

**KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

**AKILLI ELEKTRİK ŞEBEKELERİ İÇİN DERİN ÖĞRENME KULLANILARAK ELEKTRİKLİ EV
ALETLERİNİN SINIFLANDIRILMASI**

DOKTORA TEZİ

VAHİT FERYAD

**EKİM 2021
TRABZON**



KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

**AKILLI ELEKTRİK ŞEBEKELERİ İÇİN DERİN ÖĞRENME KULLANILARAK
ELEKTRİKLİ EV ALETLERİNİN SINIFLANDIRILMASI**

Vahit FERYAD

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

Tezin Enstitüye Verildiği Tarih : 10 / 09 / 2021

Tezin Savunma Tarihi : 20 / 10 / 2021

Tez Danışmanı : Prof. Dr. İsmail Hakkı ÇAVDAR

Trabzon 2021

ÖNSÖZ

Müdahaleci olmayan yük izleme (MOYİ) veya enerji ayrıştırması (disaggregation), toplam düzeyde ölçülen ev enerjisini sayaçtan bileşen cihazlara ayırmayı amaçlar. Bu çalışmada; çeşitli evlerden toplanan elektrik kullanım bilgilerini içeren ve akademik araştırmalarda ortak kriter (benchmark) olarak kullanılan REDD adlı veri kümesi ile Derin Öğrenme kullanarak bir enerji ayrıştırma modeli tasarlanmıştır. Ayrıca akıllı şebeke ve akıllı ölçüm bağlamında, bu tez REDD bilgisine dayalı olarak ev aletlerini sınıflandırmasında bir müdahaleci olmayan (non-intrusive) yük izlemesi modeli önerilmektedir. MOYİ enerji yönetimi teknolojisinin talep yönündeki yoğun yükler sırasında kullanıcıların enerji talebini azaltarak üretim varlıklarının daha verimli kullanılmasına yardımcı olur. Böylece son kullanıcılara farkındalık kazandırmak ve pahalı akıllı priz sensörlerine ihtiyaç duymadan talep tarafından ayrıntılı geri bildirim oluşturmak için yüksek doğrulukta enerji ayrıştırma modeli tanıtılmıştır. Ayrıca tez kapsamında akıllı şebekede cihaz sınıflandırma ve akıllı enerji yönetimi araştırmalarını ilerletmeyi hedefleyen ve ev elektriği kullanım verilerini içeren Türkiye Elektrikli Ev Aletleri Veri Seti (TEEAVS) adı verdiğimiz bir yerli veri kümesi sunulmuştur.

Doktora eğitimim boyunca bilgi ve tecrübelerinden yararlandığım, katkılarıyla bana yol gösteren ve desteğini benden esirgemeyen danışman hocam Prof.Dr. İsmail Hakkı ÇAVDAR'a sonsuz teşekkürlerimi sunarım. Ayrıca hayatım boyunca beni destekleyen ve bu günlere ulaşmamı sağlayan aileme ve çalışmalarım süresince desteklerinden ötürü teşekkür ederim.

Vahit FERYAD

Trabzon, 2021

TEZ ETİK BEYANNAMESİ

Doktora Tezi olarak sunduğum "Akıllı Elektrik Şebekeleri İçin Derin Öğrenme Kullanarak Elektrikli Ev Aletlerinin Sınıflandırması" başlıklı bu çalışmayı baştan sona kadar danışmanım Prof.Dr. İsmail Hakkı ÇAVDAR'ın sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuvarlarda yaptığımı/yaptırdığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 20/10/2021

Vahit FERYAD

İÇİNDEKİLER

	Sayfa No
ÖNSÖZ	III
TEZ ETİK BEYANNAMESİ.....	IV
İÇİNDEKİLER.....	V
ÖZET	VII
SUMMARY	VIII
ŞEKİLLER DİZİNİ.....	IX
SAMBOLLER DİZİNİ.....	xv
1. GENEL BİLGİLER	1
1.1. Giriş.....	1
1.2. Literatür Özeti	5
1.3. Problem Açıklaması.....	8
1.3.1. Amaç	8
1.3.2. Enerji İzleme ve Takibi.....	9
1.3.2.1. Elektrik Enerjisi Terminolojisi.....	9
1.3.2.2. Güç Sistemleri.....	11
1.3.2.3. Akıllı Şebekeler.....	13
1.3.2.4. Sayaçlar.....	16
1.3.2.5. Akıllı Ölçüm	18
1.3.2.6. Cihaz Yük İzleme.....	20
1.3.2.7. Müdahaleci Olmayan Yük İzleme (MOYİ)	20
1.3.2.8. Müdahaleci Yük İzleme	22
1.3.2.9. Cihaz Yük Tipleri ve Analizde Örneklem Hızları	22
1.3.2.10. Veri Ön-İşleme ve Eşikleme Uygulamaları	24
2. YAPILAN ÇALIŞMA	25
2.1. Türkiye Elektrikli Ev Aletleri Veri Seti (TEEAVS).....	25
2.2. Nesnelerin İnterneti ile Canlı Veri Toplama Sistemi ve Kurulumu.....	26
2.3. Nesnelerin İnterneti Veri Toplama Donanımları ve Kurulumu	30
2.4. Veri Çerçevesi Oluşturması	34

2.5.	MOYİ Akademik Çalışmalarında Makine Öğrenimi.....	37
2.5.1.	Derin Öğrenme ve Gradyan İniş Optimizasyon Yöntemleri	37
2.5.1.1.	Yapay Zeka	37
2.5.1.2.	Makine Öğrenimi	38
2.5.1.3.	Derin Öğrenme.....	40
2.5.1.4.	Nöron Tahminleri ve Yapay Sinir Ağları	41
2.5.1.5.	Nöron Öğrenmesi.....	47
2.5.1.6.	Sinir Ağları Tahminleri	51
2.5.1.7.	Olasılıksal Dereceli Azalma.....	56
2.5.1.8.	Momentum	61
2.5.1.9.	Nesterov Momentum.....	63
2.5.1.10.	Değişken Öğrenme Oranları	65
2.5.1.11.	Adaptif Öğrenme Oranları	69
2.5.1.	LSTM Mimarisi	71
2.5.2.	MOYİ Geçmişi.....	72
2.5.3.	MOYİ Veri Kümesi	73
2.5.4.	Cihaz Sınıflandırıcının LSTM Model Yapısı.....	74
2.5.5.	Hata Fonksiyonu	75
2.5.6.	Optimizasyon Fonksiyonu	75
2.5.7.	ADAM Optimizasyon Algoritması.....	75
2.5.8.	AdaX Optimizasyon Algoritması.....	76
2.5.9.	LSTM Sınıflandırıcının Öğrenme Yapısı	78
2.5.10.	Model Başarı Metrikleri.....	79
2.5.11.	Önerilen BERT-MOYİ Transformer Mimarisi.....	80
3.	BULGULAR.....	83
3.1.	REDD Üzerinde BERT-MOYİ Yöntem Doğrulaması	86
4.	İRDELEME	90
4.1.	TEEAVS ile BERT-MOYİ Eğitimi ve Başarı Metrikleri.....	90
5.	SONUÇLAR	95
6.	ÖNERİLER.....	96
7.	KAYNAKLAR	97
ÖZGEÇMİŞ		

Doktora Tezi

ÖZET

AKILLI ELEKTRİK ŞEBEKELERİ İÇİN DERİN ÖĞRENME KULLANARAK ELEKTRİKLİ EV ALETLERİNİN SINIFLANDIRMASI

Vahit FERYAD

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı
Danışman: Prof. Prof.Dr. İsmail Hakkı ÇAVDAR
2021, 105 Sayfa

Akıllı elektrik şebekelerinde (smart grid) enerji talep tarafı yönetiminin (TTY) başarılı bir şekilde uygulanmasının temel koşullarından biri elektrik yük izleme sistemi ile farklı yüklerinin izlemesidir. Enerji ve sürdürülebilirlik sorunları, veri madenciliği ve makine öğrenmesi yaklaşımları kullanılarak ele alınabilecek çok sayıda sorunu beraberinde getirmektedir, Ancak, kamuya açık veri kümelerinin bulunmamasından dolayı bu tür problemlerin çözümü yavaş olmuştur. Bu çalışmada, önce verimli bir Enerji Ayırıştırma (energy disaggregation) modeli tasarlanmış ve Konutsal Enerji Ayırıştırma Veri Kümesi (REDD) adı verilen kamuya açık ortak kriter (benchmark) verilerine ve başarı metriklerine dayanarak model değerlendirmesi yapılmıştır, daha sonra akıllı şebekede cihaz sınıflandırma araştırmalarını ilerletmeyi hedefleyen ve ev elektriği kullanım verilerini içeren Türkiye Elektrikli Ev Aletleri Veri Seti (TEEAVS) adı verdiğimiz bir veri kümesi sunulmuştur. Ayrıca, bu çalışmada TEEAVS ile önerilen cihaz sınıflandırıcı modeli eğitilmiş ve performans değerlendirmesi yapılmıştır. Yapılan bu değerlendirmenin başarısı REDD veri seti ile yapılan performans testi ile doğrulanmıştır. Bu tezde veri toplamak için akım sensörleri ve Node-Red yazılım kurulumları ile Nesnelerin İnterneti (IoT) mimarisi kurulmuştur.

Anahtar Kelimeler: Enerji Ayırıştırması, Derin Sinir Ağları, Adaptif Öğrenme Optimizasyonu

PhD Thesis

SUMMARY

CLASSIFICATION OF HOME ELECTRICAL APPLIANCES USING DEEP LEARNING FOR
SMART GRID

Vahid FERYAD

Karadeniz Technical University
Institute of Science and Technology
Department of Electrical and Electronics Engineering
Supervisor: Prof. Prof.Dr. İsmail Hakkı ÇAVDAR
2021, 105 Pages

One of the basic conditions for the successful implementation of energy demand side management (TTY) in smart grids is the monitoring of different loads with an electrical load monitoring system. Energy and sustainability issues present a multitude of issues that can be addressed using data mining and machine learning approaches. However, resolution of such problems has been slow due to the lack of publicly available datasets. In this study, first an efficient Energy Disaggregation model was designed and model evaluation was performed based on publicly available benchmark data and success metrics called the Residential Energy Disaggregation Dataset (REDD). A dataset we call the Turkish Electrical Appliances Dataset (TEAD), which includes data on domestic and household electricity usage, is presented. In addition, in this study, the device classifier model proposed with TEAD was trained and performance evaluation was made. The success of this evaluation was confirmed by the performance test with the REDD data set. In this thesis, Internet of Things (IoT) architecture was established with current sensors and Node-Red software installations to collect data.

Key Words: Energy Disaggregation, Deep Neural Networks, Adaptive Learning Optimization

ŞEKİLLER DİZİNİ

	<u>Sayfa No</u>
Şekil 1. Ev 1 olarak ilk 2 günlük sinyal grafiklerinin çizimi.....	2
Şekil 2. Güç Sistemi Ağı	12
Şekil 3. Akıllı Şebeke	14
Şekil 4. HAM Fişi.....	21
Şekil 5. MOYİ işlem akışının şeması	26
Şekil 6. Node-Red Port Yönlendirme yapısı	27
Şekil 7. Dataplicity Port Yönlendirme platformu.....	28
Şekil 8. ACS-712 Akım Sensörü.....	30
Şekil 9. MQTT protokolü	30
Şekil 10. MQTT haberleşme akışı	31
Şekil 11. MQTT haberleşme adresi	31
Şekil 12. IoT MQTT görsel anlatımı	31
Şekil 13. MQTT broker	32
Şekil 14. IoT mimarisinin genel yapısı.....	32
Şekil 15. NodeMCU ESP8266 pin haritası	33
Şekil 16. KTÜ Gerçek Zamanlı Elektrik Güç İzleme ve Veri Toplama Sistemi	34
Şekil 17. Cihazların Toplam Enerji Tüketimi Çizimi	35
Şekil 18. Buz dolabının gerçek (koyu ince) ve tahmin (açık dalgalı) sinyallerin grafiği....	36
Şekil 19. Özellikler ve hedefin veri çerçevesi	42
Şekil 20. Doğrusal regresyonun genel ifadesi	42
Şekil 21. Regresyonun basit gösterimi	43
Şekil 22. Sınıflandırmanın basit gösterimi	43
Şekil 23. Regresyonun vektör gösterimi.....	44
Şekil 24. Sigmoid fonksiyonu	45
Şekil 25. Softmax fonksiyonu	45
Şekil 26. Kategorik çapraz entropi	46
Şekil 27. Kategorik çapraz entropi	47
Şekil 28. Öğrenme Oranı.....	48
Şekil 29. Hataların gradyanı	49
Şekil 30. Gradyanların vektör gösterimi.....	49

Şekil 31. Öğrenme adım büyüklüğü	50
Şekil 32. Çok katmanlı bir algılayıcı (MLP)	52
Şekil 33. MLP vektörlerle gösterimi	52
Şekil 34. MLP ve Çok Sınıflı SoftMax	53
Şekil 35. X , $N \times D$ ve Z , $N \times M$ büyük matrisleri.....	54
Şekil 36. Kayıpların döngüde hesaplanması.....	55
Şekil 37. Düzenleme cezası	56
Şekil 38. Kayıpların ortalama hesaplaması	57
Şekil 39. Yığınlar (batch) halinde Grdient	58
Şekil 40. Takviyeli öğrenme.....	59
Şekil 41. Öğrenmede takma durumları.....	60
Şekil 42. Momentum görsel benzetimi.....	61
Şekil 43. Sürtünme ifadesi.....	62
Şekil 44. Momentumlu gradyan inişi	62
Şekil 45. Momentumlu gradyan inişinin hareketleri	63
Şekil 46. Nesterov Momentumlu gradyan inişi.....	64
Şekil 47. Nesterov Momentumlu gradyan ifadeleri	64
Şekil 48. Nesterov ve Normal Momentumlular	65
Şekil 49. Öğrenme oranı planlaması.....	66
Şekil 50. Üstel azalma öğrenme oranı	66
Şekil 51. Üstel çürüme öğrenme oranı	66
Şekil 52. Optimal öğrenme adımlarının ayarlaması	67
Şekil 53. Geçmiş gradyanların önbellegi.....	68
Şekil 54. Momentumlu Gradyan inişi notasyonları.....	70
Şekil 55. LSTM ağıının mimarisi	71
Şekil 56. MOYİ sürecinin blok diyagramı	72
Şekil 57. REDD 2011.04.24 gününün buzdolabı grafiği.....	73
Şekil 58. Veri Kümesi Ayrımı.....	74
Şekil 59. LSTM katmanlarla gösterimi	74
Şekil 60. AdaX ile LSTM-MOYİ'nin Parametre Ayarlaması	78
Şekil 61. BERT mimarisi	80
Şekil 62. AdaX ile BERT-MOYİ Parametre Ayarlaması.....	82
Şekil 63. Ev 1'de belirlenen LSTM-MOYİ eğitim ve doğrulama hataları (losses).....	84

Şekil 64. Ev 1'deki validasyon seti ile LSTM-MOYİ eğitimin doğruluğu.....	85
Şekil 65. Adam ile ev 1'in 6 test gününde gerçek ve tahmini buzdolabı.....	85
Şekil 66. AdaX ile ev 1'in 6 test gününde gerçek ve tahmini buzdolabı	85
Şekil 67. SGD ile ev 1'in 6 test gününde gerçek ve tahmini buzdolabı.....	85
Şekil 68. BERT-NILM buzdolabı eğitiminde iterasyon ve kayıpları.....	87
Şekil 69. REDD ev 1'de AdaX'e göre buzdolabı başarı metrikleri.....	87
Şekil 70. BERT-NILM Çamaşır Makinesi iterasyon ve kayıpları	88
Şekil 71. REDD ev 1'de AdaX'e göre Çamaşır Makinesi başarı metrikleri	88
Şekil 72. BERT-NILM mikrodalga iterasyon ve kayıpları	89
Şekil 73. REDD ev 1'de AdaX'e göre mikrodalga başarı metrikleri	89
Şekil 74. BERT-NILM bulaşık makinesi iterasyon ve kayıpları	90
Şekil 75. REDD ev 1'de AdaX'e göre bulaşık makinesi başarı metrikleri	90
Şekil 76. BERT-NILM TV iterasyon ve kayıpları	91
Şekil 77. TEEAVS ev 2'de AdaX'e göre TV başarı metrikleri.....	91
Şekil 78. BERT-NILM buzdolabı iterasyon ve kayıpları.....	92
Şekil 79. TEEAVS ev 2'de AdaX'e göre buzdolabı başarı metrikleri	92
Şekil 80. BERT-NILM lambalar iterasyon ve kayıpları.....	93
Şekil 81. TEEAVS ev 2'de AdaX'e göre lambalar başarı metrikleri.....	93
Şekil 82. BERT-NILM çamaşır makinesi iterasyon ve kayıpları.....	94
Şekil 83. TEEAVS ev 2'de AdaX'e göre çamaşır makinesi başarı metrikleri	94

TABLÖLAR DİZİNİ

	<u>Sayfa No</u>
Tablo 1. Veri Kümesindeki Ev 1 için veri çerçevesi gösterimi.....	35
Tablo 2. TEEAVS ile Eğitilen MOYİ Modelinin Metrikleri	35
Tablo 3. Eğitim sırasında görülen evlerdeki enerji ayrıştırma performansları.....	84
Tablo 4. REDD test seti ile BERT-MOYİ ortalama başarı performansı.....	86
Tablo 5. REDD ev 1'deki AdaX'e göre buzdolabı başarı metrikleri	86
Tablo 6. REDD ev 1'deki AdaX'e göre çamaşır makinesi başarı metrikleri	87
Tablo 7. REDD ev 1'deki AdaX'e göre mikrodalga başarı metrikleri.....	88
Tablo 8. REDD ev 1'deki AdaX'e göre bulaşık makinesi başarı metrikleri	89
Tablo 9. TEEAVS ev 2'deki AdaX'e göre tüm cihazların ortalama başarı metrikleri.....	90
Tablo 10. TEEAVS ev 2'deki AdaX'e göre TV başarı metrikleri	91
Tablo 11. TEEAVS ev 2'deki AdaX'e göre Buzdolabı başarı metrikleri	92
Tablo 12. TEEAVS ev 2'deki AdaX'e göre lambalar başarı metrikleri.....	92
Tablo 13. TEEAVS ev 2'deki AdaX'e göre çamaşır makinesi başarı metrikleri.....	93

SEMBOLLER DİZİNİ

MA	Ortalama Doğruluk
MP	Ortalama Keskinlik
MR	Ortalama Duyarlılık
$MF1$	Ortalama F1 Puanı
MRE	Ortalama Göreceli Hata
MAE	Ortalama Mutlak Hata
$\sigma(t)$	Açıklanamayan gürültü
$y_t^{(i)}$	Tüketim gücü
MSE	Ortalama kare hatası
LR	Öğrenme Oranı
m_t	İlk moment
v_t	İkinci moment
β_1, β_2	Gradyanın birinci ve ikinci anlarındaki ilk çürüme (decay) oranları
g_t	Hataların gradyanı
$\frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}$	Zamanla azalan geçmiş ağırlık değerlerinin “önbelleği”
η	Öğrenme adımı boyutu

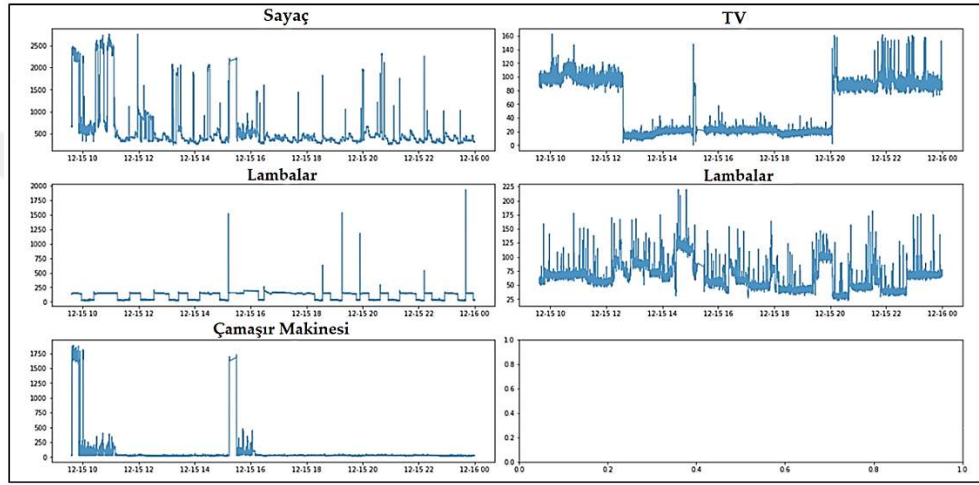
1. GENEL BİLGİLER

1.1. Giriş

Enerji ve sürdürülebilirlik sorunları, toplumun karşılaştığı en büyük zorluklardan birini temsil ediyor. Dünyadaki enerjinin % 83'ünden fazlası (sürdürülemez) fosil yakıtlardan geliyor; rüzgar, güneş enerjisi, jeotermal ve biyokütleden yenilenebilir enerji toplamın sadece % 2'sini oluşturuyor. Bu arada, enerjiye olan talep sürekli artıyor: dünya genelinde enerji üretimi, 1987'den 2007'ye kadar olan 20 yılda % 46 arttı [1]. Mevcut enerji kaynaklarımızın basit fiziksel sınırlarının yanı sıra, büyük miktarlarda fosil yakıt yakmanın çevresel ve iklim üzerindeki etkisi, sürdürülebilirliği zorunlu kılan konular üzerinde araştırma yapmaya odaklanmaktadır. Ayrıca, sürdürülebilirlik konusunda temel olarak veri analizi ve tahmin görevleri, veri madenciliği ve makine öğreniminden elde edilen tekniklerin paha biçilmez olduğunu kanıtlayabilecekleri birçok sorun vardır. Sürdürülebilirlik araştırmalarının önemine ve veri madenciliği ve makine öğrenimi tekniklerinin uygunluğuna rağmen, bu alanlarda, en azından hesaplamalı biyoloji veya makine görme gibi diğer uygulama alanlarıyla karşılaştırıldığında nispeten az bir çalışma yapılmıştır [2]. Bu makale, bu durumun en azından kısmen bu tür alanlar için halka açık verilerin azlığından kaynaklandığını iddia etmektedir. Örneğin, enerji alanları (gelişmiş ülkelerdeki her bina ve ev halkının enerji tüketimi, her elektrik iletim ve dağıtım hattının yükü) ile ilgili çok miktarda veri olmasına rağmen, bu verilerin çoğunluğu araştırmacılar için mevcut değildir. Ayrıca, kamuya açık veri setlerinin makine öğrenimi ve veri madenciliğinde önceki uygulama alanlarını teşvik ettiğine dair önemli kanıtlar vardır: biyolojik uygulamalara, biyolojik dergilerin ve devlet kuruluşlarının veri paylaşım zorunlulukları büyük ölçüde yardımcı olmuştur [3, 4].

Doğal dil işlemedeki birçok erken başarılar, şu andaki klasik Wall Street Journal corpus tarafından yapıldı [5] ve makine görme araştırmalarına, MNIST rakam tanıma [6], CalTech 101 [7] ve PASCAL challenge [8] gibi ortak kriter veri setleri tarafından büyük ölçüde desteklenmiştir. Enerji ve sürdürülebilirlik alanları için bu aynı hedefe doğru bir miktar ilerleme olmasına rağmen [9], şu anda makine öğrenmesi ve veri madenciliği topluluklarına yönelik bu tür veri setleri mevcut değildir. Bu tezde, TEEAVS olarak adlandırılan, bu türden bir veri seti geliştirmede çalışmalar sunulmuştur. Veriler özellikle

enerji ayrıştırma görevine yöneliktir, TEEAVS, birkaç ay boyunca çok sayıda gerçek ev için bütün ev ve cihaza özgü elektrik tüketiminden oluşur. İzlenen her ev için, tüm ev elektrik sinyalleri dijital akım sensörler ile loglanmıştır. Ayrıca, evde 5 adede kadar fiş seviyesi izlemesi, devre halinde gruplandığı cihazların günlüğe kaydedilmesine odaklanarak 1 Hz'de kaydedilmiştir. Bu veri tipine bir örnek Şekil 1'de TEEAVS'nin birinci evdeki tarihi 2020-12-15 olan cihaz verilerin çizimini ve günlük zaman ölçeğinde cihaz aktivitelerini gösterilmektedir.



Şekil 1. TEEAVS'nin birinci evdeki tarihi 2020-12-15 olan cihaz verilerin çizimi
(Yatay eksenler: saat-saat dakika, düşey eksenler: W)

Ticari ve konutlar, dünyadaki elektriğin yaklaşık %60'ını tüketmektedir. Örneğin, binalar Amerika Birleşik Devletleri'nde üretilen elektriğin %74,9'unu kullanır ve Afrika'daki bu rakam% 56'dır. 2050 yılına kadar inşaat sektörünün enerji talebinin ve çatılı alanlarda enerji kullanımının yaklaşık% 80 oranında artacağı tahmin edilmektedir. Bu rakamlar, binalarda verimli enerji kullanımı yoluyla enerji talebini azaltma potansiyelini vurgulamaktadır [10]. Konutlardaki etkin verimli enerji kullanımı, elektrikli ev aletlerinin enerji tüketiminin gerçek zamanlı izlenmesi ile yapılabilir, ayrıca üç kullanıcılara, belirli zamanlarda hangi elektrikli cihazların kullanılması gerektiği, ne kadar ve elektrikli cihazların enerji tüketiminin nedeni olduğu konusundaki farkındalıklarını arttırmaları için gerçek zamanlı geri bildirimler sağlanabilir [11]. Bu şekilde, uç kullanıcılar, enerji israfını kontrol etme ve önleme konusunda aktif bir rol oynama fırsatına sahip olacaktır. Ayrıca, enerjinin verimli kullanımına katkıda bulunma biçimleri, enerji tasarrufu için

motivasyonlarının bir ölçüsü olacaktır. Araştırmalar, bir hanehalkına gerçek zamanlı geri bildirimin yanı sıra enerji tüketimi konusundaki farkındalığın olumlu değişiklikler yarattığını ve uç kullanıcıları sürdürülebilir enerji kullanımına yönlendirdiğini göstermiştir [12]. Genel olarak, elektrikli cihazların anlık enerji kullanımı, her cihazın enerji tüketimini belirleyen akıllı priz sensörleri kullanılarak belirlenebilir. Bu tür cihazlar pahalıdır ve benzersiz iletişim protokolleri talep eder [13].

