tez çalışmasında da yararlanılan CELM modelini kullanarak Song vd. (2020) yaptıkları çalışmada çok katmanlı ELM modeli önermişlerdir. R^2CELM isimlerini verdikleri model ile 6 gizli katman ve her katmanda 2000 nöronlu toplamda 12000 nörondan oluşan bir yapay sinir ağı oluşturmuşlardır. OS-CELM modeli 10000 nöron ile de eğitilmiştir. Diğer çalışmalar ve önerilen modelin kıyaslaması çizelge 5.3'te verilmiştir.

Çizelge 5.3 MNIST veri seti üzerinde model karşılaştırması

Kaynak	Kullanılan Metot	Model Doğruluk Oranı (%)
Zhu vd. (2014)	CELM	91
Song vd. (2020)	R^2CELM	98,47
Önerilem Model	OS-CELM	98,62

Bu bölümde önerilen model ile aynı veri setleri üzerinde yapay sinir ağları kullanılarak yapılan çalışmaların karşılaştırmaları yapılmıştır. Buna göre önerilen model OS-CELM'in daha iyi bir performans sergilediği görülmüştür.

6. SONUÇ

Bu tez çalışmasında farklı veri setlerinde performansı ölçülmüş ve sınıflandırma başarısı diğer yöntemlerle yarışabilecek ELM tabanlı bir model önerilmiştir. OS-ELM algoritmasının rastgele atanan gizli katman giriş parametreleri CELM yöntemi ile hesaplanmış ve OS-CELM modeli oluşturulmuştur. OS-CELM gerçek dünya uygulamalarında kullanılabilecek tek katmanlı, implementasyonu kolay, hızlı bir yapay sinir ağı modelidir.

OS-ELM tabanlı algoritmalarda lineer olmayan sitemler lineer sistemler gibi ele alınıp en küçük kareler yöntemi ile çözülmeye çalışılmaktadır. Lineer sistemlerin çözülmesinde de matrislerin tersinir olması önemlidir. Yapılan bu tez çalışmasında önerilen OS-CELM modelinde de eşitlik 2.11'de verilen ($H^T H$) matrisi nadir olsa da tekil matris olmaktadır. Matris tekil olduğunda çözüme ulaşılamadığından ağırlık değerleri tekrardan hesaplanarak yeni bir ($H^T H$) matrisi oluşturulmaktadır. Önerilen modelin geliştirilebilmesi için ($H^T H$) matrisinin tekil olmamasını sağlayan daha etkili yöntemler bulunabilir.

KAYNAKLAR

Alpaydın, E. 2010. Introduction to machine learning. MIT Press, Cambridge.

Austin Community College (ACC). 2005. Nervous tissue. Web Sitesi: https://www.austincc.edu/histologyhelp/tissues/tx_nerv_tis.html, Erişim Tarihi: 05.07.2020.

Bhatt, R. B., Sharma, G., Dhall, A., and Chaudhury, S. 2009. Efficient skin region segmentation using low complexity fuzzy decision tree model. 2009 Annual IEEE India Conference.

Bhatt, R., Dhall, A., Sharma, G., Chaudhury, S. 2009. Efficient skin region segmentation using low complexity fuzzy decision tree model. Proceedings of INDICON an IEEE India Council Conference. 1 - 4.

Blake, C., Keogh, E. and Merz, C. J. 1998. UCI repository of machine learning databases. University of California, Irvine

Bryson A. E., Ho, Y.C. Siouris, G.M. 1979. Applied optimal control: optimization, estimation, and control. Systems, Man and Cybernetics, IEEE Transactions on. 9; 366-367.

Çatak, F.Ö. 2017. Classification with boosting of extreme learning machine over arbitrarily partitioned data. Soft Computing, 21; 2269–2281. https://doi.org/10.1007/s00500-015-1938-4

Chen J., Dong C., He G., and Zhang X. 2019. A method for indoor Wi-Fi location based on improved back propagation neural network. Turkish Journal of Electrical Engineering & Computer Sciences.

Chy, T. S., and Anisur Rahaman, M. 2019. A comparative analysis by KNN, SVM & ELM classification to detect sickle cell anemia. 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST). https://doi.org/10.1109/icrest.2019.8644410

Deng, W.Y., Zheng Q.H., Wang, Z.M. 2014. Cross-person activity recognition using reduced kernel extreme learning machine, Neural Networks. 53; 1–7.

Garcia, L. P.F., Sáez, J.A., Luengo, J., Lorena, A.C., Carvalho, A.C.P.L.F., and Herrera, F. 2015. Using the one-vs-one decomposition to improve the performance of class noise filters via an aggregation strategy in multi-class classification problems. Knowledge-Based Systems, 90; 153-164. https://doi.org/10.1016/j.knosys.2015.09.023.

Golub, G. H. and Loan, C. F. V. 1996. Matrix computations, The Johns Hopkins Univ. Press., Baltimore.

Goodfellow I., Bengio Y. and Courville A., 2016. Deep Learning. MIT Press

Herculano-Houzel S. 2009. The human brain in numbers: a linearly scaled-up primate brain. Front Hum Neurosci. 3:31.