Günümüzde akıllı sensörlerin büyük ölçüde kullanılması Müdahaleci olmayan yük izlemesi yöntemlerinin geliştirilmesine dikkat çekmiştir [14]. Bu yöntemler, evdeki her elektrikli cihazın enerji kullanımını belirlemek için akıllı bir sensör gibi tekil bir veri kaynağından toplanmış enerji kullanımı verilerinin kullanıldığı hesaplama yöntemlerini ifade eder. Bu yöntemler elektrikli ev aletlerinin verimli enerji tüketiminin gerçek zamanlı olarak izlenmesini sağlayarak ev maliyetlerini düşürmeyi garanti eder, enerjiyi tasarrufu ve koruma şansı sağlarlar. Ayrıca, MOYİ sistemi politika yapıcılarının enerji performans stratejilerinin başarısını ölçmelerine ve enerji taleplerini tahmin etmelerine yardımcı olmaktadır. Bu şekilde, tedarikçiler talepleri optimum şekilde karşılamayı planlayabilirler [15]. MOYİ konsepti on yıl eskidir. Ancak, son zamanlarda, veri iletişimindeki, ağlardaki, algılama teknolojisindeki, makine öğrenimindeki, GPU'lardaki ve derin öğrenme yöntemlerindeki paralel gelişmelerden ilham alan bu araştırma alanına ilgi duyulmuştur. MOYİ, cihazları tanımlamak ve konut tüketicilerine enerji geri bildirimi sağlamak için ana önkoşuldur. Bununla birlikte, uzaktan yük izleme ve arıza tespit servislerinde uygulanabilirliği nedeniyle, müdahaleci (intrusive) ölçüm ve pahalı akıllı sensörlere ihtiyaç duymadan sanayi sektörü için aynı derecede faydalıdır. Hart, 1990'lı yıllarda konut birimlerinin enerji tüketimini ayırtırmak için birincil MOYİ tekniklerini tanıttı [16]. Daha sonra, araştırmacılar enerji ayırtırması için farklı yöntemler önerdiler ve birincil ayırtırma planını geliştirdiler [17]. Gizli Markov modeli (GMM) [18], çizge sinyal işleme [19] ve derin öğrenme [20] gibi çalışmalarda farklı yöntemler izlenerek, birçok gelişmiş MOYİ algoritması önerilmiştir. 2012'den önce, görüntülerin sınıflandırılması için özellikleri çıkartmada kullanılan en iyi teknik, Ölçek-Değişimsiz Özellik Dönüşümü (SIFT) ve Gaussian farkı gibi el yapımı (hand-crafted) dedektörleri içermektedir [21].

2012 yılında Krizhevsky et al. tarafından önerilen AlexNet, ImageNet Büyük Ölçekli Görsel Tanıma yarışmasını % 15'lik bir hata puanı ile, % 26'lık bir hata puanıyla ikinci algoritma tarafından kovalandı. Krizhevsky et al. ve arkadaşlarının kazandığı algoritma [22], el yapımı bir özellik detektörü kullanmamıştır. Alternatif bir yöntem olarak Krizhevsky, ham

bir görüntüden bir özellik hiyerarşisini otomatik olarak çıkarmak için eğitilmiş bir derin sinir ağı (DNN) kullandı. Günümüzde derin öğrenme kullanımı görüntü sınıflandırması ile sınırlı değildir ve makine çevirisi, otomatik konuşma tanıma ve hatta bilgisayar oyunları oynamayı öğrenme için kullanılmaktadır [23]. Bu çalışma, DNN'lerin enerji ayrıştırması için kullanılma yeteneğini incelemektedir. Roos et al. (1999) [24] ilk kez küçük sinir ağlarında MOYİ kullanmıştır. Söz konusu çalışma sadece [25] 'de yer alan bir öneriydi. Ancak, küçük bir sinir ağı, hiyerarşi özelliklerinin tanınmasını öğretmek için uygun bir seçenek olarak görünmüyordu. GPU'nun DNN öğrenme gücünü dev bir veri setinde hesaplamak için tanıtılmasıyla görüntü sınıflandırmada büyük bir yenilik yaşandı. GMM, sinir ağları ve optimizasyon gibi geleneksel yöntemlerin yanı sıra, derin öğrenme yöntemleri, cihazları yüksek hassasiyetle tanımak için derinlemesine öğrenme yeteneğine sahiptir. Derin öğrenme algoritmalarının avantajları otomatik özellik öğrenme, çok katmanlı özellik öğrenme, yüksek doğruluk, yüksek genelleştirme gücü ve Nvidia, Google'ın donanım ve yazılım desteğidir. Bu noktada, DNN'leri enerji ayrıştırması için istenen bir performansa sahip olma açısından inceledik. Bu çalışmanın en büyük katkısı, DNN'ler alanındaki MOYİ'ye Transformerden bir Kodlayıcı Temsilleri modeli eklemektir.

1.2. Literatür Özeti

Günümüzde devamlı artan enerji talebini karşılamak için, üretim kapasitesini arttırırken elektrik güç tüketimini izlemek ve kullanımını azaltmak esastır. Aslında, yük ve enerji yönetimi çok önemlidir, bu nedenle daha yüksek potansiyellere ve daha iyi sonuçlara sahip talep yönü enerji yönetimi daha yaygındır. Yönetim sisteminin ev sektörüne müdahalesi, yükleri ayırt ederek hem kullanıcı hem de elektrik şirketi tarafından yük yönetimini sağlayabilir. Örneğin, yüksek güç talebi olan soğutma ve ısıtma cihazları gibi cihazların sistem tarafından pik saatlerde kontrol edilmesi, daha büyük bir kullanıcı grubuna minimum düzeyde enerji sağlamamızı sağlayacaktır. Ayrıca enerji yönetim sistemi [26], kullanıcının şebekeye bağlı her cihazın davranışını anlamasına yardımcı olabilir ve hem şebekeyi hem de kullanıcının enerji kullanımını daha iyi yönetmesini kolaylaştırır. Büyük endüstrilerdeki sistem müdahalesi, yoğun saatlerde yük üzerinde doğru kontrol, maksimum talep kontrolü, yasadışı eylemlerin önlenmesi ve daha doğru tarifelerin uygulanması şeklinde de iyi sonuçlar verir [27]. MOYİ'nin yük izlemesi, ev toplu tüketim modeline dayalı olarak ev aletlerinin bireysel kullanımını tahmin etmek için kullanılan bir yöntemdir. Bu, pahalı sensörler kullanmadan pratik olarak ev aletleri tüketim izleme yeteneği sunar.

Zaman içinde güç ölçümünden kaynaklanan enerji yüklerinin ayrıştırılması, elektrik dağıtım hizmetleri için olduğu kadar elektrik hizmetlerinin kendileri için de kullanılabilir ve tüketicilere kişiselleştirilmiş hizmetler sunarken daha iyi bir kullanıcı ihtiyaç tahmini sağlar [28]. Garcia ve diğerleri tarafından tanıtılan MOYİ yöntemi. [29] 1980'lerin ortalarında, ev aletlerinin AÇIK ve KAPALI durumlarını belirlemek için aktif ve reaktif gücün geçici analizini kullanan ilk kişiydi. Bugüne kadar, konuyla ilgili birkaç kapsamlı incelemeyi içeren birçok orijinal araştırma makalesi yayınlanmıştır [30,31]. Souza ve diğerleri tarafından açıklandığı gibi ayrıştırma işlemi [32] üç aşamayı içerir: (1) olayların tanımlanması, (2) sınıflandırma için optimal özelliklerin sentezi ve (3) gerçek yük sınıflandırması ile enerji ayrıştırması. Yükleri sınıflandırmak için kullanılan özellikleri sentezlemek için ev aletlerinin durumundaki değişikliklerle ilgili olaylar kullanılmıştır. Massidda ve ark. [33], öznitelik örnekleme hızına dayalı olarak bir karakterizasyon sunmuşlar ve bunlar şu şekilde bölünmüştür: (1): bir dakikadan az, (2): bir dakika arasında, bir saniye, temel frekansta 1 Hz'den yüksek, 2 kHz'e kadar ve (3): 2 ila 40 KHz arasında. Bir dakika/bir saniye ile bir dakikadan az arasındaki kategorilerin özellikleri, zaman serilerinin alt dizilerinin istatistiksel karakterizasyonu olarak doğrudan kullanılabilir [34]. Ayrıca, önceki örnekleme oranları

kategorizasyonu bağlamında sinyal işleme [35], yüksek frekanslı örnekleme hızı, ev aletleri tüketiminde [36] daha ayrıntılı geçişleri karakterize etmeyi sağlar. Yüksek frekanslı örnekleme hızında, Hızlı Fourier Dönüşümü (FFT) veya Ayrık Dalgacık Dönüşümü (DWT) gibi sinyal dönüşümünün uygulanması, sınıflandırmayı temel alan önemli yeni özelliklerin geri kazanılmasıyla sonuçlanacaktır [37]. Yüksek örnekleme frekansı oranı, ev aletleri hakkında yük dalga formu verileri sağlar. Örneğin, Wang ve diğerleri tarafından bildirildiği gibi voltaj-akım hesaplamasının yörüngelerinde [38].

Yoğun olarak, yüksek örnekleme oranları, elektrik gürültüsü ile birlikte yoğun bir harmonik setinin elde edilmesini sağlar [39]. Az sayıda çalışma, tüketim ölçümlerinden elde edilen özellikleri, ev aletleri kullanım sıklığı [40] veya hava koşulları [41] gibi diğer veri kümeleriyle bütünleştirmiştir. Ayrıştırma işlemlerinin son aşaması, çıkarılan özelliklerden yük içi tanımlamayı (in-loads identification) içerir. İlgili araştırma topluluklarında zorluklara işaret eden bu aşama hakkında literatürde birçok yaklaşım tanıtılmıştır. İlk topluluk birleşimsel (combinatorial) arama için optimizasyon algoritmaları kullanır [42], ancak gerekli hesaplama kaynakları bu algoritmaları sınırlar. Sonuç olarak topluluk, denetimli ve denetimsiz (supervised and unsupervised machine learning) makine öğrenimi yöntemlerine odaklandı. Çok Katmanlı Algılayıcı (MLP) [43], Aşırı Öğrenme Makinesi [44], Evrişimli Sinir Ağı (CNN) [45] ve Tekrarlayan Sinir Ağı gibi Sinir Ağı (NN) mimarileri ile bazı denetimli öğrenme yöntemleri daha önce sunulmuştur. RNN [46], K-En Yakın Komşular (KNN) [47], Destek Vektör Makineleri (SVM) [48], Rastgele Orman sınıflandırıcısı [49], Naive Bayes sınıflandırıcıları [50] ve Koşullu Rastgele Alanlara dayalı yöntemler [51]. Denetimsiz öğrenme temel olarak ilgili alanda kullanılan Gizli Markov Modeli'ne (HMM) dayanıyordu [52], ancak kümeleme algoritmaları da kullanıldı [53].

Modern MOYİ yöntemleri genellikle makine öğrenmesi veya optimizasyon algoritmaları kullanılarak gerçekleştirilir [54]. Örüntü tanıma stratejileri bire bir eşleştirmeye en uygun olanıdır ve yine de bu teknikler, yanlış algılamayı ortaya çıkarabilen gürültülü sinyal kenarlarına karşı hassastır. Optimizasyon algoritmaları, yanlış kenarların algılanmasına karşı daha az hassasiyetle gelişmiş MOYİ performansına izin verir. Machlev et al. [55] cihazları verilen yük profillerine göre tanımlayan evrimsel optimizasyon yaklaşımları önermiştir. Programın fikri, potansiyel cihaz profillerinin verilen yük profili ile minimum hatayla eşleşmesi gerektiğiydi [56]. NP-zor olan Sırt Çantası problemi ile ilgili tanıtılan problem [57], bununla birlikte, NP-zor problem karmaşıklığı nedeniyle hesaplama performansının gerçek zamanlı durumlarda kullanılması hala oldukça zordur. Ek olarak, [58]

makalede, web servislerinin API'sine ve internet bağlantısına ihtiyaç duymadan, herhangi bir Mbed kartı destekleyebilen uTensor adlı son teknoloji bir çıkarım kütüphanesi kullanarak düşük maliyetli bir gömülü kart üzerinde MOYİ için FOG Hesaplama Kenarda (edge) bir derin öğrenimi tanıttı.

Bu yazıda, enerji ayrıştırması için en etkili optimize edici tarafından ayarlanmış BERT tabanlı MOYİ'yi (BERT-MOYİ) uygulamayı amaçladık. BERT, Google tarafından geliştirilen, doğal dil işleme ve ön eğitim için dönüştürücü (transformer) tabanlı bir makine öğrenimi tekniğidir. BERT-MOYİ'nin iki optimizasyon modeli (BERT-MOYİ Adam ve BERT-MOYİ AdaX) uygulanmış ve enerji ayrıştırma sistemi için karşılaştırılmıştır. Derin öğrenme modellerinin eğitiminde Adam, stokastik gradyan inişi için bir alternatif optimizasyon algoritması olarak kullanıldı, AdaX ise yeni bir uyarlanabilir gradyan iniş algoritması önererek Adam'ı geliştirdi. Geçmiş gradyanları görmezden gelen Adam'ın aksine, AdaX öğrenme oranlarını uyarlanabilir bir şekilde ayarlamak için eğitim sırasında uzun vadeli gradyan bilgilerini katlanarak biriktirir. REDD ve TEEAVS veri kümeleri, tasarlanan modellerin eğitimi ve etkinliğinin doğrulanması için kullanılmıştır. Bu çalışma, yakın gelecekte IoT akıllı sensörler zaten kuruluysa etkinleştirilebilen müdahaleci olmayan “enerji farkındalığı” yoluyla tüketimi azaltmak için önlemler almak için ayrıntılı enerji verilerini kullanarak enerji verimliliğini izleyen enerji ayrıştırma yöntemlerini tanıttı. Ayrıca sağlık sektörlerinde kullanılmak üzere gelişen alternatif bir teknoloji haline gelen ev içi faaliyetler hakkında da bilgi sağlayabilir.

1.3. Problem Açıklaması

Ev aletlerinin enerji tüketimini artırmak, müşteriler için daha fazla elektrik faturasına yol açacaktır. Tüketiciler, enerji tüketim alışkanlıklarını iyileştirmek veya karbon ayak izlerini azaltmak için elektrikli cihazların davranışlarını ve tüketim kalıplarını anlamaya çalışmalıdır. Bu, tüketicilerin cihazlarını nasıl kullanacaklarına karar verebilmeleri için her bir cihazın nasıl çalıştığını ve ne kadar güç tükettiğini önceden bilmeleri gerektiği anlamına gelir. Bu alan veya araştırma, yük kontrolü olarak tanımlanır. Genel olarak, yük kontrolü, bir güç sisteminde yük ölçümünü tanımlama ve elde etme sürecidir. Bu prosedürün nihai amacı, sistem / tesis genelinde bireysel yüklerin davranışını anlamaktır. Yük kontrolü, cihazların kontrolünde kullanılan yöntemle bağlı olarak müdahaleci (intrusive) yük izleme (MYİ) ve müdahaleci olmayan yani MOYİ olarak ikiye ayrılır. Müdahale terimi, müşterinin varsayımlarına ve varsayımlarına dayanan ölçüm cihazlarının genellikle her seferinde izlenmekte olan cihazın yakınına yerleştirildiği anlamına gelir.

Bu tez, düşük kaliteli, düşük maliyetli ölçüm ekipmanı kullanarak bir veya daha fazla elektrikli cihazın güç tüketimini ölçmeyi içeren ilk yöntemle odaklanmaktadır. Bu çalışma, güç tüketimi modellerini makine öğrenimi tekniklerini kullanarak ev yüklerinden ayırmak ve tesislerdeki elektrikli cihazları tanımak/sınıflandırmak için kapsamlı bir çözüm sağlar. Ayrıca çalışma kapsamında doğru bir veri seti elde etmek için her bir cihazın enerji tüketimi kaydetmek için Node-Red kullanarak bir IoT tabanlı veri toplama sistemi ile ESP8266 WiFi ve ACS712 akım ölçüm sensörleri kullanılmıştır.

1.3.1. Amaç

Bölüm 1.3.'de belirtildiği gibi, bu tezin temel amacı, uygulanan yük kontrol yöntemleri üzerine araştırmalara yardımcı olmaktır. Diğer çalışmaların çoğu, farklı veri örnekleri için bir model ve algoritmayı incelerken, bu çalışma aynı veriler üzerinde farklı model ve algoritmaların testlerini içerir, böylece yöntemler arasında nicel karşılaştırmalar mümkündür. Uygulanan modeller arasındaki farklılıklar temel olarak yapılarına, tiplerine ve her bir örnekte kullanılan katmanların sayısına ve ayrıca hiper-parametrelerine odaklanmıştır. Modellerin türü söz konusu olduğunda, araştırmadaki en önemli değişiklik, sorunun zamana bağlı doğasından sorumlu olan modelin ilk katmanıdır. Bu amaçla, bu tezde uzun süreli bellek ve kapı özyinelemeli sinir ağları olmak üzere iki farklı seçenek

ele alınmıştır. İlk katman dışında, modelin performansını etkileyen ana faktörlerden biri olduğu için çeşitli hiper-parametrelerin uygulamasının etkinliği incelenmiştir. Son olarak araştırmalara göre sinir ağının en önemli özelliklerinden biri ara katman sayısıdır. Bu tez çalışmasında, bu özelliğin modellerin performansına etkisi, farklı modeller arasında değişen yapay nöron sayısı ile birlikte incelenmiştir.

1.3.2. Enerji İzleme ve Takibi

Bu bölümün iki ana direği, güç sistemleri ve makine öğrenimidir. Enerji tüketiminin, akıllı şebekelerin ve akıllı ölçümün temel kavramlarının ve en önemli yönlerinin ayrıntılı bir açıklamasını sunulmuştur. Odak noktamız temel olarak yük kontrolüne müdahale etmektir, dolayısıyla bu çalışmada bazı yöntem ve teknikler kullanılmıştır. Ek olarak, derin öğrenmeyle birlikte makine öğrenimi teorisi tanıtılır ve yapay sinir ağlarına genel bir bakış ve bunların sınıflandırılması sağlanmıştır.

1.3.2.1. Elektrik Enerjisi Terminolojisi

Enerji birçok biçimde gelebilir ve birçok farklı şey yapabilir. Enerji sürecinin yönetimine rehberlik eden önemli doğa kanunlarından biri, enerjinin ne üretildiği ne de yok edildiği, sadece bir formdan diğerine dönüştüğüdür. Sistemdeki temel enerji birimi jul (J) 'dir. Enerji biçimleri kimyasal enerji, nükleer enerji, ısı enerjisi, mekanik enerji ve elektrik enerjisini içerir. Elektrik enerjisi, elektrik akımının bir tel aracılığıyla hareketini içerir. Elektrik enerjisi çok kullanışlı bir enerji şeklidir çünkü birçok işlevi yerine getirebilir. Sonunda, elektrik enerjisinin çoğu, somut ısı biçiminde ısı (termal) enerjiye dönüştürülür. Elektrikli ısıtıcılar gibi bazı cihazlar enerjiyi doğrudan dönüştürür; Motorlar gibi diğer cihazlar, elektriği mekanik enerjiye dönüştürür ve bu da sonunda ısıya dönüştürülür. Güç tüketimini optimize etmenin püf noktası, ısıya dönüştürülmeden önce elektrik tarafından yapılan iş miktarını en üst düzeye çıkarmaktır.

Bu tezin metnini derinlemesine anlamak için güç ve enerji arasındaki temel farkı açıklamak önemlidir, çünkü bu iki terim aşağıdaki çalışmada yaygın olarak kullanılmıştır. Güç, enerji akışının miktarı veya hızıdır. Başka bir deyişle, bu güç enerjinin ne kadar hızlı kullanıldığını veya dönüştürüldüğünü ölçer. Temel güç birimi (J/s)'dir ve saniyede bir jul ise James Watt (1819-1736) olarak adlandırılan bir watt'a (W) eşittir. Elektrik gücünü

tanımlamanın başka bir yöntemi, iki akım ve voltaj miktarlarının çarpımıdır, dirençle birlikte elektronların hareketini açıklayabilir, aksi takdirde elektrik olarak adlandırılır. Voltaj, bir elektrik devresindeki olayları gerçekleştiren potansiyel enerji kaynağıdır. Bazen elektrikli tahrik veya EMF olarak adlandırılır. Temel voltaj birimi voltur (V). Volt, adını pili icat eden İtalyan fizikçi Alessandro Giuseppe Antonio Anastasio Volta'dan (1827-1745) almıştır.

Gerilim, elektrik güç sistemlerinin potansiyel enerji kaynağıdır. Voltaj tek başına çalışmaz, fakat basınç veya kuvvettir; Ve her zaman iki nokta arasında görünür. Doğal olarak, voltaj ya sabittir (DC) ya da değişkendir (AC). Elektrik güç sistemleri, alçak gerilim konut sistemlerinden yüksek gerilim iletim sistemlerine kadar AC uygulamalarına dayanmaktadır. Elektrik güç sistemlerinde yüksek voltaj ve alçak voltaj uygulamaları söz konusudur, ancak dağıtım ve tüketim yoluyla üretimi kapsamak için standart bir aralık kullanılır. Akım, bir iletkenindeki (tel) elektronların akımıdır. Elektronlar, voltajla bir elektrik devresi veya kapalı döngü yolu üzerinden itilir ve çekilir. Bir iletkenindeki elektronlar her zaman voltaj kaynaklarına geri döner. Elektrik akımı genellikle "I" sembolü ile gösterilir ve amper (A) cinsinden ölçülür. Bir döngüde veya devrede elektron sayısı asla azalmaz. Bir iletkenindeki elektron akışı, iletkenin direncinden dolayı ısı oluşturur. Gerilim her zaman akımı iletir veya çeker, bu nedenle, tam bir devre veya kapalı bir döngü yolu sağlandığında, voltaj bir akımın akmasına neden olur. Devredeki direnç, enerjinin bir kısmı ısıya dönüştürülürken akım miktarını azaltacaktır. Sonuç olarak, voltaj kaynağının potansiyel enerjisi, bir elektron akışı biçiminde kinetik enerjiye dönüştürülür. Kinetik enerji daha sonra yük (yani tüketim cihazları) tarafından kullanılır ve diğer enerji türlerine dönüştürülür. Gerilim sabit kalırsa, teldeki (yani iletken) direnç sürtünmeye neden olur ve akım miktarını azaltır. Bu elektriksel sürtünmenin yan ürünleri ısı ve ışıktır.

Direnç birimine ohm denir. Ampuller, tost makineleri, elektrikli sıcak su ısıtıcıları ve vb. dirençli yüklerdir. İndüksiyon yüklerinin çalışması için bir manyetik alan gerekir. Manyetik alan oluşturmak için bobin kullanan tüm elektrik yüklerine endüktif yükler denir. Endüktif yüklerin örnekleri arasında saç kurutma makineleri, fanlar, karıştırıcılar, elektrikli süpürgeler ve diğer birçok motorlu cihaz bulunur. Prensip olarak, tüm motorlar endüktif yüklerdir. Endüktif yükler ile diğer yük türleri arasındaki benzersiz fark, akımın endüktif bir yükte uygulanan voltajı geciktirir. Gerilim uygulandığında, indüklenen yüklerde bir manyetik alanın üretimi zaman alır, bu nedenle akım gecikir. İndüksiyon birimi (H) Henry adlandırılır. Elektrik motorları ile ilgili olarak, bir işlemi gerçekleştirmek için dönen bir shafta yerleştirilen bir yük, elektrik enerjisi kaynağından aktif güç (P) olarak adlandırılan şeyi

çeker. Aktif güce ek olarak, motorda manyetik alanlar oluşturmak için elektrik enerjisi kaynağından reaktif güç (Q) de alınır. Motor tarafından tüketilen toplam güç, hem aktif gücün hem de reaktif gücün toplamına eşittir. Aktif güçle ilişkili elektrik gücü birimleri şunlardır: Watt, reaktif güç VAR'ları ile ilişkili olanlar ise (kısaltma, volt-amper-reaktif anlamına gelir).

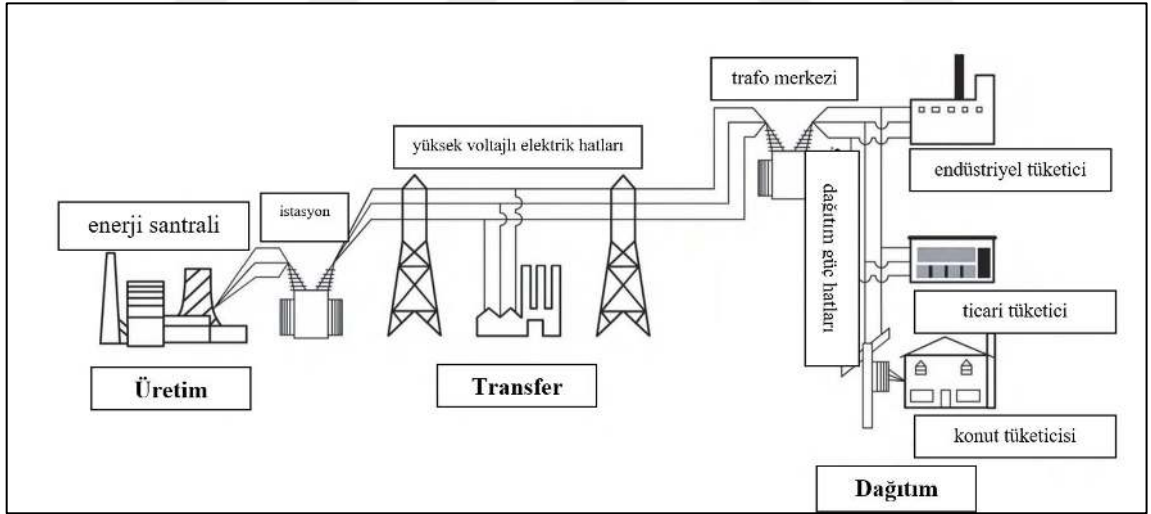
Kapasitör, dielektrik adı verilen bir yalıtkanla (örneğin kâğıt, cam ve diğer iletken olmayan malzemeler) ayrılan iki metal iletkenle oluşan bir cihazdır. Bu dielektrik malzemeler, bağlı iletkenlere voltaj uygulandığında yüklenir. Voltaj kaynağı çıkarıldıktan sonra kapasitörler uzun bir süre şarjlı kalır. Kapasitif yük örnekleri, TV resim tüpleri, uzun güç kabloları ve elektrikli cihazlarda kullanılan bileşenlerdir. İndüktörlerin aksine, kapasitörlerle ilişkili akım, dielektrik malzemenin şarj akımından tam gerilime kadar şarj olması için geçen süre nedeniyle gerilime (gecikmek yerine) yol açar. Bu nedenle, bir kapasitördeki akımın gerilime yol açtığı söylenir. Kapasitans ölçü birimlerine Farad (F) denir. İndüktörlere benzer şekilde, kapasitörlerle ilişkili güç de reaktif güç olarak adlandırılır, ancak indüktörlerin reaktif gücünden zıt kutuplara sahiptir. Bu nedenle, indüktörlerin pozitif VAR'ları ve kapasitörlerin negatif VAR'ları vardır. İndüktörlerin negatif VAR'larının, kondansatörlerin pozitif VAR'ları tarafından iptal edilebileceğini ve bunun net sıfır reaktif güç gereksinimi doğuracağını unutmamalım. Genel bir kural olarak, kapasitif yükler, insanların dirençli ve endüktif yükler gibi büyük miktarlarda mağazada satın aldıkları öğeler değildir. Bu nedenle, enerji şirketleri, endüktif talep ile reaktif güç dengesini sürdürmek için düzenli olarak kendi büyük ölçekli kapasitörlerini kurmak zorundadır.

1.3.2.2. Güç Sistemleri

Elektrik güç sistemleri anlık ve gerçek zamanlı enerji sağlayan sistemlerdir. Anlık ve gerçek zamanlı, güç anahtarını açtığımız anda elektrik üretildiği, iletildiği ve tedarik edildiği anlamına gelir. Elektrik güç sistemleri esas olarak gaz ve su sistemleri gibi depolama sistemlerini desteklemez. Aksine, jeneratörler ihtiyaç duydukları enerjiyi sağlar. Tam ölçekli gerçek bir birbirine bağlı elektrik Güç Sistemi, şekil 2'de gösterilenden çok daha karmaşıktır; ancak temel ilkeler, kavramlar, teoriler ve terminolojiler aynıdır.

Santralde elektrik enerjisinin üretildiği üretimle başlar. Santraller, elektrik enerjisi üretme sürecinde diğer enerji kaynaklarını dönüştürür. Örneğin elektrik enerjisi üretiminde

ısı, mekanik, hidrolik, kimyasal, güneş, rüzgar, jeotermal, nükleer ve diğer enerji kaynakları kullanılır. Jeneratörler döndükçe, yaklaşık 150 kV voltajla elektrik üretirler. Bir sonraki adım, elektrik enerjisinin bedelini ödeyen müşterilere aktarılmasıdır. Elektrik uzun mesafeler kat ettiğinde, daha yüksek voltajlara dönüştürülmek daha verimlidir. Bu amaçla, elektrik önce Yüksek Gerilim (HV) ve ekstra yüksek gerilim (EHV) iletim hatları üzerinden dağıtım için bir transformatör aracılığıyla voltajı 400 kV'a kadar yükselten elektrik santralindeki bir trafo merkezine gidiyor. Bu iletim hatları elektriği uzun mesafeler için iletim merkezlerine taşır. Bu iletim trafo merkezlerinde, birincil dağıtım ağına bağlanan dağıtım trafo merkezlerine yüksek gerilim iletim hatları vasıtasıyla Güç Sistemindeki diğer noktalara dağıtım için voltajda bir azalma meydana gelir. Bu trafo merkezleri, bu HV elektrik enerjisini, daha uygun dağıtım güç hatları üzerinden konut, ticari ve endüstriyel tüketim için nihai varış bölgesine iletilen Alçak Gerilim (AG) enerjisine dönüştürür. Modern güç şebekeleri son derece karmaşık ve yaygındır. Elektrik hatlarındaki dalgalanmalar, büyük şebeke arızalarına ve elektrik üretim tesislerindeki multi milyon dolarlık ekipmanlarda kalıcı hasara neden olabilir.



Şekil 2. Güç Sistemi Ağı

Tüm elektrik güç sistemleri, bir elektrik santralinde elektrik enerjisinin üretildiği üretimden başlar. Santrallerde elektrik üretme sürecinde diğer enerji kaynakları dönüştürülür. Örneğin, elektrik enerjisi üretmek için termal, mekanik, hidrolik, kimyasal, güneş, rüzgar, jeotermal, nükleer ve diğer enerji kaynakları kullanılır. Jeneratörler, dönüş

sırasında yaklaşık 150 kW voltajda elektrik üretir. Bir sonraki adım, elektriği ödeyen müşterilere (abonelere) transfer etmektir.

Elektrik uzun mesafeler kat ettiğinde, yüksek voltajlarda iletimi daha verimli hale gelir. Bu amaçla elektrik, yüksek gerilim (HV) ve çok yüksek gerilim (EHV) iletim hatları üzerinden dağıtım için önce bir transformatör aracılığıyla gerilimi 400 kW'a kadar yükselten bir elektrik santraline gider. İletim hatları, elektriği uzun mesafelerde iletim istasyonlarına taşır. Bu iletim trafo merkezlerinde, güç sistemindeki diğer noktalara dağıtım için gerilim düşüşleri, ana dağıtım ağına bağlı dağıtım trafo merkezlerine güçlü gerilim iletim hatları vasıtasıyla gerçekleşir. Bu trafo merkezleri, yüksek gerilim elektrik enerjisini, elektrik dağıtım hatları aracılığıyla konut, ticari ve endüstriyel kullanım için nihai varış noktasına iletilen düşük gerilim (AG) enerjisine dönüştürür. Modern güç ızgaraları çok karmaşık ve kapsamlıdır. Elektrik hatlarındaki dalgalar, yaygın ağ arızalarına ve milyonlarca dolarlık elektrik santrali ekipmanında kalıcı hasara neden olabilir.

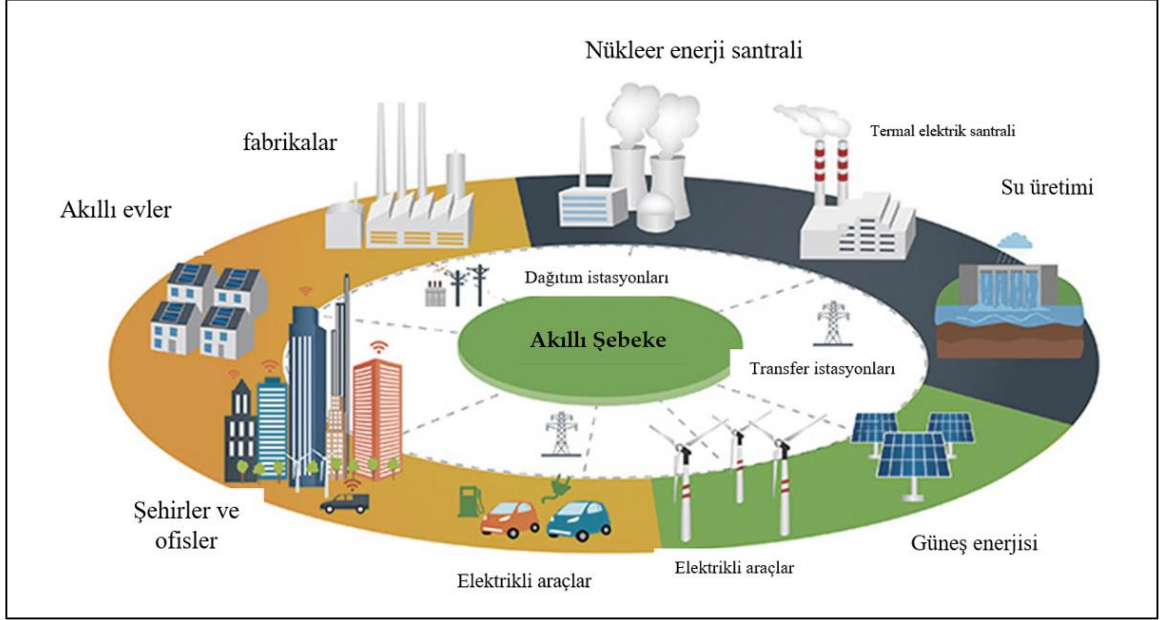
1.3.2.3. Akıllı Şebekeler

Şebeke terimi geleneksel olarak elektrik üretimi, iletimi, dağıtımı ve kontrolü için kullanılır. Akıllı Şebeke, geleneksel elektrik Güç Şebekesinin geliştirilmiş halidir. Bu, güç dağıtım sisteminin modernizasyonu, eski tek yönlü elektrik şebekesinin çift yönlü elektrik gücü ve bilgi alışverişi için otomatik akıllı sisteme dönüştürülmesidir. Akıllı Şebeke, tüketicilere teslimat altyapısını (veya Şebekeyi) toplu olarak daha güvenilir, daha çok yönlü, daha uyumlu, daha esnek ve nihayetinde daha kullanışlı hale getiren teknolojilerin, donanımların, yazılımların veya uygulamaların herhangi bir kombinasyonu olarak tanımlanabilir. Akıllı Şebeke, temel olarak fiziksel güç sistemini bilgi sistemi ile kaplamaktan oluşur. Teknik açıdan Akıllı Şebeke üç ana sisteme ayrılabilir:

Akıllı altyapı sistemi: Bu, Akıllı Şebekenin altında yatan enerji, bilgi ve iletişim altyapısıdır. Bu, iki yönlü elektrik ve bilgi akışına izin verir. Bu, kullanıcıların şebekeye geri elektrik sağlayabileceği anlamına gelir. Sistem, birden fazla varlığın (akıllı cihazlar, özel yazılım, kontrol merkezi vb.) Etkileşimde bulunmasını sağlar.

Akıllı yönetim sistemi: Bu, gelişmiş yönetim ve kontrol hizmetleri sağlar. Akıllı şebekelerin verimli çalışması için verimli yönetim esastır. Akıllı Şebeke Yönetimi, akıllı ölçümün geliştirilmesini ve uygulanmasını, gerçek zamanlı fiyatlandırmayı, yenilenebilir enerji kaynaklarının verimli yönetimini ve iletim ve dağıtım ağlarının yönetimini içerir.

Akıllı koruma sistemi: Bu, gelişmiş güvenilirlik analizi, hata koruması ve güvenlik hizmetleri sağlar. Mevcut altyapı, çeşitli güvenlik tehditlerine karşı savunmasız hale gelmiştir.



Şekil 3. Akıllı Şebeke

Akıllı Şebeke, elektrik şebekesine sensörler, akıllı sayaçlar ve sahada otomasyon cihazlar uygulanarak mümkün hale getirilmiştir. Akıllı Şebekeler kendilerini güvenilir ve verimli bir şekilde tahmin edebilir, uyarlayabilir ve yeniden yapılandırabilir. Zamanlamalardaki belirsizlikleri, bölgeler arasında güç aktarımını yönetebilir, öngörülemez olayları yönetebilir ve çözebilir ve güvenilir tedarik talebini karşılayabilirler. Bunu yaparken, Akıllı Şebekeler hem Güç Şebekesine hem de enerji tüketicilerine (i) iletim sıkışıklığında azalma, (ii) kesinti ve zorunlu kesintilerde azalma, (iii) kendi kendine teşhis, (iv) kendi kendini iyileştirme gibi çeşitli avantajlar sunar. , (v) en yüksek talep azalması, (vi) artan sistem kapasitesi, (vii) artan güç sistemi güvenliği ve azaltılmış güvenlik açığı ve (viii) hibrit ve elektrikli araçların yönetiminde gibi kolaylık sağlamaktadır. Akıllı Şebekenin çalışması için vazgeçilmez olan hususlar şunlardır: enerji depolama, gelişmiş sayaçlar ve sensörler, şebeke-dostu fişli (plug-in) hibritler, şebeke dostu yükler, trafo merkezi ve dağıtım otomasyonu, iletişim, veri yoğunluk analizi, görselleştirme ve insan arayüzü ve aşağıdakileri içeren yenilenebilir enerji entegrasyonu hava tahmini ve kontrolü.

Akıllı Şebekelerin uygulanması, teknolojik ve finansal yatırım içerdiği için uzun ve sürekli bir süreç olacaktır. Akıllı Şebekenin geliştirilmesi, mevcut elektrik şebekesinin değiştirilmesini içermez. Teknik ve ekonomik nedenlerle böyle bir süreç imkansız olurdu. Bunun yerine, Akıllı Şebeke geliştirme, eski fiziksel altyapıyı mümkün olduğunca korurken, yeni hizmetler ve özellikler uygulayarak mevcut ağın iyileştirilmesidir. Aynı zamanda, uluslararası çabalar gerektirir, bu da her ülkenin hükümetinin Şebekelerini Akıllı Şebekeye yükseltmek için bir politika geliştirmesi gerektiği anlamına gelir.

Burada, Akıllı Şebekenin sağlaması gereken en önemli işlevlerden bazılarını tanımlıyoruz:

1. Tüketici katılımı: Tüketiciler, elektrik güç sisteminin ayrılmaz bir parçası haline gelir. Arz ve talebi dengelemeye yardımcı olurlar ve güvenilirliği sağlarlar: elektrik kullanma ve satın alma yöntemlerini değiştirmek. Bu değişiklikler, tüketicilerin farklı satın alma modellerini ve davranışlarını motive eden seçeneklere sahip olmasının bir sonucu olarak ortaya çıkar. Bu seçenekler, yeni teknolojileri, elektrik kullanımları hakkında yeni bilgileri ve yeni elektrik fiyatlandırma biçimlerini ve teşvikleri içerir.

2. Yüksek kaliteli güç: Tüm ticari işletmeler ve kesinlikle tüm konut müşterileri aynı kalitede güce ihtiyaç duymaz. Akıllı Şebeke, çeşitli güç seviyeleri sağlar ve buna göre değişken fiyatlandırmayı destekler. Birinci sınıf güç kalitesi özelliklerinin maliyeti, elektrik servis sözleşmesine dahil edilebilir. Gelişmiş kontrol yöntemleri, temel bileşenleri izleyerek yıldırım, anahtarlama dalgalanmaları, hat arızaları ve harmonik kaynaklardan ortaya çıktıklarında güç kalitesi olaylarına hızlı teşhis ve hassas çözümler. Akıllı Şebeke ayrıca, elektrik sistemini tüketici elektroniği yüklerinin neden olduğu düzensizliklerden korumaya yardımcı olur.

3. Esnek çalışma: Esneklik, bir sistemin, sistemin geri kalanı normal çalışmaya geri getirilirken sorunlu unsurların izole edilmesi gibi olaylara tepki verme yeteneğini ifade eder. Bu kendi kendini iyileştirme eylemleri, tüketicilere verilen hizmet kesintisinin azalmasıyla sonuçlanır ve hizmet sağlayıcıların teslimat altyapısını daha iyi yönetmesine yardımcı olur. Akıllı Şebeke, ister başkaları tarafından organize edilmiş isterse doğal afetlerin sonucu olsun, saldırılara dirençli bir şekilde yanıt verir. Bu tehditler, fiziksel saldırıları ve siber saldırıları içerir. Akıllı Şebeke, tüm unsurlar için bir gereklilik olarak güvenliği en başından ele alır ve sistem genelinde entegre ve dengeli bir yaklaşım sağlar.

4. Farklı depolama ve üretim türleri için destek: Akıllı Şebeke yalnızca büyük, merkezi enerji santrallerini değil, aynı zamanda büyüyen dağıtılmış enerji kaynakları (DEK) dizisini

de barındırır. DEK entegrasyonu, tedarikçilerden pazarlamacılara ve müşterilere kadar tüm değer zinciri boyunca hızla artacaktır.

Ayrıca, yük şekillendirme (shaping), Güç Şebekelerinde önemli ve zorlu konulardan biridir. Akıllı Şebekeler, enerji depolamaya ve dinamik fiyatlandırmaya dayalı yeni yük şekillendirme stratejileri kullanır. Bir tüketici, Şebekeden nominal bir enerji kotası için kaydolacaktır. Kullanım kotayı aşarsa, tüketici daha yüksek bir elektrik fiyatı ile karşı karşıya kalır. Enerji depolaması yapıldığında tüketici, talebe ve fiyata bağlı olarak şarj etme ve boşaltma akışını değiştirerek enerji tüketimini optimize edebilir. Bu, optimum yük şekillendirmeyi sağlar. Bu stratejiler düşük karmaşıklıkla ve dağıtılmış bir şekilde uygulanırsa, çok sayıda tüketiciye ölçeklenebilirlik mümkündür.

1.3.2.4. Sayaçlar

Endüstri tabiriyle, her elektrik sayacı, modeli veya türü ne olursa olsun bir "son nokta" (end-point) dır. Farklı şirketlerden temin edilebilen çeşitli sayaç türleri vardır. Ölçüm cihazlarından bazıları yükseltilebilir veya çeşitli şekillerde çalışacak şekilde yapılandırılabilir, bu nedenle model numarası işlevselliği temsil etmez. Sayaçların hepsi birbirine benziyor, ancak yetenekleri büyük farklılıklar gösteriyor. Aşağıdaki geniş kategorilere ayrılırlar:

- Mekanik sayaçlar: Yüzyılı aşkın süredir kullanımda olan sayaçlardır. Bir tür elektrik motoru gibi çalışırlar, sayaçtan geçen akım mekanik bir sayaç çalıştıran bir çarkı döndürür. Basit ve güvenilirlerdir, ancak yalnızca zaman içinde kullanılan toplam elektrik miktarını ölçebilirler. Daha karmaşık bir şeye ihtiyaç duyduğunuzda, bir yükseltme gerektirecektir. Bu, çalışma saatleri dışında oranın düşürüldüğü kullanım süresi ölçümünü içerir. Bu ölçüm cihazlarından bazıları elektronik eklentiler kullanılarak yükseltilebilir.

- Basit dijital sayaçlar: Düz dijital sayaç, geleneksel mekanik sayacın elektronik bir versiyonudur. Mekanik bir tekerlek yerine, bir sensör tarafından kullanılan elektriği kaydeder. Bu sayaçlar genellikle elektrik fiyatının kapalı saatlerde daha düşük olduğu basit günün saati ölçümü için kullanılır. Bu türden bir sayaç, mekanik sayaç gibi görsel olarak veya ön tarafa monte edilmiş bir port kullanılarak okunabilir. Mekanik sayaçlar gibi bir sayaç okuyucu tarafından okunmalıdır. İçeride kablosuz alıcı veya verici yok. Çoğu modelin ön tarafında, ölçülen değeri okumak ve bazen programlamak için kullanılan bir kızılötesi iletişim bağlantı noktası vardır. Sayaç okuyucu, bağlantı noktasına takılan bir çubuk kullanır.

- Yükseltilmiş mekanik sayaçlar: Eski mekanik sayaçlara yükseltme satan bazı satıcılar vardır. Küçük bir devre kartı tipik olarak sayaç muhafazasının içindeki mevcut mekanik mekanizmanın etrafına takılır. Genellikle, tüketilen elektriğin doğru bir ölçüsü olan mekanik rotorun dönme sayısını sayarak sayacı okur ve ardından devre kartı verileri bir şekilde iletir. Bir sayacın yükseltilip yükseltilmediğini görmek mümkün olmayabilir. Çoğu yükseltilmiş mekanik sayaç, dijital bir ekrana sahip olmama eğilimindedir. Yükseltmeler, daha az karmaşık olma eğiliminde olsalar da, çeşitli iletişim yöntemlerini kullanıyor gibi görünüyor.

- Uzaktan okunan sayaçlar: Uzaktan okunan sayaçlar genellikle Otomatik Sayaç Okuma (OSO) olarak adlandırılır. Bu sayaçlar, her sayacın yakınında yürümek veya sürmek için bir sayaç okuyucusuna ihtiyaç duymaması dışında, yukarıda belirtilen dijital sayaçlar gibi çalışır. Bunun yerine verilerini iletebilirler. Yaygın iletim yöntemleri telefon hattı, radyo, hücreli (GPRS), güç hattı (PLC) veya kablosuz (Wi-Fi) ağıdır. Bazı modellere Elektronik Alıcı Verici (EAV) denir. Bu modeller tipik olarak ayda bir küçük bir radyo alıcısı ile mahalleyi dolaşan bir araçtan okunur. Bir kablosuz bilgisayar, araç geçerken sayaçlar tarafından iletilen yanıtları alır. Bu sayaçların çoğu, kamyonun geçip geçmediğine bakılmaksızın her zaman iletim yapar. Bunlar bazen "kabarcıklanma" ölçerler olarak adlandırılır. Bazı sayaçlar, iletime başlamadan önce geçen bir ticari araçtan gelen sinyali dinleyen yerleşik bir alıcıya sahiptir. Bunlar bazen "uyanma" ölçüm cihazları olarak adlandırılır. Diğer sayaç modelleri, belirli bir saatte, genellikle her gün gece yarısı civarında iletim yapacak şekilde programlanabilir. Bu modellerde, bir bilgisayarı çağırdıkları yerleşik bir telefon modemi olabilir. Veya yerel alandaki bir baz istasyonuna kablosuz bir verici kullanarak veya elektrik hattı üzerinden sinyaller göndererek bağlanırlar. Daha sık, bazen 15 saniyede bir kadar sık ileten modeller de vardır. Bu sayaçlar, kablosuz veya elektrik hattı üzerinden taşınan bir tür ağ kullanır. Bunlar genellikle gerçek akıllı sayaçlardır, ancak tüm yetenekleri henüz tam olarak kullanılmamaktadır. Uzaktan okunan sayaçlar ile gerçek akıllı sayaçlar arasında bir miktar örtüşme vardır.

- Akıllı sayaçlar: En az bir mikroişlemci içermeleri nedeniyle "akıllı" olarak adlandırılırlar. Akıllı sayaçlar 15 yıldan fazla bir süredir HV veya büyük ölçekli enerji müşterileri (tesisler, büyük şirketler) tarafından kullanılıyordu, ancak son zamanlarda haneler ve küçük işletmeler de dahil olmak üzere toplu uygulamalar için mevcuttu. Bununla birlikte, esas olarak akıllı sayaçların maliyeti nedeniyle, ancak aynı zamanda hem sayaç ağını hem de sayaçlardan gelen büyük miktarda veriyi yönetebilecek ve aynı zamanda

cihazlar için cihazlar sağlayabilecek altyapı eksikliğinden dolayı, çeşitli son kullanıcı uygulamaları için hala oldukça yavaş gelişen bir alandır. Akıllı sayaçlar genellikle Gelişmiş Ölçüm Altyapısı (GÖA) olarak adlandırılır. Çeşitli kaynaklar, bir akıllı sayacı her iki yönde de iletişim kurabilen biri olarak tanımlar, yani, sayaç bilgileri hizmete iletebilir ve yardımcı program da sayaca bilgi gönderebilir. Bunların hepsi, herhangi bir personelin sayacı gerçekten ziyaret etmesine gerek kalmadan gerçekleşir. Yardımcı programa gönderilen bilgiler, ayrıntılı kullanım bilgilerini, voltaj izleme verilerini ve çeşitli alarmları içerebilir.

Sayaca gönderilen bilgiler, fiyatlandırma bilgileri, ön ödeme bilgileri, bağlantı kesme/yeniden bağlama talimatı, alarm/yük atma talimatı, sayacın programlanması, yazılım sayacının yükseltilmesi, tarih ve saat vb. İçerebilir. Bazı akıllı sayaçlar henüz tam olarak kullanılmayabilir. Bir alana kademeli olarak kurulabilirler veya yardımcı program şu anda bazı özellikleri yalnızca teknolojiyle ilgili deneyim elde etmek için kullanıyor. Uygulamada, daha sade modellerle çok fazla örtüşme vardır ve aynı sayaç gelecekte aniden daha fazlasını yapabilir. Akıllı sayaçların, basit ölçüm cihazlarından daha sık veri iletmesi muhtemeldir. Bazı modeller birkaç saniyede bir kadar sıklıkta iletim yapabilir. Bu örnekleme oranıyla tüketim ölçümleri göndermek, yardımcı kuruluşun sorunları (düşük voltaj gibi) zamanında çözecek kadar hızlı bir şekilde keşfetmesine yardımcı olabilir.

1.3.2.5. Akıllı Ölçüm

Akıllı ölçüm, genellikle müşterinin yerinde bir akıllı ölçüm cihazının kullanılması ve tüketim verilerinin müşteriye geri bildiriminin okunması, işlenmesi ve geri bildiriminin düzenli olarak verilmesi sürecini ifade eder. Akıllı ölçüm sisteminin evrensel bir tanımı yoktur, ancak literatüre atıfta bulunulduğunda (Ajenikoko ve Olaomi [2014]), aşağıdaki tanım geliştirilebilir: "Akıllı sayaçlar, kullanım, faturalama hakkında bilgi sağlamak için kamu hizmetleri tarafından kullanılan elektronik ölçüm cihazlarıdır. ve elektrik, su, gaz veya ısıtma sistemlerini çalıştırmak ".