Huang, G., Ding, X., and Zhou, H. 2010. Optimization method based extreme learning machine for classification. Neurocomputing, 74; 155-163. https://doi.org/10.1016/j.neucom.2010.02.019

Huang, G.B. Zhu, Q.Y. and Siew, C. 2004. Extreme learning machine: A new learning scheme of feedforward neural networks. IEEE International Conference on Neural Networks Conference Proceedings. 2; 985-990.

Idoko, J., Arslan, M. and Abiyev, R. 2018. Fuzzy neural system application to differential diagnosis of erythemato-squamous diseases. Cyprus Journal of Medical Sciences. 90-97.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11); 2278-2324. https://doi.org/10.1109/5.726791.

Liang, N., Huang, G., Saratchandran P., and Sundararajan, N. 2006. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Transactions on Neural Networks, 17(6); 1411-1423. doi: 10.1109/TNN.2006.880583.

Ma C.H, and Shih H. 2018. Human skin segmentation using fully convolutional neural networks. IEEE 7th Global Conference on Consumer Electronics. 168-170.

Maaten, L.V.D., Hinton, G. 2008. Visualizing high-dimensional data using t-SNE. Journal of Machine Learning Research 9;2579-2605.

McCarthy, J. 2007. What is artifical intelligent? Erişim tarihi: 03.07.2020, Erişim adresi: http://www-formal.stanford.edu/jmc/whatisai.pdf

Mohri, M., Rostamizadeh A. and Talwalkar A. 2018. Foundations of machine learning. MIT Press: Cambridge.

Negnevitsky, M. 2005. Artificial intelligence: a guide to intelligent systems. Pearson Education: Essex.

Nilsson N. J., 2005. Introduction to machine learning. Web Sitesi: https://ai.stanford.edu/~nilsson/MLBOOK.pdf. Erişim Tarihi: 05.07.2020.

Ojha, V.K, Abraham, A. and Snášel, V. 2017. Metaheuristic design of feedforward neural networks: a review of two decades of research. Engineering Applications of Artificial Intelligence. 60; 97–116

Pentreath, N. 2015. Machine learning with spark. Packt Publishing: Birmingham

Qiu, S., Gao, L., and Wang, J. 2015. Classification and regression of ELM, LVQ and SVM for e-nose data of strawberry juice. Journal of Food Engineering, 144; 77–85. https://doi.org/10.1016/j.jfoodeng.2014.07.015.

Rohra, J. G, Perumal, B., Swathi, J. N., Thakur P., and Bhatt R. 2017. User localization in an indoor environment using fuzzy hybrid of particle swarm optimization & gravitational search algorithm with neural networks. Proceedings of Sixth International Conference on Soft Computing for Problem Solving. 286-295.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review, 65; 386–408.

Sabanci, K. Yigit, E. Ustun, D., Toktas, A. and Aslan, M.F. 2018. WiFi based indoor localization: application and comparison of machine learning algorithms. XXIIIrd International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory (DIPED)

Shalev-Shwartz, S. and Ben-David S. 2014. Understanding machine learning from theory to algorithms. Cambridge University Press: New York.

Song, G., Dai, Q., Han, X. and Guo, L. 2020. Two novel ELM-based stacking deep models focused on image recognition. Applied Intelligence. 50; 1345–1366. https://doi.org/10.1007/s10489-019-01584-4

Turing, A. M. 1950. Computing machinery and intelligence. Mind, 49; 433-460.

Wang, Z., Xin, J., Yang, H., Tian, S., Yu, G., Xu, C., and Yao, Y. 2017. Distributed and weighted extreme learning machine for imbalanced big data learning. Tsinghua Science and Technology, 22(2); 160–173.

Xu S. And Wang J. 2016. A fast incremental extreme learning machine algorithm for data streams classification. Expert Systems with Applications, 65; 332–344.

Zhu, Q.Y., Qin, A.K., Suganthan, P.N. and Huang, G.B. 2005. Evolutionary extreme learning machine. Pattern Recognition, 38(10); 1759–1763.

Zhu, W., Miao, J. and Qing, L. 2014. Constrained extreme learning machine: A novel highly discriminative random feedforward neural network. International Joint Conference on Neural Networks (IJCNN), 800-807, Beijing. doi: 10.1109/IJCNN.2014.6889761.

ÖZGEÇMİŞ

Adı Soyadı : Olcay TOSUN

Doğum Yeri : Kırıkkale

Doğum Tarihi : 24/08/1985

Medeni Hali : Evli

Yabancı Dili : İngilizce

Eğitim Durumu

Lise : Çok Programlı Astsubay Hazırlama Okulu (2002)

Lisans : Anadolu Üniversitesi, İİBF, Kamu Yönetimi (2007)
Atatürk Üniversitesi, Mühendislik Fakültesi Bilgisayar Mühendisliği (2017)

Yüksek Lisans : Ankara Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği ABD (Ekim 2020)

Çalıştığı Kurumlar ve Yıl

Genelkurmay Elektronik Sistemler Komutanlığı (2003-2012)

Lojistik Yönetim Okul Komutanlığı (2012)

Bakım Okulu ve Eğitim Merkez Komutanlığı (2012-2014)

Sinop Üniversitesi Bilgi İşlem Daire Başkanlığı (2019-)