Akıllı olabilmek için akıllı ölçüm sistemlerinin aşağıdaki özelliklere sahip olması gerekir:

- Ölçüm verilerinin otomatik olarak işlenmesi, aktarılması, yönetilmesi ve kullanılması,
- Otomatik sayaç yönetimi,
- Sayaçlarla iki yönlü veri iletişim protokolü,

- Kamu hizmetlerine anlamlı ve zamanında tüketim bilgileri sağlayın ve sistemleri ve
- Enerji, su, gaz veya ısınma tasarrufu sağlayan destek hizmetleri

Bu özellikler tüketicilere, zamana dayalı fiyatlandırma ve kullanım süresi fiyatları ve ayrıntılı enerji kullanımı, maliyet ve faturalama gibi zamana dayalı fiyatlandırma seçenekleri konusunda yardımcı olur. Bu özellikler aynı zamanda kamu hizmetlerinin altyapılarını ve hat kayıplarını daha iyi yönetmelerini sağlar. Genellikle, elektriğin kaydı için akıllı bir ölçüm sistemi düşünülür, ancak aynı zamanda su, gaz ve ısıtma kullanım kaydı da bir olasılıktır. Akıllı Şebekeler, elektrik, su, gaz veya ısıtma üretiminin ve dağıtımının verimliliğini, güvenilirliğini ve sürdürülebilirliğini iyileştirmek için tedarikçilerin, araçların ve son kullanıcıların davranışları hakkındaki bilgilere dayalı olarak çalışmak için Bilgi ve İletişim Teknolojilerini (BİT) kullanır. Genel olarak, akıllı bir ölçüm mimarisi üç unsurdan oluşur: (i) bir ölçüm ünitesi, (ii) bir iletişim ünitesi ve (iii) veri tabanı olan bir merkezi sistem. Ölçüm birimi, müşterinin hizmet kullanımını izler, kullanım ve tahminler hakkında bilgi sağlar ve faturalandırmayı işler. İletişim birimi, yardımcı program tedarikçisi ile iki yönlü iletişim sağlar. Merkezi sistem tedarikçinin sahasındadır ve kamu hizmeti tedarikini başlatma/kapatma, müşteri ilişkileri amacıyla verileri işleme ve yasal gerekliliklere göre verileri arşivleme özelliğine sahiptir.

Akıllı sayaçlardan 1990'ların başlarında zaten literatürde bahsedilmiştir, (De Almeida ve Vine [1994] ve Koponen ve diğerleri [1996]). 1993'te katı hal ölçerler hala oldukça yeniydi ve elektromekanik sayaçlardan 10 - 20 kat daha pahalıydı. Bu nedenle, kullanımları esas olarak büyük müşterilerle sınırlıydı. Uzaktan okunabilen akıllı sayaçların sayısı arttığında, sayaç okuma ve veri yönetimi için daha iyi sistemler geliştirmek gerekli hale geldi. 1990'ların ortalarındaki en son teknolojinin bir örneği Saari et al. [1996]. Bu tür sistemlerle, ölçülen veriler diğer sistemler tarafından kullanılabilir hale gelmeye başladı, ancak bu diğer sistemlerin çoğu bu tür verileri iyi bir şekilde kullanmaya henüz hazır değildi. Bu diğer sistemler, dağıtım ağı otomasyonu ve yönetimi için sistemlerin yanı sıra müşteri bilgi sistemlerini içerir. Benzer şekilde, faturalama sayaçlarından alınan zamanında tüketim verileri nadiren enerji yönetimi ve enerji otomasyonunu sonlandırmak için veya enerji tasarrufu potansiyelinin düzenli olarak değerlendirilmesi için kullanıldı. Statik sayaçlar elektromekanik sayaçların yerini aldığı için günümüzde akıllı ölçüm sistemleri geliyor. Dağıtılmış veri işleme ve depolama kapasitesi, seri üretilen sayaçlarda çok ucuz hale geldi ve bu faktörler artık küçük müşteri sayaçlarına akıllılık ekleme olasılıklarını sınırlamıyor.

1.3.2.6. Cihaz Yük İzleme

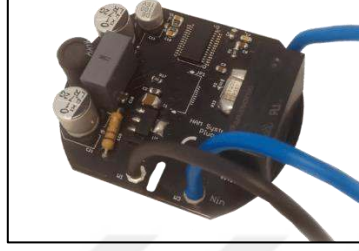
Enerji tüketimini en aza indirmek için bazı teknik önlemleri planlamadan önce, bir tesisin güç tüketimini izlemek gerektiğinden, enerji izleme, enerji yönetiminin önemli yönlerinden biridir. Cihaz Yükü izleme, gerçek yaşamdaki yük davranışını gerçekten anlamak için araçlar sağlar. Bu anlayış, simülasyon çalışmalarında yük davranışını temsil edecek yük modelleri geliştirmeye yardımcı olur. Cihaz yük izleme, yük modeli doğrulaması, yük bileşimi çalışmaları ve yük belirsizliği analizi için gereken ölçülü verileri sağlar. MOYİ ve Müdahaleci Yük İzleme (MYİ) olmak üzere iki ana yük izleme yaklaşımı vardır.

1.3.2.7. Müdahaleci Olmayan Yük İzleme

MOYİ, yalnızca tüm evi kapsayan bir güç ölçer okuması verildiğinde, bireysel cihazlar tarafından tüketilen enerjiyi tahmin etme sürecidir. Diğer bir deyişle, toplam kurulum güç ölçerden (tahmini) kalemlere ayrılmış bir enerji faturası üretir. MOYİ teknolojisine sahip elektrik sayaçları, çeşitli kurulumlarda elektrik gücünün belirli kullanımlarını araştırmak için kamu hizmeti şirketleri tarafından kullanılır. MOYİ, her kurulum cihazına ayrı izleme cihazı takmaya düşük maliyetli bir alternatif olarak kabul edilir. Bununla birlikte, gizlilikle ilgili endişeler ortaya çıkarır. Elektrikli cihazların durumunu çevrimiçi olarak izleyebilir ve müşterilerin enerji kullanım analizleri ve elektrik yönetimi yapmaları için temel olan ayrıntılı güç tüketim verilerini sağlayabilir. Müdahaleci yaklaşımlarla karşılaştırıldığında, müdahaleci olmayan düşük maliyet, kolay kurulum ve gelecek vaat eden ölçeklenebilir ticarileştirme potansiyellerine sahiptir. Bununla birlikte, MOYİ'nin tek bir noktadan ölçülen toplam enerji tüketimini ayrı elektrikli cihaz enerji tüketimine ayırmak için etkili ve verimli çözümler bulması gerekiyor.

Bireysel elektrikli cihazların enerji tüketimi, ayrıştırılmış verilerden belirlenebilir. Bu, standart müdahaleci enerji izleme türünde cihaz başına bir sensörün konuşlandırılmasının tersidir. MOYİ, masrafları düşürdüğü için değerlidir çünkü çoklu sensör konfigürasyonları ve müdahaleci yük izleme ile bağlantılı kurulum karmaşıklığı önlenir. MOYİ yöntemi genellikle finansal ve işlevsel nedenlerden dolayı tercih edilir. Sonuç olarak, akademik araştırmacılar ile birlikte kurulan ve yeni kurulan şirketler, gerçekçi bir ortam için uygulanabilir bir çözüm haline getirmek için dikkatlerini MOYİ tabanlı yaklaşımların

iyileştirilmesine odakladılar. Birçok sensörün konuşlandırılması, MOYİ'yi maliyet temelinde müdahaleci yönteme göre açıkça tercih ediyor. Öte yandan, müdahaleci yöntemlerin en önemli yararı, her bir cihazın güç tüketimini tahmin etmedeki optimum hassasiyettir. Bu durum biraz farklıdır, çünkü HAM fişin'in sunduğu yenilikçi çözüm, kolay kurulumuyla birlikte prosedürü kolaylaştırmıştır.



Şekil 4. HAM Fişi

HAM Fişi, geleneksel duvar Elektrik Prizlerinin yerini alacak şekilde tasarlanmıştır. Takılı cihazın kontrolünü ele almanıza ve güç ve enerji analizleri almanıza olanak tanır. Akıllı telefonu veya web tarayıcı kullanarak dünyanın her yerinden elektrikli cihazları kontrol edilebilir, evden çıktıktan sonra ütüyü unutup unutmadığınızı tekrar kontrol etmenize gerek yok, kWh veya para cinsinden günlük, haftalık veya aylık enerji bütçesi ayarlanabilir, çalışma zamanlarını ayarlanabilir, “If-This-Then-That” türü Kuralları kullanarak görevleri otomatikleştirilebilir, Ne kadar elektrik harcadığınızı bilin ve tasarruf edin gibi olanaklar sağlayabilir.

1.3.2.8. Müdahaleci Yük İzleme

MYİ yaklaşımlarında birden fazla sensör kullanılır ve ortama dağıtılır. Sensörler, panel seviyesinde, fiş seviyesinde veya doğrudan tek bir cihaz üzerine yerleştirilir. Veri kümesinde bulunan detay ölçeği veya seviyesi, birleştirilmiş cihazların sayısına bağlıdır. Çoğu konut binası için, kurulum karmaşıklığı, yüksek maliyet, gizlilik sorunları, teknoloji sınırlaması vb. Nedeniyle bir kurulumdaki her cihaza bir sensör takmak pratik değildir. Bu olumsuz nedenlere rağmen, MYİ teknikleri son derece güvenilir ve gerçek sonuçlar sağlar. Bu nedenle, bazı çalışmaların kapsamında HAM Automation and More P.C. sağlanan HAM Fişi (<https://hamsystems.eu/plug>) kullanılmıştır. HAM Fişi, geleneksel duvar elektrik prizlerinin yerini alacak şekilde tasarlanmıştır. HAM Fişi başlangıçta Wi-Fi'ye bağlanır ve

veri toplama prosedürü sırasında geleneksel elektrik prizlerine bağlanır. Daha sonra ev aletleri takılır. Cihaz tanıma, genellikle Makine Öğrenimi teknikleri kullanılarak gerçekleştirilir. Bu nedenle, standart adımları takip ederek makul miktarda veri toplanır ve daha sonra geliştirilen modelleri eğitmek, doğrulamak ve test etmek için kullanılır.

1.3.2.9. Cihaz Yük Tipleri ve Analizde Örneklem Hızları

MOYİ çalışmalarında 2 tür yaklaşımlar mevcuttur, 1: cihaz durumu analizinde geçici durum analizi (transient analysis) için Khz düzeyde yüksek frekanslı veriler kullanılmaktadır, 2: günlük zaman ölçeğinde Hz düzeyde düşük frekanslı güç verilere dayanan yöntemlerdir. Tez kapsamında düşük frekanslı verilere dayanan yöntemler kullanılmıştır, Hz. bazlı yöntemleri seçmekte motivasyonumuz şöyledir:

- IoT gelişmeleri ile düşük frekanslı veriler büyük olasılıkla yakın gelecekte ölçekte mevcut olan tek veri olacaktır.

- Günlük ölçekte cihaz aktivitelerinin analizi ile ev içi aktiviteler hakkında bilgi sağlayabilmesi, sağlık ve bakım uygulamalarında kullanılmak üzere ortaya çıkan bir alternatif haline gelmiştir.

- Derin öğrenme modelleri çok ağır olduğu için bulut tabanlı son kullanıcı/sunucu sistemlerin yapısına daha uygundur. Şöyle ki sunucu tarafta güçlü bir GPU bulunur ve son kullanıcılardan gelen verileri analiz ederek sonuçları web ve mobil uygulamalarda görselleştirir.

Genel olarak konuşursak, döngü yönü, kapalı alan ve kendi kendine kesişme sayısı gibi belirli özellikleri çıkarmanın mümkün olduğu durumlarda voltaj-akım yörüngesi, farklı çalışma prensipleri olan, dirençli veya endüktif olan ve dalga şekli özellikleriyle toplanan cihazlar için benzersiz özellikler sunar. Daha yeni uygulamalar, voltaj-akım yörüngesinden çıkarılan ek özellikleri kullanmıştır. Dalga biçimlerinin şeklini içeren diğer özellikler güç $p(t)$ 'den ve Ani Admittans Dalga biçiminden elde edilebilir. Toplam yükten hesaplanan yukarıda açıklanan özellikleri kullanarak, buradaki amaç, belirli bir zamanda çalışan cihazları tanımlamaktır. Bu, genellikle dört cihaz modeli dikkate alındığından, o kadar basit olmayan bir optimizasyon veya sınıflandırma problemi olarak formüle edilebilir:

- Tip I—Açma/kapama cihazları: ampuller ve ekmek kızartma makineleri gibi evdeki çoğu cihaz;

- Tip II—Sonlu Durumlu Makineler (FSM): Bu kategorideki cihazlar, tipik olarak periyodik bir şekilde durumları gösterir. Örnekler, yıkayıcı/kurutucular, buzdolapları ve benzerleridir;

- Tip III—Sürekli Değişen Cihazlar: Bu cihazların gücü zaman içinde değişir, ancak periyodik olarak değişmez. Örnekler, dimmerler ve araçlardır.

- Tip IV—Kalıcı Tüketici Cihazları: Bunlar, alarmlar ve harici güç kaynakları gibi sabit güce sahip ancak 24 saat çalışan cihazlardır.

Bu şekilde, tip II cihazlar için tanımlama sadece hangi cihazların aktif olduğuna değil, aynı zamanda durumlarına da çevrilir. Ek olarak, bazı cihazlar çoğaltılabilir (örneğin, bir evde iki buzdolabı olabilir) ve benzer yük imzalarını kullanarak çoğaltılan her cihazın çalışmasını/durumunu belirlemek gerekebilir. MOYİ'nin bu son adımı için sayısız yaklaşım önerildiğinden, bu bölümün amacı mevcut alternatiflerin derinlemesine bir incelemesini sağlamak değil, optimizasyon ve makine öğrenimi (denetimli ve denetimsiz) algoritmaları üzerine önemli çalışmalara işaret etmektir. yük sınıflandırması için kullanılır. Tez çalışması kapsamında verileri daha karmaşık, endüktif yük profillerine sahip ve analizi zor olan buzdolabı, çamaşır makinesi ve TV gibi cihazlar seçilmiştir.

1.3.2.10. Veri Ön-İşleme ve Eşikleme Uygulamaları

Saniyelik akıllı sayaç analizi aşağıdaki aşamalardan oluşur. Verilerin ön işlenmesi, AÇMA/KAPAMA (ON/OFF) olaylarından eleman üretimi ve Eğitim veritabanından cihaz bazında filtre parametresi oluşturma (Hem akıllı sayaç hem de cihaz verileri).

Ön işleme: Ön işleme aşamasında, veri ani yükselmeleri ve kesintileri gibi sinyal gürültüsü işlenir ve giderilir. Eleman algılamaya müdahale ettikleri için sivri uçların çıkarılması önemlidir. Sivri uçlar bir medyan filtresi kullanılarak çıkarılır. Sunucudan alınan verilerde, verilerdeki kesinti önceden tanımlanmış bir eşiğin üzerindeyse, dönem boyunca cihaz tanımlaması dikkate alınmaz.

AÇMA/KAPAMA olaylarından eleman üretimi: Güç seviyesindeki değişiklik 35 W'ın üzerinde olduğunda olay kaydedilir. Bu minimum eşik, olay karmaşıklığını azaltmak ve küçük dalgalanmaların hatasını azaltmak için seçilir. Bazı akıllı sayaç verilerinde görülen düşük güç dalgalanması gösterilmektedir.

Öge oluşturma: Zaman penceresi içinde (maksimum uygulama süresine dayalı olarak deneysel olarak dört saate ayarlanmıştır), ardışık olumlu olayların bir listesi sıralanır ve son olumlu olaydan sonra gelen ardışık olumsuz olayların bir listesi sıralanır. Her seferinde bir negatif olay alınır ve ardından pozitif olay listesindeki bir pozitif olayla eşleştirilir. Eşleşen olaylar kaldırılır ve süreç yinelemeli olarak takip edilir.

Cihaz imza parametresi hesaplaması: Öğeler oluşturulduktan sonraki adım, özelliklerini veya imzasını oluşturmaktır. Özellikler, cihaz durumları içindeki güç seviyeleri, süre, oluşum sayısı ve boşlukları dikkate almak için değerlendirilir.

- Penceredeki AÇIK ve KAPALI süresinin ortalama ve standart sapması.
- Pencere içindeki oluşum sayısı.
- Güç seviyesinin ortalama ve standart sapması.
- Çeşitli cihaz elektrik elemanları arasındaki zaman farkı.

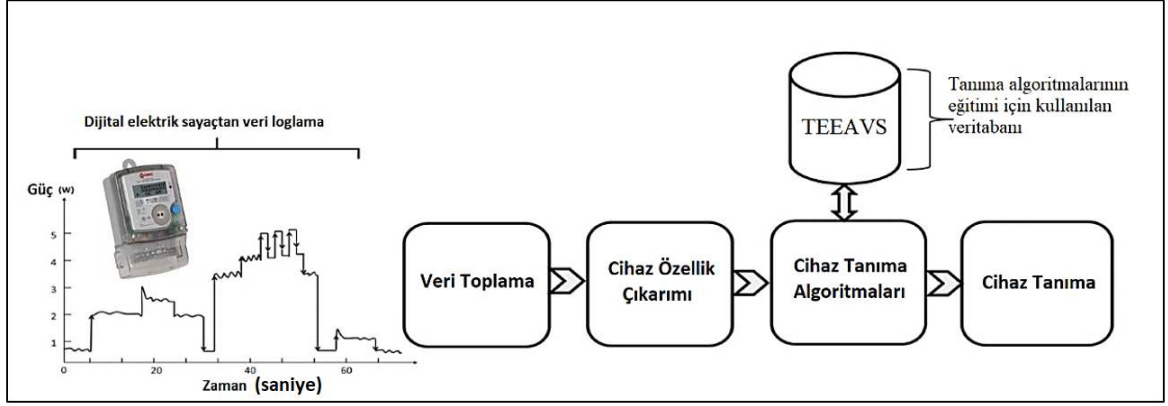
Daha önce belirtildiği gibi, güç düzeyine dayalı kümeleme, kabaca çok durumlu bir aygıtın çeşitli durumlarına veya öğelerine karşılık gelir ve çeşitli durumlar arasındaki zamansal ilişkiyi düşünmemize yardımcı olur.

2. YAPILAN ÇALIŞMALAR

2.1. Türkiye Elektrikli Ev Aletleri Veri Seti

Bu tez üzerinde iki yönüyle Akademik ve Pratik olacak şekilde çalışılmıştır. Akademik çalışmalar bölüm 2.5.'te açıklanacaktır, ayrıca çalışmamızda bilimsel makalelerde ortak kriter (benchmark) olarak kullanılan REDD veri kümesi kullanıldı ve detayları bölüm 2.5.3.'te anlatılacaktır. Pratik yönü için bir yerli veri kümesi oluşturması düşünülmüştür. Çalışmalarda evlerde çeşitli cihazlardan toplanan elektrik kullanım bilgilerini içeren ve enerji ayrıştırma (Disaggregation) araştırmalarını ilerletmeyi amaçlayan Türkiye Elektrikli Ev Aletleri Veri Seti (TEEAVS) adlı bir veri kümesi oluşturmaya çalışılmıştır. Ayrıca enerji ayrıştırma konusunda veri toplamadaki tasarım seçimleri ve nasıl etkilediklerini tartışarak veri toplama için donanım ve yazılım kurulumları açıklanmıştır.

Tez kapsamında önerilen MOYİ algoritması, evlerden toplanan veri kümeleri üzerinde metriklerle dayalı olarak incelenmiş ve bir hedef cihaz için eğitilmiştir. Dahası, önerilen yöntemin genellenebilirliği, eğitim sürecinde görülmeyen cihazlar için incelenmiştir, çünkü, sonunda, MOYİ bir ölçekte çalışan bir ortamda kullanıldığında amacımızın ayrıştırmak olduğu evler için kullanılan cihazlara dair gerçek referans değer (ground-truth) veriler çok nadirdir. Bu nedenle, MOYİ yöntemlerinin görünmeyen evlere genelleme yapabilmesi önemlidir. Eğitimden sonra, tasarlanan enerji ayrıştırma modelinin, her evden gerçek referans değer cihazlarının verilerini gerektirmemesi dikkat çekicidir, uç kullanıcıların sadece toplu veri edinmesi gerekir. Bunun nedeni, cihazın görünmeyen örneklerini genelleme gücüne sahip olacak şekilde her yapay derin sinir ağı modelinin hedef cihazının özelliğini eğitmesidir. Aynı şekilde, görüntülerin sınıflandırılmasını gerçekleştirmek için eğitilmiş modeller, kediler, köpekler vb. Gibi birçok kategori örneğinde eğitilir ve her kategorideki görünmeyen örnekleri için genelleştirilir. MOYİ'nin bir teknik geçmişi olarak, temel bir MOYİ çerçevesi üç önemli aşamadan oluşur: veri toplama (DAQ), özellik çıkarma ve örüntü tanıma. Şekil 5, bir MOYİ platformu için bu temel adımları açıklamaktadır.



Şekil 5. MOYİ yönteminin işlem akışının şeması

Öncelikle, enerji bilgisi bir veri kayıt cihazı veya veri toplama cihazı kullanılarak elektrik girişlerinden toplanır. Ardından, toplanan veriler özellik çıkarma algoritmasına göre analiz edilir. Bu evrenin amacı, orijinal veri setinin boyutsallığını azaltmak ve ham sinyallerin dalga formlarını pürüzsüz hale getirmektir. Makine öğrenmesi algoritması nihayet elde edilen bu özellikleri veritabanındaki çeşitli cihaz modelleriyle karşılaştırır. MOYİ, tek tek cihazların toplam tüketimini dikkate alarak enerji tüketimini tahmin eder, bu noktada, araştırmanın Türkiye elektrik şebekeleri için katkıları, derin sinir ağları hakkında yeni bilgiler sağlayan ve daha etkili bir teknik önerisi sunan son araştırmaların bir incelemesidir. Bu şekilde, etkili enerji ayrıştırmasına uç bir öğrenme makinesi yaklaşımı sunulur ki hanehalkının sadece tek bir merkezi noktasından izlenebilir, ölçüm cihazları için yüksek doğruluk ve düşük maliyet gibi çeşitli avantajlar sağlar.

2.2. Nesnelerin İnterneti ile Canlı Veri Toplama Sistemi ve Kurulumu

Bu tezin pratik çalışmaları yönünde Nesnelerin İnterneti (IoT) kullanarak bir veri toplama mimarisi tasarlanmıştır. Sistemin yazılım kısmında Node-Red kullanıldı. Node-Red, NodeJS 'in üzerinde inşa edilen ve onun olay güdümlü (event-driven), bloklamayan (non-blocking) model anlayışının tüm avantajlarını kullanan, ağ tarayıcıları üzerinde çalışan Akış tabanlı (flow-based), Uygulamaların Node'ların sürükle-bırak (drag-n-drop) yöntemiyle oluşturulduğu bir prototip geliştirme ortamıdır.

Kurulum için öncelikle bilgisayarınızda NodeJS'in kurulu olması gerekiyor. Değilse nodejs.org adresi üzerinden önerilen bir versiyonu kurulmalıdır. Daha sonra komut satırı (shell, bash) terminal'i açıp `node -v` komutu ile NodeJS'in kurulu olduğunu test ederek

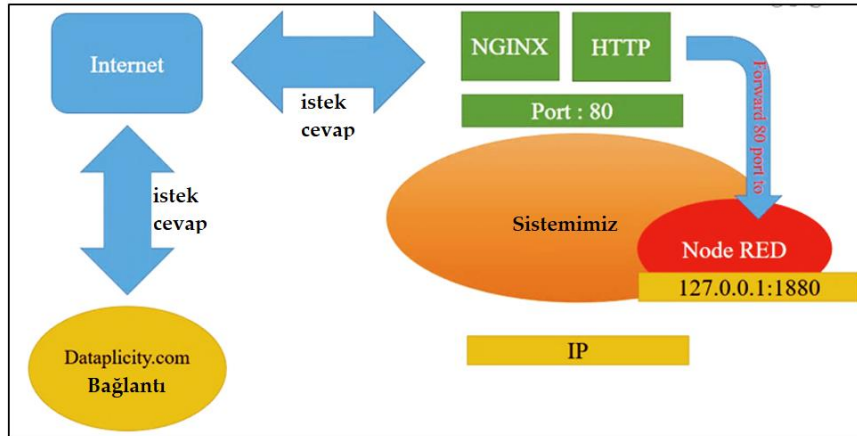
versiyon kontrolünü yapılmıştır. npm -v komutu ile Node Paket Yöneticisinin (node package manager) kurulu olduğunu kontrol edilmiştir. Eğer sonuç alınamıyorsa NodeJS'yi işletim sisteminin ortam değişkenlerine eklenilmesi gerekebilir. Kurulumu tekrarlamayı ve hatayı çözmeyi deneyelim. Ardından yine komut satırı üzerinden

Linux için => sudo npm install -g --unsafe-perm node-red

Windows için => npm install -g --unsafe-perm node-red

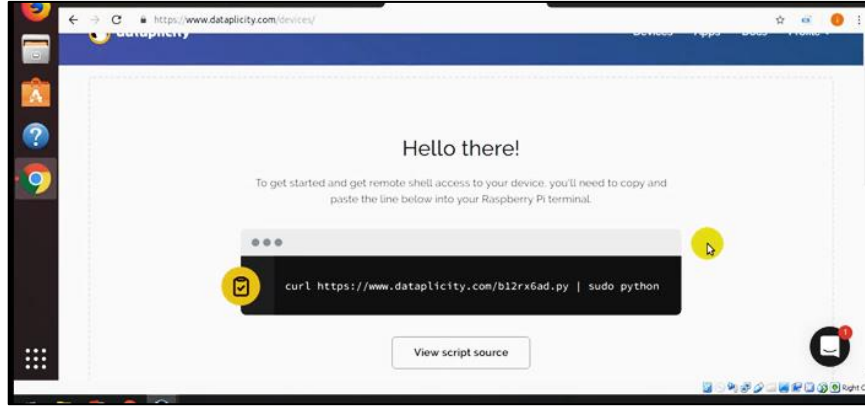
Komutları çalıştırılmıştır. Kurulum tamamlandığında Node-Red'in çalışması için gerekli dosyalar yüklenmiş olacaktır. Aynı komutları kullanarak Node-Red güncellenmiştir. Daha sonra komut satırı üzerinden node-red yazılarak Node-Red'in <host>:1880 portunda çalışması sağlanmıştır. Host => localhost, 127.0.0.1 veya genel (public) ip olabilir. Bir internet tarayıcısı kullanarak http://127.0.0.1:1880 adresine bağlanılmıştır. Böylece sunucuya bir Node-Red uygulaması kurulmuştur. Sonraki yazı dizisinde veri toplama arayüzünü keşfedip neler yapabileceğine bakılmıştır.

Uzaktan herhangi bir internet tarayıcı üzerinden kurulan sisteme erişmek için aşağıdaki Port Yönlendirme yapısı kurulmuştur.



Şekil 6. Node-Red Port Yönlendirme yapısı

Yukarıda gösterildiği gibi “Dataplicity” Port Yönlendirme işlemi yapılarak, HTTP isteği ile her bağlantı yapıldığında bize atanan IP adresini gönderen <https://www.dataplicity.com/> kullanılmıştır:



Şekil 7. Dataplicity Port Yönlendirme platformu

Yukarıda gösterilen sitede mail adresimiz girilmiştir, sora verilen kod alınıp linux terminal’de çalıştırılmıştır, bu sayede linux terminal’i ağ tarayıcıda açılmasını sağlar. Açılan sayfada (other linux device) tıklayarak linux command ortamına girilmiştir. Açılan sayfada gösterildiği gibi tuşu aktif edilerek (yeşil olması lazım) ve mavi bağlantıya tıklayarak bu sayfa kurulu olan node-red sunucusuna bir istek atıyor (request), fakat daha ağ sunucusu kurulmadığından dolayı yanıt dönmeyecektir. Şimdi Node-Red uygulamasına HTTP ile internetten (uzaktan web ya da mobil tarayıcıdan) erişmek için Nginx adında bir ağ sunucusu kurulumu yapılarak ubuntu terminal açılarak aşağıdaki komutlar kullanılmıştır:

```
sudo apt install nginx
```

Nginx bir sunucu olduğu için süper kullanıcı (super user) olması gerekmektedir, bu yüzden (sudo bash) terminalde aşağıdaki sırasıyla komutlar:

```
(base) root@vahit:# cd /etc/nginx
```

```
# cd sites-enabled/
```

sonra LS komutu ile klasör içeriğine bakılmıştır, içerisinde varsayılan (default) dosyası bulunmaktadır, kullanılmadığı için aşağıdaki komut ile silinmiştir:

```
rm default
```

sonra nano editör ile bir dosya açılarak, “KTÜSmartGridAnalytics” ismi verilmiştir, herhangi bir isim verilebilir: (base) root@vahid:# nano KTÜSmartGridAnalytics

```
server {
    listen 80;
    server_name blowzier-tapir-7926.dataplicity.io;
    access_log /var/log/nginx/access.log;
    location / {
```

```

proxy_pass http://localhost:1880;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";}}

```

Sonra terminalden aşağıdaki komutu kullanılarak nginx sunucusu yeniden başlatılmıştır:

```
systemctl restart nginx
```

bu sayede kullanılan bağlantı ile dünyanın her bir noktasından internet tarayıcı kullanılarak uygulamaya giriş yapılabilir, fakat kullanıcı ile şifre ayarlanması gerekmektedir, ilk olarak node-red uygulaması durdurulması lazım:

```
sudo systemctl stop node-red
```

şimdi hash fonkyonu ile şifre üreteceğiz:

```
vahit@vahit:cd ~/.node-red && ls
```

```
lib node_modules package.json settings.js
```

şimdi nano ile settings.js açıyoruz:

```
vahit@vahid: nano settings.js
```

```
adminAuth: {
```

```
  type: "credentials",
```

```
  users: [{
```

```
    username: "vahit",
```

```
    password:
```

```
"$2b$08$IXIgp06EFdR3dzB6IBlsAeablAjrZpy1rK.SgcCo8nGRrkDGEyvjm",
```

```
    permissions:"*" ]}],
```

```
httpNodeAuth:{ user:"vahit",pass:"$2a$08$zZWtXTja0fB1pzD4sHCMYOCMYz2Z6dNbM6tl8sJogE
NOMcxWV9DN."},
```

yukarıda kullanılmak üzere linux “hash” üreteç komutu ile bir şifre üretilmiştir:

```
vahit@vahit: node-red-admin hash-pw
```

```
pass: 1234vahit
```

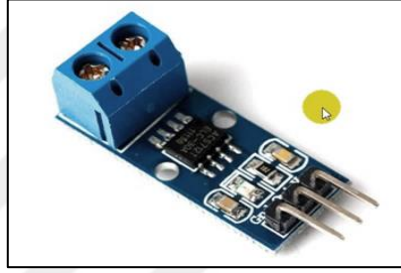
uygulamaya şifre koyulduktan sonra tekrar node-red uygulaması başlatılmıştır:

```
vahid@vahid: sudo systemctl start node-red
```

2.3. Nesnelerin İnterneti Veri Toplama Donanımları ve Kurulumu

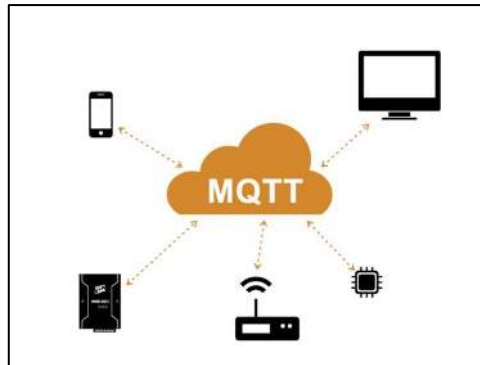
IoT yapısı izleme ve denetim için algılayıcılar (Sensors) ve eyleyiciler (Actuators) bileşenlerden oluşur, bu çalışmada akıllı elektrik şebekeleri (Smart Power Grid) uygulamalarına yönelik akım sensörleri ile elektrikli ev aletlerinin tükettiği güç ölçümü yapılarak bu verilerin kaydı ile bir Veri Kümesi oluşturularak Yapay Zeka aracılığıyla cihaz sınıflandırılması için kullanılmıştır. Kullanılan malzemeler aşağıda verilmiştir:

- Nodemcu ESP8266 mikrodenetleyici kartı
- Modem (WiFi)
- Ubuntu 18.04 LTS (linux işletim sistemi)
- ACS-712 (30A modül) akım sensörü



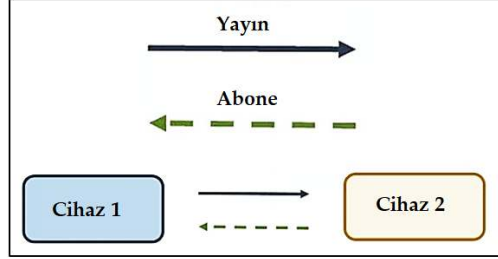
Şekil 8. ACS-712 Akım Sensörü

MQTT (Message Queuing Telemetry Aktarım) protokolü, internette yaygın olarak kullanılan makinalar arası (M2M) mesaj tabanlı bir protokoldür.



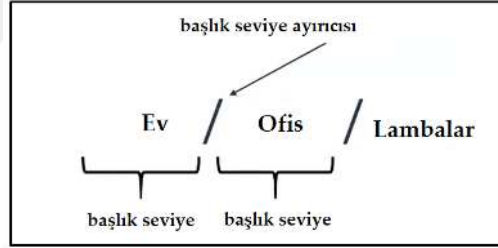
Şekil 9. MQTT protokolü

Bu haberleşme trafiğini kontrol eden yöneticiye arabulucu (BROKER), mesaj yayınına (PUBLISH) ve bu mesaj yayınına abone olanlara (SUBSCRIBE) denmektedir, MQTT de asenkron bir haberleşme kullanılmaktadır.



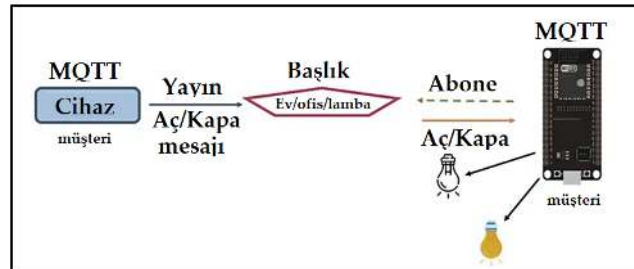
Şekil 10. MQTT haberleşme akışı

Aşağıda gösterildiği gibi mesajların gönderildiği yayınlara başlık (topic) adı verilir yani bir mesaj aşağıdaki adrese yayın edilir ve abone ise bu adrese bakarak (subscribe yaparak) mesajı alabilir.



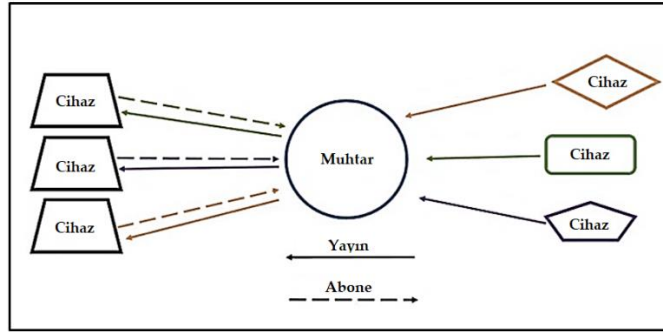
Şekil 11. MQTT haberleşme adresi

Daha iyi bir görsel anlatım adına IoT MQTT yapısı aşağıdaki gibidir:



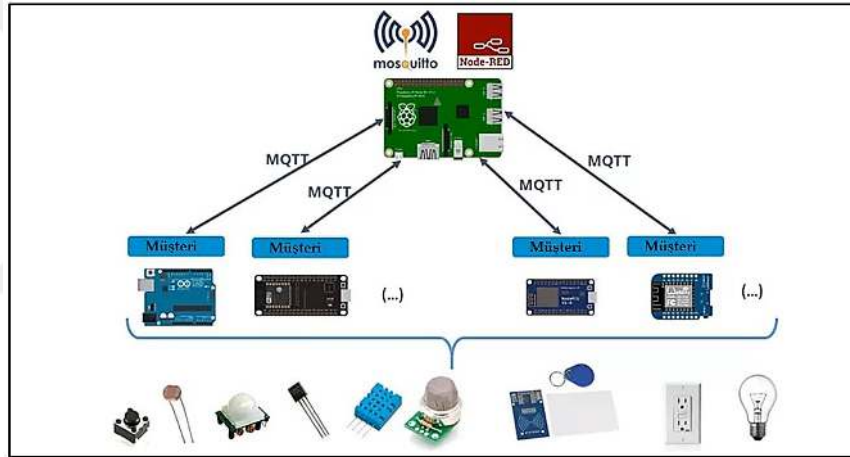
Şekil 12. Nesnelerin İnterneti MQTT görsel anlatımı

Bu haberleşme trafiğini kontrol eden yöneticiye BROKER (arabulucu) denilmektedir.



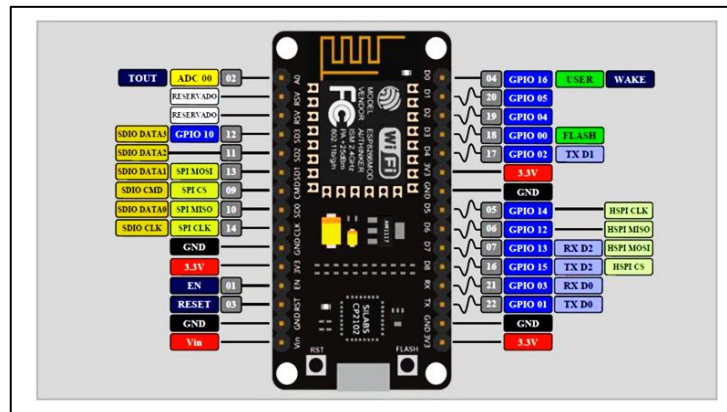
Şekil 13. MQTT arabulucu

Bir IoT sisteminin genel yapısı aşağıdaki gibidir:



Şekil 14. Nesnelerin İnterneti mimarisinin genel yapısı

NodeMCU ESP8266 pin haritası:



Şekil 15. NodeMCU ESP8266 pin haritası

Bazı işlemcilerin çalışma gerilimleri ve sensörlerin çıkış gerilimleri farklı olmaktadır, bazıları NodeMCU gibi 3.3 Volt ile çalışmaktadır, bazıları 5volt, bazı sensörler 3.3v bazıları 5volt, bundan dolayı gerilim bölücülerin kullanımı önemlidir, NodeMCU 3.3 volt ile çalışır ve ACS-712 akım ölçer 30A modülü 5volt ile çalışır, 5V çıkışını NodeMCU'ya bağlantısı yapılmadan önce 3.3v aralığında olacak şekilde ayarlamak gerekir. $V_{in} = 5v$ ve $V_{out} = 3.3v$ olarak bilinen değerlerdir ve bölücünün R1 ile R2 direnç değerlerini hesaplamak gerekir. Buraya kadar Nginix sunucu ve Node-Red kurulumu yapılmıştır, NodeMCU'yu Nginix sunucu üzerinden Node-Red'e bağlanmak için "Mosquitto" adlı bir MQTT arabulucu yüklenmiştir, aşağıdaki terminal komutları kullanılmıştır:

```
sudo apt install mosquitto
```

```
mosquitto -v
```

daha sonra mosquitto yöneticisi portu 1883 olarak ağımda (network) bulunmaktadır, cihazlar (NodeMCU gibi) arabulucuya mesaj yollamak için sunucu ağıma erişebilmesi gerekir, bu yüzden bir WiFi Modem kullanılarak, Modem yerel ağ kablosu aracılığı ile sunucuya bağlanmıştır, cihazlar wifi üzerinden sunucu ağına bağlanır, daha önce Node-Red portu 1880 ve mosquitto ise 1883 olarak ayarlanmıştı. Ports:

```
mosquitto 1883
```

```
Node-Red 1880
```

Tarayıcıya 192.168.1.1 yazılarak modem ayarlarına girilmiştir, aşağıda gösterildiği gibi sunucu IP'si bulunmuştur. Buraya kadar Node-Red'e yerel "host" olarak bağlanılıyordu, Model üzerinden bağlantı doğrulanmıştır, bu çalışmadaki IP 192.168.1.61 kullanılarak 1880 olan Node-Red'e bağlanılmıştır, tarayıcıda 192.168.1.61:1880 yazılarak node-red ortamına girilebilmelidir, modem'e her bağlanıldığında IP'nin değişmemesi için IP ayarının statik yapılması gerekir.

```
static IP: 192.168.1.10
```

```
192.168.1.10:1880
```

NodeMCU ile WiFi Modem'e bağlanılarak, MQTT arabulucuya bağlanılabilecektir.



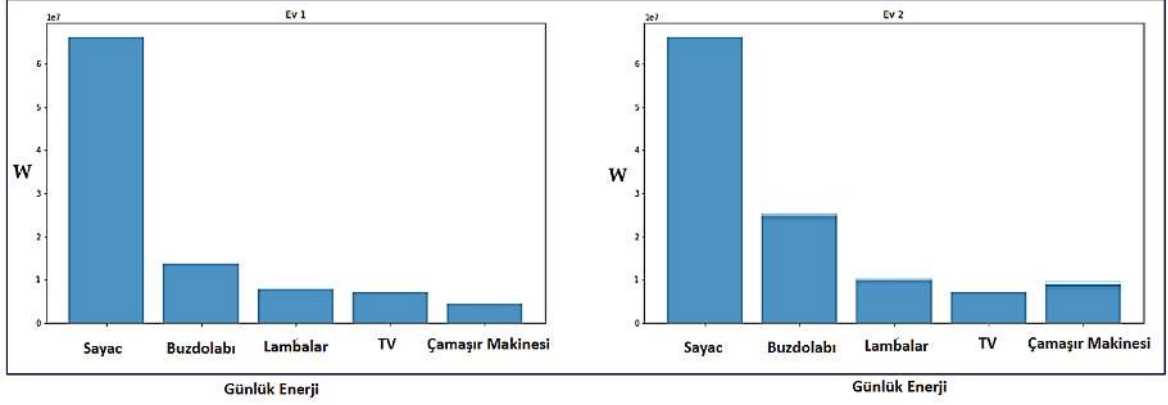
Şekil 16. KTÜ Gerçek Zamanlı Elektrik Güç İzleme ve Veri Toplama Sistemi

2.4. Veri Çerçevesi Oluşturması

Veri çerçeveleri (Data Frame) çok boyutlu veri yapılarıdır, örnek olarak verilebilecek yapı sql veya excel tabloları verilebilir. Bunlar gibi satır ve sütunlardan oluşurlar. Veri Çerçevesi yapıları veri analizi işlemlerinde oldukça faydalı olmaktadır. Bu çalışmada toplanan TEEAVS verileri için aşağıda gösterildiği gibi bir Veri Çerçevesi oluşturulmuştur. Ev 1 olarak 4 gün data içerir yani 2020-12-15 den 2020-12-18 e kadar yani ['2020-12-15', '2020-12-16', '2020-12-17', '2020-12-18']:

Tablo 1. Veri Kümesindeki Ev 1 için veri çerçevesi gösterimi

Zaman Damgası	Sayacı (w)	TV (W)	Dolap (W)	Lamba (W)	Çamaşır M (W)
2020-12-18 18:11:53	365.0	97.0	17.0	88.0	26.0
2020-12-18 18:11:55	370.0	92.0	29.0	92.0	20.0
2020-12-18 18:11:57	365.0	100.0	24.0	82.0	21.0
2020-12-18 18:11:59	371.0	93.0	17.0	87.0	20.0
2020-12-18 18:12:01	364.0	91.0	24.0	81.0	26.0
2020-12-18 18:12:03	365.0	104.0	24.0	91.0	18.0
2020-12-18 18:12:05	366.0	94.0	24.0	88.0	24.0



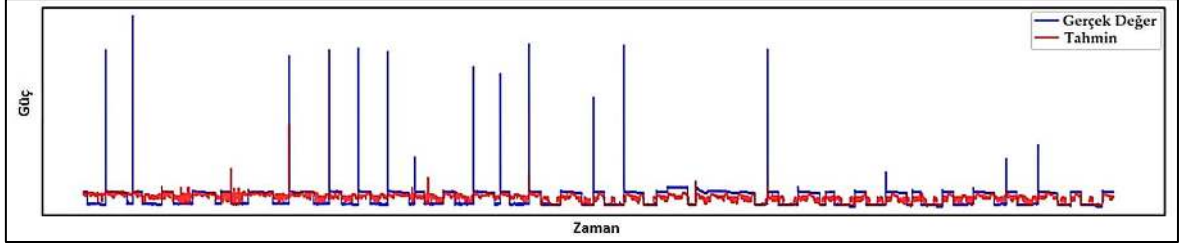
Şekil 17. Cihazların Toplam Günlük Enerji Tüketimi Grafikleri

Burada çalışmanın pratik kısmında toplanan veriler ile oluşturulan dataframe'in doğruluğunu test etmek amacıyla bir derin öğrenme modeli eğitilmiştir. Tezin akademik çalışmaları ise Yapılan Çalışmalar kısmında anlatılacaktır, çalışmanın yapay zeka, LSTM mimarisi, optimizasyon algoritmaları, benchmark dataset ve doğruluk metrikleri vs bilimsel konulardan oluşmaktadır. Ev 1'den yapay zeka modelin eğitimi için seçilen buzdolabı verileri, 4 kullanım günü verisi ve 1 günlük test verilerini içerir. Tablo 2, bir optimizer tarafından eğitilen yapay sinir ağının, ev 1'den gelen verilerle performans metriklerini göstermektedir.

Tablo 2. TEEAVS ile Eğitilen MOYİ Modelinin Metrikleri

Yöntem	MSE	Kayıp	Doğruluk	LR	İterasyon
LSTM-MOYİ ADAM	272.310	147.9198	0.5293	5e-3	223188

Bu çalışmanın pratik kısmı, enerji ayrıştırması için bir veri seti olan TEEAVS'i tanıttı. Enerji ayrıştırması, ilerlemelerin enerji verimliliği ve sürdürülebilirlik üzerinde gerçek bir etkiye sahip olabileceği algoritmik bir zorluktur. Donanım ve yazılım kurulumunu tanımladık ve ayrıştırma görevi için standart bir LSTM algoritma ile çıkan sonuçları gösterdik. Tablo 2. Ve Şekil 18'de gösterildiği gibi performans sonuçları düşük çıkmıştır. Bunun nedeni veri kümesinin az olmasıdır, derin öğrenme algoritmalarının eğitimi büyük verilere ihtiyaç duymaktadır, ayrıca yüksek sayıda evlerden çeşitli elektrik tüketimi alışkanlıklardan veri toplamak oldukça maliyetli ve zaman alıcıdır, dolayısıyla tez kapsamında yaşanan zorluklardan dolayı büyük veri seviyesinde veri kümesi toplanamamıştır.



Şekil 18. Buz dolabı gücünün gerçek ve tahmini sinyallerinin zamana göre değişimi

Bu tezin pratik kısmı, ülkemizde akıllı elektrik şebekelerin araştırmaları ve geliştirmelerini ilerletmeyi amaçlamıştır. TEEAVS'i geliştirmedeki nihai hedefimiz, veri madenciliği veya makine öğrenimi alanında çalışan araştırmacılar için kolay erişilebilir bir veri kümesi sağlamaktır. Bu nedenle, sunduğumuz deneylerde MOYİ'lerin belli bir ölçüde performans göstermesine rağmen, iyileştirme için çok yer olduğu gerçeğinin altını çiziyoruz. Ayrıca cihaz sinyallerine katı kısıtlamalar eklemek veya güç sinyalinin daha karmaşık özelliklerine bakmak, bu performansın iyileştirilmesine yardımcı olabilir. Bizi özellikle ilgilendiren, bu tür tekniklerin birden fazla evdeki farklı cihazlara nasıl genişleyebileceğidir. TEEAVS daha büyük bir kaynak olmayı amaçlasa da, yalnızca bu kadar çok evi bu kadar ayrıntılı algılamayla donatabiliriz ve geriye kalan büyük bir zorluk, bu tür yüksek doğruluk ölçümünü muazzam miktarlarda (etiketsiz) akıllı sayaç verileriyle birleştirmenin yollarını keşfetmektir. Umudumuz, TÜBİTAK gibi kurumların desteği ile TEEAVS gibi bir veri kümesinin mevcudiyetinin makine öğrenimi ve veri madenciliği topluluklarını bu sorunu çözmek için daha da motive edebilmesidir.

Yapılan çalışmalar kısmında tezin bilimsel ağırlığı ve akademik teorilerini kanıtlamak için MOYİ makalelerinde ortak kriter olarak kullanılan REDD adlı veri kümesi kullanılmıştır.

2.5. MOYİ Akademik Çalışmalarında Makine Öğrenimi

2.5.1. Derin Öğrenme ve Gradyan İniş Optimizasyon Yöntemleri

Bu bölümde Yapay Zekâ, Makine Öğrenimi ve Derin Öğrenmede kullanılan Gradyan İniş Yöntemlerinin temel kavramları ve teorileri açıklanmış ve en yaygın teknikler ile bunların arasındaki farklar kısaca anlatılmıştır. Ayrıca, bu tez kapsamında bilimsel

makalelerde ortak kriter olarak (benchmark) kullanılan Tekrarlayan Sinir Ağları (RNN) türünden Uzun Kısa Süreli Bellek (LSTM) mimarisi ve teorik arka planında sunulmuştur.

2.5.1.1. Yapay Zekâ

Yapay Zeka (YZ), bir bilgisayarı, robotu veya ürünü insan kadar akıllı düşünmeye yönelik bir yaklaşımdır. YZ, insan beyninin problemleri çözmeye çalışırken nasıl düşündüğünü, öğrendiğini, karar verdiğini ve çalıştığını inceleyen bir çalışmadır. YZ'nin amacı, insan bilgisi ile ilgili, örneğin akıl yürütme, öğrenme ve problem çözme gibi bilgisayar işlevlerini geliştirmektir. Bu çalışmanın çıktısı akıllı yazılım sistemleridir. 1940'larda dijital bilgisayarın gelişmesinden bu yana, bilgisayarların matematiksel teoremler için kanıtlar keşfetmek veya büyük bir ustalıkla satranç oynamak gibi çok karmaşık görevleri yerine getirmek üzere programlanabileceği gösterilmiştir. Yine de, bilgisayar işlem hızı ve bellek kapasitesindeki devam eden ilerlemelere rağmen, daha geniş alanlarda veya çok fazla günlük bilgi gerektiren görevlerde insan esnekliğini karşılayabilecek hiçbir program hala yoktur. Öte yandan, bazı programlar, belirli görevleri yerine getirmede insan uzmanların ve profesyonellerin performans seviyelerine ulaşmıştır. Bu nedenle, bu sınırlı anlamda YZ, tıbbi teşhis, bilgisayar arama motorları ve ses veya el yazısı tanıma gibi çeşitli uygulamalarda bulunur. YZ kapasitesini tanımlarken, genellikle zayıf veya güçlü YZ açısından kategorize edilir.

Zayıf YZ (dar YZ), gözlemlenen bir davranışı olabildiğince doğru bir şekilde yeniden üretmeyi amaçlayan bir YZ'dir. Kesinlik (precision) eğitimi almış oldukları bir görevi yerine getirebilir. Bu tür yapay zekâ sistemleri, kendi alanlarında son derece verimli hale gelebilir, ancak genelleme yeteneklerinden yoksundur. Makine Öğrenimi, örüntü tanıma, veri madenciliği veya doğal dil işleme kullanan mevcut akıllı sistemlerin çoğu, zayıf YZ örnekleridir. Zayıf YZ ile desteklenen akıllı sistemler arasında öneri sistemleri, spam filtreleri, kendi kendine giden arabalar ve endüstriyel robotlar bulunur.

Güçlü YZ genellikle gerçek bilince sahip, bir insan gibi düşünebilen ve akıl yürütebilen akıllı bir sistem olarak tanımlanır. Güçlü bir yapay zekâ, yalnızca zayıf bir yapay zeka gibi bilgileri özümsemekle kalmaz, aynı zamanda kendi işleyişini de değiştirebilir, yani genel akıllı görevleri gerçekleştirmek için YZ'yi özerk olarak yeniden programlayabilir. Bu süreçler, bilinç, duyarlılık, zekâ ve öz farkındalık dahil olmak üzere insan benzeri bilişsel yetenekler tarafından düzenlenir. Güçlü bir yapay zekâ oluşturmayı amaçlayan çabalar, tüm

beyin simülasyonlarına odaklanmıştır. Bununla birlikte, zekâ basitçe tek bir organdan kaynaklanan biyolojik bir süreç olarak açıklanamayacağından, zeki bir varlık ile çevresi arasındaki karmaşık bir etki ve etkileşim birleşimi olduğu için, bu yaklaşım eleştiriler aldı ve birbiriyle bağlantılı biyolojik yollarla bir dizi farklı yolu kapsayan süreçtir. Makine Öğrenimi, büyük ölçüde bilgisayar donanımındaki iyileştirmeler sayesinde yürüyüşüne devam etti. Şirketler ve hükümetler, dar alanlarda Makine Öğrenimi yöntemlerini başarıyla kullandı. Bilgisayar işlem gücü ve depolama yeteneklerindeki üstel ilerleme, şirketlerin büyük miktarda veriyi ilk kez depolamasına olanak tanıdı. Geçtiğimiz 15 yılda, birçok şirket Makine Öğrenimini büyük ticari avantajları için kullandı. Yapay zekâ uzmanları, tüketici davranışını anlamak için kullanıcı verilerini işlemenin yanı sıra bilgisayar görüşü, doğal dil işleme ve çok çeşitli diğer yapay zeka uygulamaları üzerinde çalışmaya devam ettiler. Makine Öğrenimi artık kullandığımız çevrimiçi hizmetlerin çoğunda yerleşiktir. Sonuç olarak, bugün, teknoloji sektörü uluslararası pazarı yönlendiriyor.

2.5.1.2. Makine Öğrenimi

Makine Öğrenimi (MÖ) , bilgisayar sistemlerinin açık yönergeler kullanmak yerine kalıplara ve çıkarımlara dayanarak belirli bir görevi etkin bir şekilde gerçekleştirmek için kullandığı algoritmaların ve istatistiksel modellerin bilimsel çalışmasıdır. MÖ, Yapay Zekanın bir alt kümesi olarak görülmektedir. Bilgisayarları kullanarak tahminler yapmaya odaklanan hesaplamalı istatistiklerle yakından ilgilidir. MÖ, tahmine dayalı analitik olarak da adlandırılır. Matematiksel optimizasyon çalışması, Makine Öğrenimi alanına yöntemler, teori ve uygulama alanları sunar. MÖ algoritmaları, görevi gerçekleştirmek üzere açıkça programlanmadan tahminlerde bulunmak veya kararlar almak için "eğitim verileri" olarak bilinen matematiksel bir örnek veri modeli oluşturur. Görevi gerçekleştirmek için belirli talimatların bir algoritmasını geliştirmenin mümkün olmadığı yerlerde e-posta filtreleme ve bilgisayarla görme gibi çok çeşitli uygulamalarda kullanılırlar. Makine Öğrenimi algoritmaları, algoritmanın istenen sonucuna göre düzenlenir, En yaygın MÖ algoritmik türleri şunlardır:

Denetimli Öğrenme: Algoritma, eğitim aşamasında makineye doğru cevaplar görünürken, girdileri istenen çıktılarına eşleyen bir işlev üretir. En yaygın Denetimli Öğrenme yöntemlerinden ikisi şunlardır:

- Sınıflandırma, bir dizi veri noktasının sınıfını tahmin etme sürecidir. Sınıflar bazen hedefler, etiketler veya kategoriler olarak adlandırılır. Sınıflandırma kestirimci modelleme, girdi değişkenlerinden (X) ayrık çıktı değişkenlerine (Y) bir eşleme fonksiyonunu (f) yaklaşıklştırma görevidir. En sık kullanılan sınıflandırma algoritmalarından bazıları Lojistik Regresyon, k-En Yakın Komşular, Destek Vektör Makineleri, Naive-Bayes ve Karar Ağaçlarıdır.

- Regresyon, girdi ve çıktı değişkenleri arasındaki ilişkiyi tahmin etme sürecidir. Unutulmaması gereken bir şey, çıktı değişkenlerinin sürekli değerli gerçek sayılar olmasıdır. Dolayısıyla, aksine çıktı sınıflarının sayısının sabit olduğu sınıflandırma, sonsuz sayıda olasılık vardır.

Denetimsiz Öğrenme: Algoritma, verilerdeki gizli kalıpları veya içsel yapıları bulur. Etiketli yanıtlar olmadan girdi verilerinden oluşan veri kümelerinden çıkarımlar yapmak için kullanılır. En yaygın Denetimsiz Öğrenme teknikleri şunlardır:

- Kümeleme: Benzer varlıkların bir arada gruplandırılması işlemidir. Bu Gizli kalıpları bulmak için keşifsel veri analizi için kullanılır veya verilerdeki gruplamalar. Makine Öğreniminde en popüler ve yaygın olarak kullanılan kümeleme algoritmaları, k-Ortalamlar ve Hiyerarşik kümelemedir.

- Boyutsallık Azaltma: Boyutluluk, boyut sayısı olarak tanımlanır. Bir veri kümesindeki özellikler. Veri kümesi içinde bu boyutlar sütunlarla temsil edilir. Böylece amaç, sağlanan veri kümesindeki sütun sayısını azaltmaktır. Çoğu durumda, bu sütunlar bağıntılıdır ve bu nedenle, gereksiz olan ve veri kümesinin gürültüsünü artıran bazı bilgiler vardır. Bilgi fazlalığı, Makine Öğrenimi modellerinin eğitimini ve performansını olumsuz etkilediğinden, boyut azaltma yöntemlerinin kullanılması önemlidir. Modellerin karmaşıklığını azaltmanın ve aynı zamanda fazla takmayı önlemenin çok kullanışlı bir yoludur. Boyut azaltmanın iki ana kategorisi, Özellik Seçimi ve Özellik Çıkarma'dır.

Yarı Denetimli Öğrenme: Hem etiketli hem de etiketsiz verileri birleştirerek Denetimli ve Denetimsiz Öğrenme paradigmaları arasında yer alır. Bu yaklaşım, zaman alıcı veya elde edilmesi pahalı olduğu için etiketli veri noktalarının az olduğu durumlarda sıklıkla kullanılır. Algoritma, Denetimli Öğrenme algoritmalarını geliştirmek için etiketlenmemiş verileri kullanır.

Takviyeli Öğrenme: Algoritma, dünyayı gözlemleyerek nasıl hareket edileceğine dair bir politika öğrenir. Her eylemin çevre üzerinde bir etkisi vardır ve çevre, öğrenme algoritmasına rehberlik eden geri bildirim sağlar. Performansını en üst düzeye çıkarmak için

makinelerin ve yazılım araçlarının belirli bir bağlamda ideal davranışı otomatik olarak belirlemesine olanak tanır. Bu tezin nihai amacı, her bir akıllı fiş cihazına belirli bir zaman noktasında hangi ev cihazının bağlı olduğunu belirlemektir. Bu amaçla problem "Denetimli" kategorisine girer ve çözmek için Sınıflandırma algoritmaları uygulanır.

2.5.1.3. Derin Öğrenme

Hem Makine Öğrenimi hem de Derin Öğrenme, Yapay Zekanın alt kümeleridir, ancak Derin Öğrenme, Makine Öğreniminin evrimindeki bir sonraki adımı temsil eder. Makine Öğreniminde, insan programcılar tarafından oluşturulan algoritmalar, verilerin ayrıştırılmasından ve öğrenilmesinden sorumludur ve daha sonra öğrendiklerine göre kararlar alırlar. Derin Öğrenme teknikleri, insan beyni gibi davranan ve makinenin bir yapıdaki verileri insanlar gibi analiz etmesine izin veren yapay bir sinir ağı aracılığıyla öğrenir. Derin Öğrenme makineleri, bir insan programcının onlara verilerle ne yapacaklarını söylemesini gerektirmez. Bu, topladıkları ve tükettikleri olağanüstü miktarda veriyle mümkün oluyor - veriler Derin Öğrenme modellerinin yakıtıdır. Derin Öğrenme, çeşitli doğrusal olmayan dönüşümleri birleştiren karmaşık mimarilerle verileri modellemeye çalışan bir dizi öğrenme yöntemidir. Derin Öğrenmenin temel tuğlaları, Derin Sinir Ağlarını oluşturmak için birleştirilen Sinir Ağlarıdır. Bu teknikler, özellikle yüz tanıma, konuşma tanıma, bilgisayarla görme, otomatik dil işleme, metin sınıflandırma (örneğin istenmeyen posta tanıma) dahil olmak üzere ses ve görüntü işleme alanlarında önemli ilerlemeler sağlamıştır. Bununla birlikte, Derin Öğrenmenin potansiyel uygulamaları çok sayıda ve çeşitlidir.

2.5.1.4. Nöron Tahminleri ve Yapay Sinir Ağları

Bu kısımda, bu konuların geri kalanına geçmeden önce bahsedilmesi gereken tüm kavramları gözden geçireceğiz, bu konu aynı zamanda bir kontrol listesi görevi görecektir. Şimdi bu alanın ana hatlarını inceleyelim. Bu kısım, makine öğreniminde bu model için nöron olarak da adlandırmayı sevdiğimiz Lojistik Regresyonun bir incelemesiyle başlayacaktır. Birden fazla adım olacak. Özellikle, nöronun tahminlerde bulunmak için nasıl kullanıldığını gözden geçireceğiz. Sonra nöronların nasıl öğrendiği hakkında konuşacağız ki bunun merkezi kısmı hata ya da kayıp fonksiyonudur (loss function). Farklı görevler için

farklı türde kayıp fonksiyonlarımız vardır. Bu elbette sınıflandırmaya odaklanacak, ancak regresyon hakkında da bilgi sahibi olmamız gerekmektedir. Öğrenmeye gelince tartışmamız gereken bir sonraki şey, kayıp fonksiyonunun nasıl kullanıldığıdır, özellikle öğrenme, model parametrelerine göre kayıp fonksiyonunu en aza indirerek gerçekleştirilir.

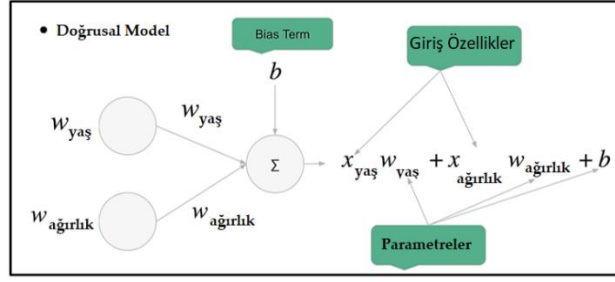
Şimdi, tipik olarak derin öğrenmede gradyan inişi yönteminden bahsediyoruz ve bu gözden geçireceğimiz öğrenme yöntemidir. Başka olasılıklar da var, ancak bunlar derin öğrenmede kullanılmaz. Daha sonra, sinir ağları için nöronlardan sinir ağlarına geçeceğiz. Aynı adımları tekrar gözden geçireceğiz. İlk olarak, nasıl tahminlerde bulunacağımızı tartışıyoruz. İkinci olarak, modelin sinir ağları için nasıl öğrendiğini tartışıyoruz. Bu işlemin geri yayılma (back-propagation) olarak adlandırıldığını hatırlıyoruz. Ayrıca, Overfitting'i önlememize yardımcı olan bir teknik olan düzenleme olarak bilinen bir kavramı tartışacağız. Bunu gözden geçirmenin amacı, bu kısımda bunu yapmanın daha gelişmiş yöntemlerini inceleyeceğiz ve daha açık hale getirmek için lojistik regresyonu gözden geçirerek başlayalım, bu konu boyunca örnek bir problem kullanalım. Asyadan girdi verilerimizin bir kişinin yaşı ve ağırlığı olduğunu varsayalım.

Hedef, o kişinin yüksek tansiyon riski altında olup olmadığıdır. Dolayısıyla, bu durumda sınıflandırma hedefimiz ikilidir. Şimdi, muhtemelen örnek için bu çok basit ama verilerimizin bir tabloda gösterilebileceğini unutmayın.

Girdi Veri (X)		Hedef (T)
Yaş	Ağırlık	Risk var mı
20	250	1
30	150	0
40	120	1
...	...	...

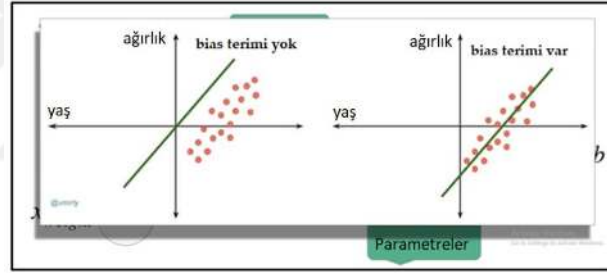
Şekil 19. Özellikler ve hedefin veri çerçevesi

Tipik olarak, girdi kümesine büyük harf X diyoruz ve hedef kümesine büyük harf Y veya T diyoruz. Lojistik regresyon çok basit bir doğrusal modeldir, yani doğrusal bir denklem kullanarak Asya yönünü birleştirir ve bunun çıktısı modelimizin tahminidir. Aşağıdada gösterildiği gibi yaş değişkenini yaş parametresi ile çarpıyoruz ve ağırlık değişkenini ağırlık parametresi ile çarpıyoruz. Özellikle derin öğrenmede bazen parametrelere ağırlıklar denilmektedir.



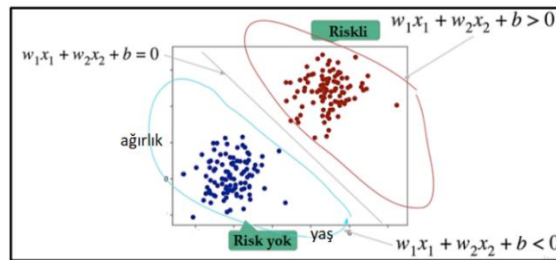
Şekil 20. Doğrusal regresyonun genel ifadesi

Ancak, sinir ağı ağırlıklarını girdilerimizden biri olarak kullanılan ağırlıkla birleştirmek istemiyoruz. Bu bölüm için onlara parametre diyeceğiz. Ayrıca başka bir parametreye bir bias eklediğimizi unutmayalım, bias terimi modelimizi daha anlamlı kılıyor. Bir bias terimi olmadan, bir çizgi yalnızca orijinden geçebilir. Ancak yanlılık terimiyle, çizgi keyfi olarak yukarı veya aşağı hareket ettirilebilir. Peki sadece iki girdimiz olduğuna göre sınıflandırmadaki amacımız nedir?



Şekil 21. Regresyonun basit gösterimi

Görselleştirmek kolaydır. Gördüğünüz gibi, yüksek tansiyon riski taşıyan kişiler kırmızı noktalarla etiketlenmiştir. Yüksek tansiyon riski olmayan kişiler mavi noktalarla etiketlenmiştir. Amacımız, farklı renkli noktaları en iyi ayıran çizgiyi bulmaktır. Şimdi, bu aşağıdaki resimden hatırladığımız önemli bir kural var, yani makine öğreniminin bir geometri probleminden başka bir şey olmadığıdır.

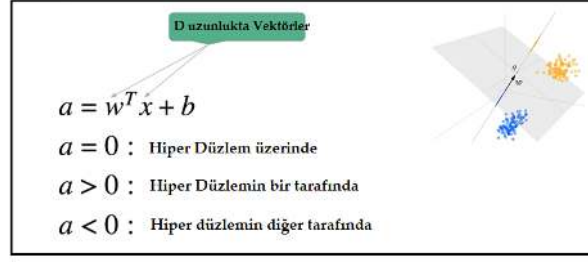


Şekil 22. Sınıflandırmanın basit gösterimi

Peki bu ne anlama geliyor? Bu, makine öğreniminin sihir olmadığı anlamına gelir, gördüğünüz gibi, tek yapmaya çalıştığımız bazı kırmızı noktalar ve bazı mavi noktalar arasına bir çizgi çekmek. Şimdi, çoğu zaman insanlar bir makine öğrenimini gizemli gibi göstermeye çalışırlar, ancak bunun aslında oldukça basit olduğunu görebiliriz. Sanırım hepimiz farklı renkli noktaları bir çizgiyle ayırmaya çalışmanın o kadar da gizemli olmadığı konusunda hemfikiriz. Başka bir ilgili kural da şudur, bu kural, tüm verilerin aynı olmasıdır. Peki bu ne anlama geliyor? Gerçi daha önce verilen örnek Asya ağırlığından yüksek tansiyon riskini tahmin etmekle ilgiliydi. Herhangi bir örnekle herhangi bir örnek verebilirdik. İki girişi olduğu ve ikili sınıflandırma yaptığımız sürece, geometrik problem her zaman aynıdır.

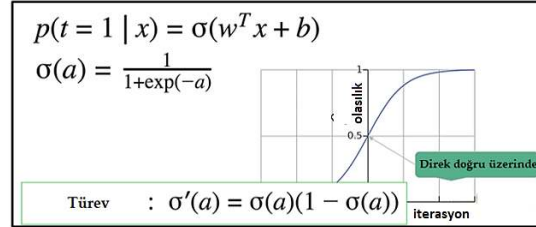
Hala kırmızı noktalardaki mavi noktaları bir çeşit çizgi veya bir eğri ile ayırmaya çalışıyoruz. Bu nedenle, kedi ve köpek resimlerini sınıflandırmaya çalışmamız veya bir kişinin kredisini temerrüde düşürme olasılığı önemli değil. Tüm veriler aynıdır, yani veri kümeniz ne olursa olsun aynı algoritmalar geçerlidir. Başka bir deyişle, kodlar değişmezdi. Yani kodunuzu yeni bir veri setinde denemek istiyorsanız, yapacak sıfır iş var. Başka bir deyişle, bir veri setinde bir sinir ağını nasıl kullanacağımızı öğrendikten sonra, tüm veri setlerinde bir sinir ağını nasıl kullanacağımızı biliriz. Yani bu algoritmayı veri setimizde nasıl kullanacağımızı merak etmeye gerek yok mu? Çünkü cevap bunu zaten biliyoruz. Şimdi, bunu söyledikten sonra, resmimize dönelim. Bu doğrunun denklem tarafından verildiğini hatırlayalım, sıfır eşittir W bir çarpı X_1 artı W iki çarpı X iki artı B , X bir'in yaşı ve X iki'nin ağırlığı temsil ettiğini varsayabiliriz. Ama az önce öğrendiğimiz gibi, bu gerçekten önemli değil. Tabii ki amacımız, bu kişinin mavi mi yoksa kırmızı mı renkli olacağını tahmin etmek için X_1 ve X değerleri verilen bir sınıflandırma yapmaktır. Açıkça çizginin bir tarafına kırmızı, diğer tarafına mavi atamak istiyoruz. Neyse ki, denkleminizi doğrunun bir tarafında kullanarak bunu belirlemek çok kolay, ifadesi W bir çarpı X bir artı W iki çarpı X iki artı B sıfırdan büyüktür. Çizginin diğer tarafında ifade sıfırdan küçüktür.

Tabii ki, kırmızı veya mavinin pozitif veya negatife karşılık gelmesi, parametrelerin işaretine bağlıdır. Daha sonra, sadece iki girdimiz olduğunu varsaymak yerine, herhangi bir sayıda girdimiz olduğunu varsayalım, bu durumda, bir modelin denklemini çarpım veya iç çarpım cinsinden yazmak daha uygundur.



Şekil 23. Regresyonun vektör gösterimi

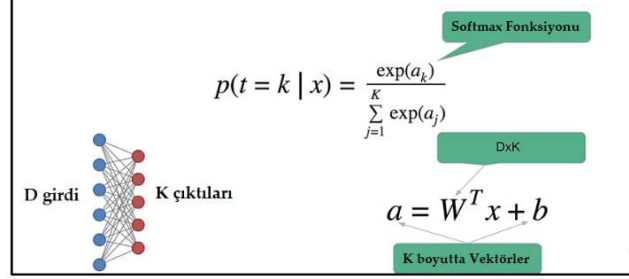
Bu durumda, her ikisi de W^T ve X 'de D boyutunda vektörlerdir, bias terimi b hala a skaleridir. Ayrıca geleneksel olarak, bu ifadeye a harfini kullanarak yeni bir değişken adı vereceğiz. a , aktivasyon anlamına gelir. Lojistik regresyon için, ifademizin sonuna bir sigmoid fonksiyonu da eklediğimizi hatırlayalım. Bu, bir modelin çıktısının bir olasılık olarak, bir inceleme olarak yorumlanabilmesini sağlar. Bu kısımda sigmoid denklemini not edelim. İşte sigmoid hakkında hatırlamamız gereken bazı önemli gerçekler Birincisi, girdi sıfır olduğunda çıktının sıfır noktası 0.5 olmasıdır. Bu, doğrunun bağlamında anlamlıdır. Bu, bir veri noktası tam olarak doğrunun üzerine düştüğünde, sigmoid içindeki ifadenin sıfır olduğunu söyler. Bu nedenle, hedefin bir olma olasılığı yüzde 50 ve hedefin sıfır olma olasılığı yüzde 50 olur.



Şekil 24. Sigmoid fonksiyonu ve olasılıksal karar çıkışı

Ayrıca, sigmoidin sağ tarafında her iki tarafta da asimptot olduğunu hatırlayın. Bire yaklaşıyor. Ve sol tarafta sıfıra yaklaşıyor. Son olarak, sigmoidin uygun bir türevi olduğunu hatırlayın. Bunun nedeni, türevin sigmoidin kendisi cinsinden ifade edilebilmesidir. Bu gerçek, birazdan gözden geçirilecek olan bir eğim inişi yapmak istediğinizde önemlidir. Ardından, çok sınıflı sınıflandırmayı desteklemek için öğrendiklerimizi genişletelim. Şimdi, sadece iki sınıflı tahmin etmek yerine, sınıfları tahmin edeceğiz. Bunu yapmak için softmax fonksiyonunu kullanacağız. softmax işlevi iki önemli şey yapar. İlk olarak, pozitif veya negatif olabilecek tüm aktivasyonları alır ve hepsini katlanarak pozitif yapar. Her sınıf için

K aktivasyonumuz olduğunu unutmayalım. İkincisi, tüm üs aktivasyonlarının toplamına bölünür. Bu, çıktıyı tekrar bir olasılık olarak yorumlayabilmemizi sağlar.



Şekil 25. Softmax fonksiyonu

Toplama böldüğümüzde, tüm değerlerin toplamının bir olduğunu unutmayalım. Bu nedenle, bu bir olasılık dağılımıdır. Spesifik olarak, küçük K çıktısı, X girdi vektörünün küçük K sınıfına ait olma olasılığı olarak yorumlanabilir. Başka bir deyişle, model çıktısı Y eşittir a , p , t eşittir K , verilen X . Son olarak, bekleme matrisi için büyük W harfini kullanıldığını dikkat edelim. Bunun nedeni, artık K Çıktısına sahip olduğumuz için, W 'nin D ile K boyutunda bir matris olmasıdır, ancak D girdileri ve çıktıları olan bir nöron dizisi olarak düşünülebilir. Ayrıca, bias terimi artık bir skaler yerine bir durum boyutu vektörüdür, şimdi hem ikili durumda hem de çok sınıflı durumda tahmin yapmak için tek bir nöronun nasıl kullanılacağını gözden geçirdik, bir sonraki adım modelimizin nasıl öğrendiğidir. Yani, modelimiz iyi tahminler yapacak şekilde nasıl buluruz. Bunu yapmak için, kayıp fonksiyonu denilen şeyi tanımlamamız gerekir. Yine, bu sadece bir inceleme olduğundan, bunlardan hiçbirini üretmeyeceğiz.

Ancak bunların negatif log olasılıkları olduğunu hatırlamalıyız. Bu nedenle, bu kayıp işlevler ile maksimum olabilirlik (maximum likelihood) tahmini arasında doğrudan bir bağlantı vardır, ikili durum için ikili çapraz entropiyi (binary cross entropy) kullanıyoruz, hatırladığımız gibi, bu Bernoulli olasılığının logunun negatifidir. Şimdi, denklem bize tanıdık geldiğinden emin olmalısınız. Çok sınıflı durum için, ikili duruma benzer kategorik çapraz entropiye (categorical cross entropy) sahibiz, bu kategorik olasılığın logunun negatifidir. Hedeflerin biçimini gözden geçirmek de faydalı olacaktır, çünkü ikili durumda çok sınıflı duruma göre biraz farklıdır, her iki durumda da, sınıflarımız varsa, o zaman hedeflerimiz her zaman sıfır değerlerini K eksi bire kadar taşıyacaktır. İkili durumda. Bunun

nedeni, hedeflerimizin yalnızca sıfır veya bir olabilmesidir. Çok sınıflı durumda, işler biraz daha karmaşıktır.

$L = - \sum_{i=1}^N \{t_i \log y_i + (1 - t_i) \log(1 - y_i)\}$	ikili durum Where $y_i = p(t_i = 1 x_i)$
$L = - \sum_{i=1}^N \sum_{k=1}^K t_{ik} \log y_{ik}$	çoklu sınıf Where $y_{ik} = p(t_i = k x_i)$

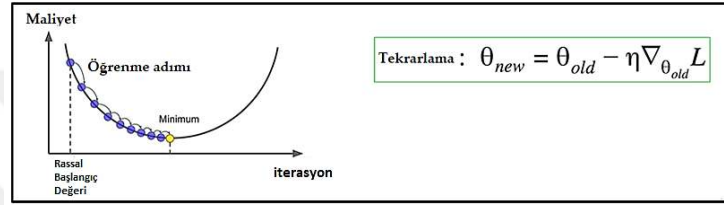
Şekil 26. Kategorik çapraz entropi

Diyeelim ki on tane sınıfımız var. Bu nedenle, teknik olarak hedefler kodda sıfırdan dokuza kadar değerleri tutmalıdır. Bu numaraları kullanmak istemiyoruz. Bunun yerine, kategorik çapraz entropi formülümüzde fark etmiş olabileceğimiz gibi, hedef t , bunun bir matris olduğunu ifade eden indekslere sahipti. Spesifik olarak, yapmak istediğimiz, sıfırdan K eksi bire kadar sayıları içeren bir N uzunluklu hedef listesini one-hot şeklinde kodlanmış ve K hedef matrisine dönüştürmektir. Temel olarak normal hedefler, sıfırdan dokuza veya daha genel olarak sıfırdan K eksi bire kadar olan sayıları içeren bir d uzunluğunda bir dizi olacaktır. Ama bir one-hot olduğunda, sadece sıfır ve bir sayılarını içeren 2 boyutlu bir dizi ve k elde edeceğiz. Yani örnek (sample) ve K sınıfı için hedef olan T ve K 'nin veri noktası K Sınıfına aitse bire, aksi halde sıfıra eşit olduğunu söyleyebiliriz. Bu kısım zaten oldukça uzun olduğu için, bir sonraki kısma geçmeden önce özetlemek gerekirse, bir makine öğrenimi modelinde verilerin nasıl yapılandırıldığını az önce gözden geçirdik. Ayrıca bir nöronun nasıl tahminlerde bulunduğunu tartıştık. Bir kayıp işlevi kullanarak bu tahminlerin nasıl değerlendirileceğini daha fazla tartıştık. Tahminler daha yanlış olduğunda kayıp işlevinin büyüdüğünü ve tahminler daha doğru olduğunda küçüldüğünü hatırlamalıyız. Bu nedenle, bir sonraki adımın kayıp fonksiyonunu en aza indirmek olması (loss minimization) gerektiği mantıklı olmalıdır.

2.5.1.5. Nöron Öğrenmesi

Bu kısımda nöronu tartışmaya devam edeceğiz, daha önce tahmin yapmayı öğrendik ve şimdi bir modelin nasıl öğrendiği veya eğittiği hakkında konuşmak istiyoruz. Sadece kayıp fonksiyona baktık ve bunun bize modelimizin ne kadar iyi olduğunu nasıl söylediğine baktık. Öğrenme aşamasındaki bir sonraki adım, bu kayıp işlevi nasıl en aza indiririz sorusuna cevap vermektir. Hatırladığımız gibi, kapalı form çözümü yoktur. Olsaydı, dL/dw

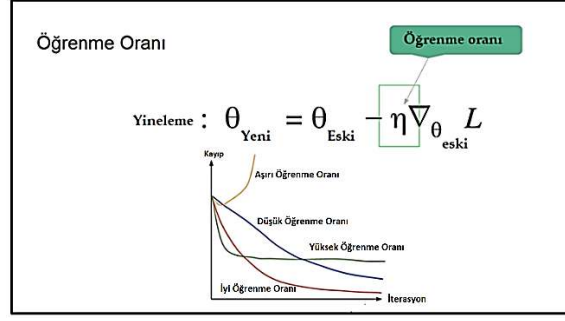
ve dL/db hesaplamak, sifra ayarlamak ve normal türevi çözmek gibi bir şey yapabiliydik. Bunun lineer regresyon için mümkün olduğunu hatırlayabilirsiniz, ancak ne yazık ki nöron ve sinir ağları gibi diğer birçok durumda mümkün değildir. Bu nedenle sayısal yöntemler ve özellikle gradyan inişi (Gradient Descent) kullanmak zorundayız. Temel olarak, model parametrelerine göre kaybın gradyanı boyunca yeterince küçük adımlar attığımız sürece, derin öğrenmede bazı yerel minimumlarda yakınsayacağız. Genelde genel minimumu bulamadığımız için endişelenmemize gerek yok. Görsel olarak eğimli (gradyan) inişi bir tepeden aşağı yuvarlamak gibi düşünebiliriz.



Şekil 27. Kategorik çapraz entropi

Yukarıdada gösterildiği gibi yapmak istediğimiz eğim yönünde çok küçük adımlar atmak ve sonunda en dibe ulaşmaktır. En altta yerel bir minimum olduğu için gradyanın sıfır olduğunu hatırlayalım. Bu nedenle, oraya vardığımızda, kayıp değerinizin değişmediğini göreceksiniz ve modelimizin yakınsadığı sonucuna varabiliriz. Lojistik regresyon için modelin tüm parametrelerini temsil eden genel bir harf olarak Teta harfini kullandığımızı unutmayalım. Teta, sinir ağları için w ve b 'lerini temsil edecektir. Her katmanın kullanımını ve b 'lerini temsil edecektir. Elbette, genel bir harf kullanarak birçok parametreye sahip olabileceğimiz için teta kolaylıktır. Tüm parametreleri aynı anda temsil ettiğini düşünebilirsiniz. Şimdi, ağırlık güncellemelerimizin (updating weights) adım boyutu olan burada tanıtılan yeni bir harf olduğunu fark edeceksiniz, bu Yunanca eta harfidir ve buna öğrenme oranı (learning rate) diyoruz.

Bu, bir derin öğrenme modeli eğitirken optimize edilebilecek birçok olası hiper parametreden biridir. Temel olarak, modelinizin makul bir sürede yakınsaması için bu değeri küçük ama çok küçük olmayacak şekilde ayarlamak istiyorsunuz. İşte aşağıdaki şekil öğrenme hızının farklı ayarlarında neler olduğunu hatırlamamıza yardımcı olacak kullanışlı bir resimdir. Çok büyük ayarlarsanız, kaybınızın sonsuza kadar patlayacağını görebilirsiniz.



Şekil 28. Öğrenme Oranı

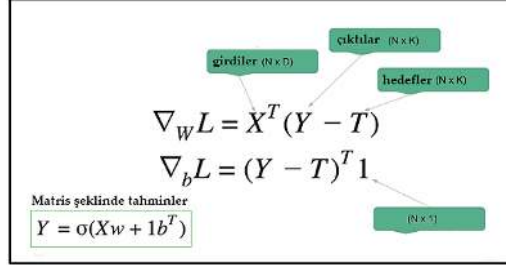
Çok küçük ayarlarsanız, modeliniz çok yavaş yakınsayacaktır. Temel olarak, modelinizin size güzel bir öğrenme eğrisi vermesi için bu değer optimize edilmesi gerekir. Bir yan not olarak, kodlamanız sırasında her zaman yineleme (iterasyon) başına kaybı çizmeniz şiddetle tavsiye edilir ve bunu tüm senaryolarımızın sonunda yapmamız gerekmektedir. Bu, modelimizin eğitimleri olduğunu ve parametrelerin optimal değerlerine yakınsadığını doğrulamak için bir akıl sağlığı kontrolü gibidir. Şimdi insanlar her zaman soruyor, en iyi öğrenme oranını nasıl seçerim? Ne yazık ki, yapabileceğimiz sihirli bir hesaplama yok. Neyse ki, bu konu kapsamında daha sonra tartışacağımız konulardan biri olacaktır. Yani o zamana kadar beklememiz gerekecek. Tartışmamız gereken bir sonraki şey, bu gradyanlar nelerdir? Bunların denklemlerinin ne olduğu hakkında henüz bahsetmedik, ama elbette, sonunda buna ihtiyacımız olacaktır, çünkü bu sadece bir inceleme olduğundan, bu kısımda senaryoya bunu koyacağız. İlerde cevaplar verilecektir. Ancak, bunları kendi başımıza nasıl türeteceğimizi bilmeliyiz. Bunu yapmazsak, bu gradyanları bulmak için ön koşulları tekrar gözden geçirmemiz şiddetle tavsiye edilir, kaybı W ve b 'ye göre basitçe ayırt ettik. Şimdi, bunu yapmanın iki yolu vardır. Bunu yapmanın bir yolu, skaler parametrelere göre kısmi türev almaktır. Bu muhtemelen daha kolay yoldur.

$$\frac{\partial L}{\partial W_{ik}} = \sum_{n=1}^N (y_{nk} - t_{nk}) x_{ni}$$

$$\frac{\partial L}{\partial b_k} = \sum_{n=1}^N (y_{nk} - t_{nk})$$

Şekil 29. Hataların gradyanı

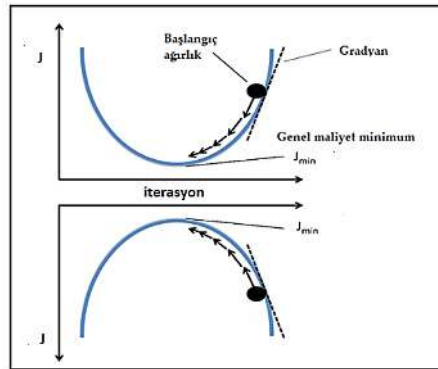
Bunu yapmanın ikinci yolu, her şeyi biraz daha karmaşık olan matris formunda denemek ve yapmaktır, ancak her iki durumda da aynı cevaba ulaşmalısınız.



Şekil 30. Gradyanların vektör gösterimi

Kodda, vektörleştirilmiş işlemleri kullanmanın for döngülerine kıyasla daha hızlı ve daha verimli olduğunu hatırlayalım. Bu nedenle, sinir ağları için denklemler genellikle vektörler cinsinden sunulsa da, bir makale veya ders kitabı okurken, kodda bu şekilde uygulanmazlar. Bunun yerine, büyük veri noktalarınız varsa, büyük N tahminlerini bir kerede hesaplayacaksınız. Hatırladığımız gibi, girdi veri matrisimiz D boyutunda olacak ve hedeflerimiz tek bir kodlanmış one-hot matrisi boyutunda olacaktır. Ve çünkü model çıktıları y , sabit kodlanmış hedeflerle aynı şekle sahip olacak ve ayrıca K ile bitecektir. Bu matrislerin tüm veri noktalarımızı bir kerede içerdiğine dikkat edelim. Yani burada tahmin ve gradyan denklemleri açısından büyük Matris X ve büyük matris T ve büyük matris Y cinsindendir. Ayrıca, bias faktörünün gradyanı için bunun bir numara değil, o zaman uzunlukların bir vektörü olduğuna dikkat edelim.

Bu, pratikte tüm Y eksi T örneklerini toplamaya eşdeğerdir, bunlar kodda kullanacağımız şeylerdir. Bu kısımda daha sonra öğreneceklerimizin bir önizlemesi olarak, özellikle ve gerçekten çok büyük hale gelirse, bunun neden sorunlu olabileceğini düşünelim. Şimdi, yaygın bir karışıklık noktası, eğimli iniş ve eğimli yükseliş arasındaki farktır. Gerçek şu ki, fark sadece semantiktir, ancak matematiksel olarak bunlar tamamen aynıdır.



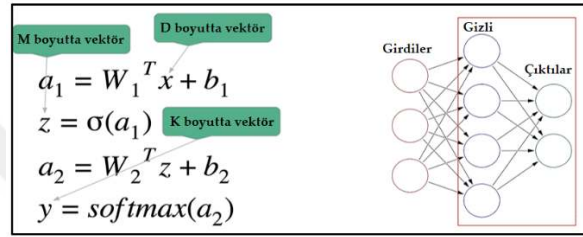
Şekil 31. Öğrenme adım büyüklüğü

Küçültmek istediğimiz bazı kayıp J fonksiyonumuz olduğunu varsayalım. Öğrendiğimiz gibi, genel anlamda gradyan iniş yaparak J 'yi optimize edebiliriz, bu güncellemeyi tekrar tekrar yapmak olacaktır. Teta eşittir teta eksi öğrenme oranı çarpı J 'nin tetaya göre gradyanı. Ancak, minimize etmek istediğimiz kayıp bir fonksiyon yerine, maksimize etmek istediğimiz bir uygunluk (fitness) fonksiyonumuz olduğunu varsayalım. Bu, makine öğreniminde çok yaygındır ve her ikisinde de rahat olmalısınız. Eksi J 'ye eşit olduğumuzu varsayalım, J 'yi maksimize etmek ile minimize etmek arasında hiçbir fark olmadığı açık olmalıdır. X kareyi en aza indirmeye ve eksi X kareyi büyümeye çalışalım. Her iki durumda da optimal X değerinin aynı olması gerektiğini görmeliyiz. Şimdi maksimize etmek için istediğimiz varlıkları gradyan yapacağız. Bu sadece güncellemedir. Teta eşittir teta artı öğrenme oranı çarpı tetaya göre gradyanımızdır. Ama sadece eksi J olduğunu hatırlayalım. Bu formüle eksi J eklersek, J 'yi en aza indirirken daha önceki düzenli güncellememizi geri aldığımızı görebilirsiniz. Gördüğünüz gibi, matematiksel olarak gradyan yükselişi ve gradyan inişi aslında tamamen aynıdır.

Tek fark, eksi işaretini nereye koyduğumuz ve onu hangi amaç fonksiyonundan ayırdığımızdır. Verdiğimiz önceki gradyanlar için bunların maksimize etmek değil, minimize etmek istediğimiz şeyin gradyanları olduğuna dikkat edelim. Ama kodda tam tersini görürseniz şaşırmayın, güzel bir alıştırma olarak, bu konu zaten oldukça uzun olduğu için, şimdiye kadar yaptıklarımızı özetlemek için bir sonraki kısımda sinir ağlarını gözden geçireceğiz. Yaptığımız şey bu. Hem ikili hem de çok sınıflı durum için nöron modelini inceledik. Model tahminleri yapmak için hesaplamaların nasıl yapıldığını inceledik. Ayrıca ikili ve çok sınıflı durum için doğru kayıp işlevlerini de inceledik. Hem ikili hem de çok sınıflı durumda hedeflerin nasıl temsil edileceğini gözden geçirdik. Daha da önemlisi, Multiclass durumunda, yalnızca sıfırları ve birleri içerecek şekilde sıcak ve soğuk hedefleri kazandık. Daha sonra, bir nöronun gradyan inişi yaparak özel olarak nasıl öğrendiğini inceledik. Bu, iki önemli kavramı hatırlamamızı sağladı. Birincisi, iyi bir öğrenme oranının nasıl seçileceğiydi. Genellikle bunu sezgiyle yapabilirsiniz, ancak bu konunun ilerleyen bölümlerinde diğer yöntemlere bakacağız. İkinci olarak, öğrenme yöntemimizin gradyan inişi olduğu model parametrelerine göre kaybın gradyanına baktık. Gradyanı hesaplamamız oldukça önemli olduğunu artık hayal edebiliriz.

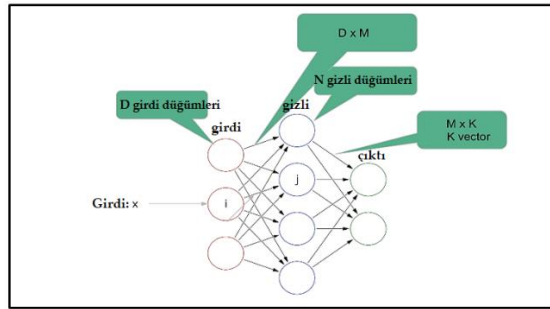
2.5.1.6. Sinir Ağları Tahminleri

Bu kısımda, nöronlar ve sinir ağları hakkındaki incelememize devam edeceğiz. Bu konu, ilk önce nasıl tahminlerde bulunacağımızla başladığımız ve ardından modelin nasıl öğrendiğini tartıştığımız önceki konularla aynı temel taslağı izleyecektir. Tamam, sinir ağlarını düşünmenin basit bir yolu, bir tür lojistik regresyon gibi olmalarıdır, birlikte kümelenirler, bir sinir ağına baktığınızda birden fazla girdiye ve tek bir ikili çıktıya sahip olduğumuz şu anda ikili lojistik regresyonu düşünelim.



Şekil 32. Çok katmanlı bir algılayıcı (MLP)

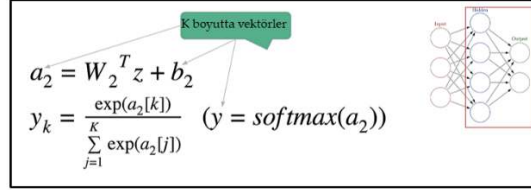
Örneğin, sinir ağları lojistik regresyonların alt birimlerinden nasıl oluştuğuna dikkat edelim. Yukarıdaki şekilde sinir ağların girdi, gizli (hidden) ve çıktı katmanlar hepsi birer lojistik regresyondur, ayrıca bu düğümlerin değerleri tam olarak hesaplanmıştır. sadece lojistik regresyon yapıyor olsaydık, normalde onları hesaplayacağımız gibi. X ile veri alırsak, bias terimini ekleriz ve ardından bunu sigmoid fonksiyonundan geçiririz. Şimdi, elbette, tüm bu hesaplamaları bir kerede yapmak istiyoruz, varsayalım ki bazı girdilerimiz var. D 'ye bağlı olan X vektörü, gizli birimlerin sayısının M olduğunu varsayalım, yani düğümlerin orta katmanında bunlardan bazıları var.



Şekil 33. MLP vektörlerle gösterimi

Bu, girdiden gizli katman boyutuna D 'ye M ile giden ağırlık matrisinin, dolayısıyla $Wl(i,j)$ 'nin, J biriminde giriş birimi göz ve kafayı birleştiren bağlantının gücü olduğu anlamına gelir. Karşılık gelen taban vektörü M 'dir. Daha sonra gizli katmandan çıktıya gitmeliyiz. Şimdi diyelim ki K çıktılarımız var. Bu durumda, çıktıdan gizliye ağırlık $M \times K$ boyutuna sahip olur ve gizliden çıktıya sapma, k boyutunda bir vektör olur. Tamam, şimdilik çalışmamız gereken tüm parametreler bunlar sadece bir gizli katmanı olan sinir ağlarıyla ilgilendiğimizi varsayacağız. Bu konunun ilerleyen bölümlerinde, herhangi bir sayıda gizli katman içeren sinir ağlarının nasıl oluşturulacağını öğreneceğiz, model çıktısını almak için tüm bu öğeleri nasıl birleştirebiliriz? Söz verildiği gibi, neredeyse tam olarak lojistik regresyon gibi. Bu yüzden girdileri ileri alıyoruz ve bir bekliyoruz, devrik (transpos) , X ile çarpıyoruz ve b_1 bias vektöründe hatırladığımız gibi, matris çarpımı yaparken iç boyutların eşleşmesi gerekiyor. Durumun böyle olduğunu teyit etmelisiniz. Bu lineer dönüşümü a_1 olarak adlandıracamız, tıpkı daha önce yaptığımız gibi. Bu, aktivasyonun kısaltmasıdır. a_1 'in M boyutunda bir vektör olduğu açık olmalıdır.

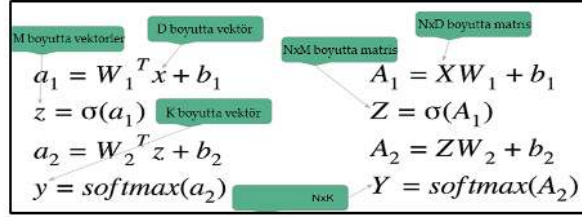
Bunun nedeni, her birinde M gizli birim bulunması ve bunların birlikte M boyutunda a vektörü oluşturmasıdır. Ardından, bir öge bazında etkinleştirme ya da aktivasyon fonksiyonu uyguluyoruz. Şimdi, tipik olarak, lojistik regresyon dünyasından geliyorsanız, bu bir sigmoid olacaktır. Aslında, bir sigmoid kullanmak tamamen iyidir. Bu günlerde daha yaygın bir seçim, bir rampa gibi görünen kurtarma işlevi olacaktır. Hatırlayacağımız gibi, şu an için etkinleştirme işlevleri için birçok olası seçenek bulunmaktadır. Bunun nedeni, her birinde M gizli birim bulunması ve bunların birlikte M boyutunda a vektörü oluşturmasıdır. Hatırlayacağımız gibi, şu an için etkinleştirme işlevleri için birçok olası seçenek bulunmaktadır. Bu herhangi bir endişe kaynağı değildir. Bu aktivasyon fonksiyonunun çıktısını Z olarak arayacağız, yani aktivasyon fonksiyonuna a_1 aktivasyonunu ilettiğimizde, Z elde ederiz. Şimdi, etkinleştirme işlevi öge akıllıca çalıştığından, Z 'nin olanla aynı şekle sahip olduğuna dikkat edelim. Her ikisi de M boyutunda vektörlerdir. Z 'ye gizli vektör diyoruz, X girdi vektörüydü, Z genellikle gözlemlemediğimiz bir şey için kullandığımız bir harfken, X genellikle gözlemlemediğimiz bir şey için kullandığımız bir harf. Bir sonraki adım, gizli katmandan çıktı katmanına gitmek, bunu yapmak için Z 'yi alıyoruz ve esasen daha önce yaptığımız aynı dönüşümü yapıyoruz.



Şekil 34. MLP ve Çok Sınıflı SoftMax

Başı transpoze etmek için çıktı ağırlık matrisine alıyoruz ve Z ile çarpıyoruz. Ardından B_2 sapma vektörünü ekliyoruz. Bu bize çıktı aktivasyonumuz olan A_2 'yi verir. W_2 'nin M ve K şekline sahip olması ve Z 'nin M boyutunda bir vektör olması nedeniyle, A_2 'nin K boyutunda bir vektör olduğunu doğrulamanız gerekir. Bu mantıklı çünkü sinir ağımızın çıktısı, her biri farklı bir değere sahip k ayrı düğümdür. Ancak, hatırladığımız gibi, sınıflandırma için hala işimizin bitmediğine dikkat edelim, softmax işlevini uygulamak istiyoruz. Softmax işlevi bir aktivasyon işlevi olarak düşünülebilir, ancak genellikle gizli katmanlarda değil, yalnızca çıktıda kullanılır. Bir önceki kısımdan hatırladığımız gibi softmax fonksiyonu çok sınıflı problemler için kullanılıyor. Tamam, bir sinir ağında nasıl tahmin yapılacağını özetlemek için, tüm denklemleri bir kerede gözden geçirelim. İlk olarak, D boyutunda bir vektör olan X girişinde doğrusal bir dönüşüm gerçekleştiriyoruz. Bunun için W_1 ve b_1 kullanıyoruz, girdilerin ağırlıkları, bir gizli katman. Bu a_1 bize aktivasyonu verir. Ardından, sigmoid, Tanh, Relu veya başka herhangi bir şey olabilen bir etkinleştirme işlevi uygularız.

Bu bize yine Z 'nin M boyutunda bir vektör olduğuna dair bir not verir. Yani giriş katmanından oradaki başa bu şekilde gidiyoruz. Ardından gizli katmandan çıktı katmanına geçiyoruz. Bunu yapmak için W_2 ve b_2 'yi kullanarak başka bir doğrusal dönüşüm uygularız. Bu bize a_2 verir. Son olarak, softmax işlevini a_2 'ye uygularız, bu bize neden bu konu bağlamında olduğu gibi, çok sınıflı sınıflandırma yapmak istediğimizi varsaydığımızı düşündüğümüz çıktı tahmini olasılıklarını verir. Y 'nin K boyutunda bir vektör olduğuna dikkat edelim. Son olarak, lojistik regresyonda olduğu gibi, kodda bu işlemi genellikle tüm N kadar veri noktalarında aynı anda gerçekleştirdiğimizi hatırlayalım. Bu nedenle, bu durumda X ve Y 'yi büyük harfle yazıyoruz ve bu durumda bunların hepsi matristir. Spesifik olarak toplu durumda, büyük X , $N \times D$ boyutunda bir matristir, büyük Z , $N \times M$ boyutunda bir matristir ve büyük Y , $N \times K$ boyutunda bir matristir.



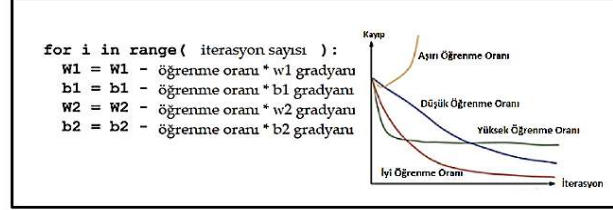
Şekil 35. X , $N \times D$ ve Z , $N \times M$ büyük matrisleri

Tamam, yani bir sinir ağıyla nasıl tahmin yapılacağıyla ilgili her şey bu. Sıradaki soru w_1 , b_1 ve w_2 ve b_2 için doğru değerleri nasıl bulacağız? Yine cevap, kayıp bir fonksiyon oluşturmamız ve ardından model parametrelerine göre kaybı en aza indirmek için gradyan inişi gerçekleştirmemiz gerektirir. Hatırladığımız gibi, nöronlar ve sinir ağıları için kullandığımız kayıp fonksiyonlar aynıdır. Aslında, nöronu, gizli katmanların olmadığı bir sinir ağının özel bir durumu olarak düşünebiliriz. Bu nedenle, aynı yasalar daha önce geçerli olmaya devam eder. İkili problemler için ikili çapraz entropiyi ve çok sınıflı problemler için kategorik çapraz entropiyi kullanıyoruz. Bildiğiniz gibi bir sonraki adım gradyan iniş veya geri yayılım yapmaktır. Temel olarak, bu cevaba nasıl ulaştığımızı hatırlamıyorsanız, matematiği kullanarak gradyanları kendi başımıza türetmeye çalışmalıyız.

Bu degradelerin, bu kısmın amacı için seçtiğimiz gizli etkinleştirme işlevine bağlı olduğunu unutmayalım. Sigmoid aktivasyon fonksiyonunun sadece ekstra dikkatli olmak için kullanıldığını varsayalım. Bu denklemler hakkında birkaç not alalım. İlk olarak, bildiğiniz gibi, her zaman maksimize etmek veya minimize etmek istediğimiz hedefin gradyanını alabiliriz. Bu durumda, kayıpla ilgili gradyanları verdik, bu da şu anlama gelir: hedefi en aza indirmek için gradyan inişi. İkincisi, konuda hatırladığımız gibi, bias terimleri için gördüklerimizin vektörünü uygulamamalıyız. Bunun yerine, aynı şeyi etkili bir şekilde yapan tüm N örnekleri boyunca toplamamız gerekir. Birlerin vektörü, yalnızca bu işlemi matematiksel olarak nasıl temsil ettiğimizdir. Üçüncüsü, daha önce görmediyseniz, içinde nokta bulunan daire Y elementinin çarpımıdır. Bu lineer cebirde çok yaygın bir işlem olmadığı için evrensel bir sembol yoktur. Ama bunun benim deneyimlerimde en yaygın olanı olduğunu buldum.

Tamam, şaşırtıcı olmayan bir şekilde, eğitim şu şekilde ilerler, her yinelemelerde birkaç kez döngü yaparız, model çıktıları almak için sinir ağından ileriye doğru bir geçiş yaparız. Ardından, her parametre için gradyanları hesaplamak için model çıktıları kullanırız. Ardından, daha önce olduğu gibi bir adım gradyan inişi yaparız, yeterince hızlı ama çok hızlı olmayan bir öğrenme oranı seçmek önemlidir. Ve eğitim sürecinin başarılı olduğunu

onaylayabilmemiz için kaybımızı yineleme başına planlamak her zaman iyi bir fikirdir. Bu konuda gözden geçirmek istediğimiz son şey, düzenlileştirme (regularization) kavramıdır. Bildiğiniz gibi makine öğrenimindeki amacımız sadece bir modeli bir veri kümesine sığdırmak değildir. Daha önce hiç görmediğimiz veriler üzerinde doğru tahminlerde bulunmaktır ki buna genelleme (generalization) diyoruz.



Şekil 36. Kayıpların döngüde hesaplanması

Modelimizin genellenmesini istiyoruz. Peki, bunu yapmak için. Veri kümemizi eğitim ve test kümelerine bölerek eğitmek yaygındır. Az önce tartıştığımız süreç sadece eğitim setini kullanıyor. Modelinizi değerlendirirken test setini kullanırız. Şimdi, yaygın bir senaryo, eğitim setinizdeki kaybın güzel bir şekilde yakınsamasına rağmen, test setinde çok iyi bir şekilde yakınsamamasıdır. Bu, modelimizin fazla uyumlu (overfitting) olduğunu veya başka bir deyişle iyi genelleme yapmadığını gösterir. Bu, modelimizin görmediği veriler üzerinde iyi performans göstermediği anlamına gelir ve bu arzu edilmez. Bu soruna bir çözüm, sınır ağı ağırlıklarımızı düzenli hale getirmektir, hatırladığımız gibi, bu aslında ağırlıklarımızın çok fazla büyümesine izin vermemek anlamına gelir. Bunu başarmak için, yasaların düzenleme cezası (regularisation penalty) işlevini değiştiriyoruz.

$$L_{\text{Düzeltilme}} = L_{\text{çapraz-entropy}} + \lambda ||\theta||_2^2$$

Düzenleme Cezası

Şekil 37. Düzenleme cezası

Spesifik olarak, bir hiper parametre lambda ekliyoruz ve onu mevcut yasalara göre ağırlık matrislerinin karesinin büyüklüğü ile çarpıyoruz. Gradyanı hesaplarken cezanın gradyanı da dahil edilmelidir. Ve gördüğümüz gibi, ağırlıklardan herhangi biri çok büyürse, bu genel kaybı artıracaktır. Bu nedenle, bu cezayı ekleyerek, ağırlıkların küçük olmasını ve

fazla sığmamasını teşvik ediyoruz. Lambda, tıpkı optimize edilmesi gereken öğrenme oranı gibi, başka bir hiper parametre değildir.

2.5.1.7. Olasılıksal Dereceli Azalma

Bu kısımda, bu konunun bir sonraki konusu olan gradyan inişinin nasıl daha verimli bir şekilde gerçekleştirileceğini tartışacağız. Bildiğimiz gibi öğrenme ve derin öğrenme yöntemimiz gradyan inişidir. Önem verdiğimiz bu gradyan, sinir ağının parametrelerine göre kaybın gradyanıdır. Kayıp, örneğin bireysel kayıpların toplamı veya ortalaması olarak tanımlanır. Şimdi, buna bir denklem gibi bakarsanız, neden daha fazla tartışılması gerektiği açık değil. Bu yüzden gerçek dünya pratik senaryolarını düşünmeye yardımcı olur. Bir çalışmada MNİST veri seti kullanılırsa, MNİST veri seti içerdiği görüntü onbinlerce mertebesinde ve nispeten az işlem gücüyle ve daha kısa sürede bu boyuttaki bir veri kümesi üzerinde eğitim gerçekleşir. Ancak, ImageNet gibi arama boyutunda bir büyük veri seti kullanırsak ne olacağını düşünün.

$$L = \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

$$\nabla_{\theta} L = \sum_{i=1}^N \nabla_{\theta} (y_i - \hat{y}_i)^2$$

Şekil 38. Kayıpların ortalama hesaplaması

Bu durumda, büyük N değeri, toplam örnek sayısı bir milyon mertebesinde. Şimdi, bildiğimiz gibi bir milyon oldukça büyük bir sayıdır. Bu nedenle, maliyetin kendisini hesaplamak bile zor olacaktır. Bir milyonun üzerinde değerin toplanmasını gerektirir. Hatırladığımız gibi, gradyanlar ayrıca tüm N sample'leri üzerinde bir şey gerektirir. Bu nedenle, kaybı hesaplamamış olsak bile, gradyanı hesaplamada yine de takılıp kaldık. Öyleyse çözümü anlamak için gerçek bir dünya senaryosu düşünelim, bir deney yapmak istediğimizi varsayalım, örneğin, yeni bir ilacın işe yarayıp yaramadığını bilmek istiyoruz veya evde okumanın ilkokul öğrencilerinin notlarını yükseltmeye yardımcı olup olmadığını bilmek istiyoruz. Doğru cevabı bulabilmek için, ilacımızı tüm dünyadaki her insan üzerinde test etmeliyiz. Açıkçası, şu anda dünyada yedi milyar insan olduğu için bu mümkün değil.

Peki bunun yerine ne yapıyoruz? Cevap, popülasyonun küçük bir örneği üzerinde bir deney yapmamızdır. Örneğin, ilacı yedi milyar insanın tamamı yerine 100 veya 1000 kişi üzerinde test edebiliriz.

Açıkçası, deneye katılmadıysanız, ilaç sizin üzerinizde asla test edilmedi. Bu nedenle, ilacın kişisel olarak sizin için çalışıp çalışmayacağını teknik olarak bilmiyoruz. Ancak, örneğimize dayanarak tahminde bulunuruz, popülasyon örneğimizin bir bütün olarak popülasyonla aynı özelliklere sahip olduğunu varsayabiliriz. Bu nedenle çalışma katılımcılarını rastgele seçmek çok önemlidir. Örneğin, çalışmanızdaki herkesin çok uzun veya çok kilolu olmasını sağlayamazsınız, çünkü bu sonuçlarınızı etkileyebilir. Çalışmanızı sadece fazla kilolu insanlar üzerinde yaparsanız, ilaca verilen yanıt farklıysa, ilacın fazla kilolu olmayan kişilerde işe yarayıp yaramadığını bilemezsiniz. Başka bir somut örnek vermek gerekirse, dünyadaki herkesin boyunu ölçmek istediğimizi varsayalım, dünyadaki herkese gidemeyeceğimizi ve onlara ne kadar uzun olduklarını soramayacağımızı varsayalım. Ancak, dünyadaki rastgele 1000 kişiden bize boylarını söylemelerini isteyebilir ve bu bin kişinin ortalama boyunun tüm dünyadaki ortalama yüksekliğe yeterince yakın olduğunu varsayabiliriz. Benzer şekilde, bir milyon boyutlu veri setimizin tamamından küçük bir numune grubu üzerindeki kaybı aldığımızı varsayalım. Bu küçük numune serisindeki ortalama kaybın, bir milyon numunelik tam veri setindeki ortalama kayba yakın olduğunu varsaymak mantıklıdır.

Tamam, peki bunu nasıl uygulamaya koyabiliriz? Cevap, gruplar (batch) halinde Grdient'i iyi yapmaktır. Veri setimizin N'nin bin'e eşit olduğunu varsayalım. Bu verileri, her biri yüz büyüklüğünde on gruba böldüğümüzü varsayalım. Ardından gradyan iniş döngümüz aşağıdaki gibi görünüyor. İlk önce, bir döngü boyunca yinelemeyi temsil eden bir dış döngümüz var. Sonra her grubun üzerinde dönen bir iç döngümüz var.

Yığın 1	Yığın 2	Yığın 3	Yığın 4	Yığın 5	Yığın 6	Yığın 7	Yığın 8	Yığın 9	Yığın 10
------------	------------	------------	------------	------------	------------	------------	------------	------------	-------------

Şekil 39. Yığınlar (batch) halinde Grdient

Bu, daha önce tam gradyan inişi (full gradient descent) diyeceğimiz şeye karşıdır. Bu durumda, her iterasyonda (epoch) sadece bir döngümüz vardı ve her iterasyonda yaptığımız her matematiksel işlem tam veri setinin üzerindeydi, ama grup durumunda yaptığımız her işlem

sadece batch üzerindedir. Bu konuyu tamamlamak için, batch büyüklüğünü bir olarak (batch=1) kullandığımız uç durumlar hakkında konuşmamız gerekiyor. Daha önce yaptığımız şey zaten teknik olarak stokastik olsa da, bazen insanlar buna stokastik gradyan inişi veya SGD diyor. Daha sonra adlandırma kuralları hakkında daha fazla konuşacağız. Bir şekilde, her seferinde bir örneğe bakmak, öğrenme bağlamında çok sezgiseldir, hayatınızdaki yaşam deneyimleriniz bağlamında bir anlam ifade eder. Bir seferde yalnızca bir şey deneyimlersiniz ve her zaman bu deneyime dayanarak biraz öğrenirsiniz. Esasen ilerledikçe öğrenirsiniz. Geçmişte gördüğünüz her şeyden sürekli olarak öğrenmeye çalıştığınız bir veritabanınız yok. Bu anlamda, bir seferde bir örnek öğrenmek, insan bakış açısından size daha tanıdık ve daha sezgisel görünebilir. Ayrıca, bilgisayarlarda her seferinde bir örneğe bakmak da yapılır, örnek olarak Pekiştirmeli ya da Takviyeli öğrenmeye (reinforcement learning) bakabiliriz.

Takviyeli öğrenme, çevresiyle etkileşime girecek bir algoritma veya etken (agent) oluşturmakla ilgilidir. Etken her eylemde bulunduğu anda, dünyanın yeni bir durumuna gelir ve öğrenebileceği bir ödül sinyali alır. Bu geçişlerin her biri, öğrenebileceği bir örnek olarak kabul edilir. Bu, bir seferde yalnızca bir örnek öğrenen gerçek dünya algoritmasının bir örneğidir. Peki, stokastik gradyan inişi için kod neye benziyor?



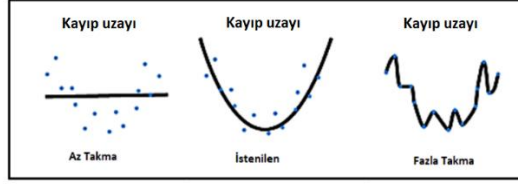
Şekil 40. Takviyeli öğrenme

Bu durumda, batch'leri nasıl indeksleyeceğimizi bulmak zorunda olmadığımız için SGD'den çok daha basittir. Yine iç döngü içinde iki for döngümüz var, her seferinde bir tane olmak üzere tüm N örnekleri arasında döngü yapıyoruz. Yani umarım bu çok karmaşık değildir. Bu konunun temalarından biri, derin öğrenmede kullanabileceğimiz çok daha fazla seçeneği öğreneceğiz. Ancak, hangi seçeneği seçmemiz gerektiği tam olarak net değil. Bu boyuttaki bu seçeneği bir şekilde tanıttığımızı göre, bu bizim için daha zor olabilir, çünkü bize seçim konusunda endişelenmemiz gereken başka bir şey daha verir. Batch boyutu için hangi değeri seçmelisiniz? Uygulamada, insanlar genellikle iki üslerini kullanırlar, örneğin 8, 16, 32, 64, 128, 256 veya 512 vb. Genel olarak konuşursak, bilgisayarınızın belleğini boğmadan mümkün olduğunca büyük bir batch boyutuna sahip olmak istersiniz. Modern

derin öğrenmede, birlikte çalıştığınız bazı ağlar o kadar büyük olacaktır ki, 8'den büyük bir batch iş boyutunu işleyebilirsiniz. Öte yandan, verilerinizdeki ağız, 512'lik bir batch iş boyutunun uygun olacağı kadar küçük olabilir.

Peki neden mümkün olduğunca büyük bir batch boyutu istiyoruz? İşte iki neden. İlk olarak, her batch için bunun sadece tam kaybın bir tahmini olduğunu unutmayalım. Bu nedenle, ne kadar çok örnek eklersek o kadar doğru olur. Bu tıpkı bir madeni para çevirmeye benzer. Yazı tura atıyorsanız, tura gelme olasılığının bir ölçüsüdür. O zaman parayı ne kadar çok atarsanız, ölçümlerinizi o kadar doğru olur. İkinci neden, unutmayalım ki for döngüleri yavaştır. Toplu gradyan inişi yaparak ikinci bir iç döngü oluşturduk. Bu for döngüsü üzerinde ne kadar çok yinelememiz olursa, kodumuz o kadar yavaş olur. Bu nedenle, her seferinde yalnızca bir örneğe (only one sample at a time) baktığımız iç for döngüsünün en fazla yinelemesi olan stokastik gradyan inişinin, aynı anda birden fazla örneğe bakmaktan daha yavaş olduğu sonucuna varabilirsiniz. Konu ile ilgili adlandırmada herkesin bu yöntemlerin tam olarak ne olarak adlandırılması gerektiği konusunda hemfikir olmadığını belirtmek isteriz. Tensorflow2 sorgulamalarında var olana gidersek, SGD'nin herhangi bir zamana atıfta bulunduğunu göreceksiniz, hiç biri bir kerede tüm veri setine bakmadığından tüm veri setine bakmıyorsunuz. Keras'ta, tüm veri kümesine bir kerede bakan bir şeyin adı yoktur.

Ancak buna şöyle de bakabiliriz. Bütün veri kümesine baktığınızda, buna tam gradyan inişi (full gradient descent) diyorsunuz. Bir seferde bir gruba baktığınızda, buna toplu gradyan inişi (batch gradient descent) diyorsunuz. Her seferinde bir örneğe baktığınızda, buna stokastik gradyan inişi (stochastic gradient descent) diyorsunuz. Ancak başka bir adlandırma kuralı şudur, bu, batch gradyan inişi terimini, özellikle tam veri kümesine baktığınızda farklı bir şekilde yeniden kullandığından kafa karıştırıcıdır. Bu, batch gradyan inişi dediğiniz şeydir. Ardından, her seferinde bir batch'e baktığınızda buna mini batch gradyan inişi denir. Her seferinde bir örneğe baktığınızda, buna hala stokastik gradyan inişi denir. Bu konu için son bir düşüneğimiz düzenleştirmedir (regularisation), bu kısımda öğrendiğimiz diğer teknikler için de durumun böyle olduğunu göreceksiniz, burada stokastik yapıları nedeniyle sinir ağını düzenlemeye yardımcı oluyorlar.



Şekil 41. Öğrenmede takma durumları

Düzenlemenin, fazla takmayı (overfitting) ya da aşırı uyum göstermeyi önlemek için kullandığımız bir teknik olduğunu unutmayalım. Peki batch gradyan inişi veya mini batch gradyan inişi aşırı takmaya nasıl yardımcı olabilir? Pekala, toplu halde verilere bakarak, kaybın hiçbir zaman eğitim setinin tam kaybına tam olarak eşit olmadığını hatırlayalım. Ve gördüğünüz her veri grubu için biraz farklı olacak. Bu nedenle, ağırlıklar her zaman biraz farklı bir yönde hareket edecektir, asla tam kayba göre tam olarak gradyanın yönünde hareket etmeyecektir. Bu biraz gradyan iniş sürecine gürültü (noise) eklemek gibidir. Bu nedenle, tam kayba uydurmak (fitting) fazla uydurmaya yol açarsa, bu sürece biraz gürültü eklemek bu etkiyi azaltabilir, çünkü asla tam kaybı tam olarak optimize etmeye çalışmıyoruz. Eğitim sürecinde farklı noktalara gürültü eklemek, sinir ağlarında aşırı uyumu önlemenin yaygın bir yoludur.

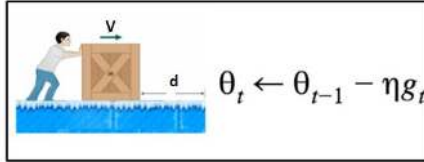
2.5.1.8. Momentum

Bu kısımda, momentum adı verilen düz gradyan inişi üzerindeki en etkili gelişmelerden biri olduğunu düşündüğümüz şeye bakacağız. Peki momentum nasıl çalışır? Tepeden aşağı yuvarlanan top benzetmesini düşünelim, bu güzel çünkü eğimli iniş sırasında Dünya'nın yüzeyini bir tepenin dibi olarak hayal ediyoruz. Ama daha da iyi bir benzetme, bir şeyi buz gibi sürtünmesiz bir yüzeyde hareket ettirmektir. Bu nedenle, daha önce paten kaymaya gittiyseniz, buz üzerinde çok hızlı ve çok fazla güç kullanmadan kolayca süzölebileceğinizi bilirsiniz çünkü momentum sizi daha önce gittiğiniz şekilde taşır.



Şekil 42. Momentum görsel benzetimi

Buz üzerinde bir kutuyu itmeyi hayal edebilirsiniz ve kutu bir noktadan diğerine oldukça kolay hareket eder çünkü bir kez itebilirsiniz ve kayar. Fizikte tam olarak olan budur. Sürtünme yoksa, bir nesne zaten gittiği yönde gitmeye devam edecektir. Tabii ki, hala biraz sürtünme var, bu yüzden sonunda kutu yavaşlayacak ve sonunda duracak. Göreceğiniz gibi, eğim inişindeki momentum da bir süre sonra yavaşlar. Bu benzetmeye devam etmek için, momentumu olmayan bir durum düşünmeye çalışalım. Şimdi, akılda tutunuz, burada gradyan iniş momentumundan bahsediyoruz, fizik momentumundan değil, çünkü momentumun fizikteki tanımı biraz farklı. Şimdi, buz yerine bir kutuyu kuma itmeye çalıştığınızı hayal edin. Şimdi, kumun çok fazla sürtünmesi olduğu için bunun çok zor olacağını hayal edebilirsiniz. Yani kumdaki bir kutuyu iterseniz, ittiğiniz kadar ileri gidecek, daha fazla değil. Kutunun tekrar hareket etmesini istiyorsanız, tekrar itmeniz gerekir. Bu, her seferinde momentum olmadan gradyan inişi gibidir. Hareket etmek istiyorsak, eğim yönünde hareket edebilmemiz için bir eğim olması gerekir. Şimdi, bu fikirleri matematiğe koymaya çalışalım, momentum olmadan düzenli gradyan inişiyle başlayalım, böylece aşağıda gösterildiği gibi $\theta_t = \theta_{t-1} - \eta g_t$ bir fade'yi elde ederiz, η öğrenme oranıdır ve g gradyandır. Ve bundan, gradyan sıfırsa, teta'ya hiçbir şey olmayacağını görebiliriz. Sadece eski değerine güncellenir ve değişmez.



Şekil 43. Sürtünme ifadesi

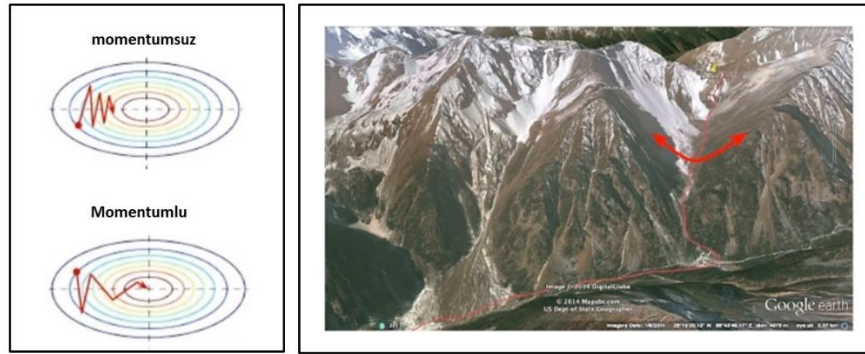
Şimdi momentumlu gradyan iniş bakalım. Gerçek fiziksel momentumla ilgisi olmadığı için burada momentum terimini çok hafif kullanıyoruz. Böylece, hızı temsil eden v adında yeni bir değişken tanımlıyoruz ve $v_t = \mu v_{t-1} - \eta g_t$ ifadesini elde ederiz. Ve bu tam olarak daha önce bahsettiğimiz şey. Buz üzerinde bir kutuyu itmeyi hayal ettiğimizde, g terimi kutuyu itmemizin etkisidir. Ancak μv_{t-1} terimi, daha önce gittiğimiz yönde hareket etmeye devam etmenin etkisidir. Şimdi, bir kutuyu buz üzerinde ittiğinizde, daha az sürtünme olmasına ve kutu kaymasına rağmen, eninde sonunda duracağından bahsettik.

$$\begin{aligned}
 v_t &\leftarrow \mu v_{t-1} - \eta g_t \\
 \theta_t &\leftarrow \theta_{t-1} + v_t
 \end{aligned}$$

"buz üzerinde kayma"
"kutuyu itme"

Şekil 44. Momentumlu gradyan inişi

Bu yüzden hareketin eski v 'nin sadece bir kısmını almasını istiyoruz. Momentum terimi olarak da adlandırdığımız tipik bir μ değeri, o bölgede bir yerde 0.9 veya 0.95 veya 0.99 yaygın olarak kullanılan değerleridir. μ 'yu sıfıra ayarlarsak, normal eski gradyan inişe geri döneceğimizi görmek kolaydır. Peki momentum kullanmanın etkisi nedir? Bildiğiniz gibi, tüm komut dosyalarımızda yinleme başına maliyeti (cost) çizmek yaygındır. Ve genellikle gördüğümüz şey, maliyetin minimum değerine, kullanılmış bir momentuma sahip olduğumuzdan çok daha hızlı yakınsamasıdır. Bu nedenle, momentumu kullanmak oldukça harika çünkü eğitimi önemli ölçüde hızlandırıyor. Şimdi, devam etmeden önce momentum üzerine bir bakış açısını daha tartışmak istiyoruz, bugünlerde genellikle derin öğrenme konusunda öğretilen şey budur. Dolayısıyla bu perspektifte, momentumun çözdüğü problem, farklı yönlerde eşit olmayan eğimlere ya da gradyanlara sahip olmamızdır. Görselleştirme amacıyla, dikey parametreyi ve yatay parametreyi optimize etmek için iki parametremiz olduğunu varsayalım.



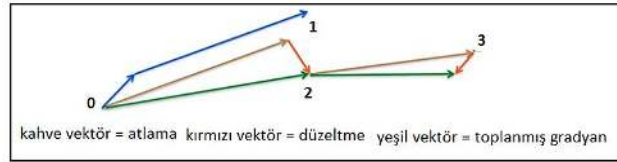
Şekil 45. Momentumlu gradyan inişinin hareketleri

Yani bir yöndeki eğim çok dik ve diğer yöndeki eğim çok sığ. Buradaki fikir, momentumunuz yoksa, o zaman tamamen eğime güvenirsiniz, bu da sığ yönden daha dik yönü işaret eder. Ve bu sadece bir gradyanın özelliğidir ve en dik iniş yönüdür. Bu eğim vektörü daha dik yönü gösterdiğinden, sonuç olarak bu yukarıda gösterildiği gibi vadide ileri

geri zikzak çizeceğiz ve bu minimuma ulaşmak için gerçekten verimsiz bir yoldur. Peki bu duruma ivme eklediğimizde ne olur? sık yönde her seferinde aynı yönde hareket ettiğimiz için bu hızlar birikecek. Böylece, o yönde ilerlememize yardımcı olması için eski hızımızın bir kısmını yeni hızımıza ekleyeceğiz. Sonuç, sık eğim yönünde daha büyük adımlar atarak oraya daha hızlı varmamızdır.

2.5.1.9. Nesterov Momentum

Bu kısımda, Nesterov Momentum adlı momentuma yönelik modern bir gelişmeyi tartışacağız. Tahmin edebileceğiniz gibi, bu Nesterov adında bir adam tarafından icat edildi ve aslında 1983'te "A Method for Unconstrained Convex Minimization Problem" adlı bir makalede yakınsama oranı $O(1/k^2)$ ile yayınlandı. Bu makale, başlığından da anlaşılacağı gibi oldukça tekniktir, ancak neyse ki uygulama basittir. Nesterov momentumunun arkasındaki temel fikir, halihazırda gittiğimiz yönde ilerlemeye devam etmek için momentumu körü körüne kullanmak yerine, önceki hız ile aynı yönde büyük bir sıçrama yaparak ilerleyelim ve gradyanı oradan hesaplayalım. Sonra hızımızı güncellemek için bu eğimi kullanırız. Yani bir bakıma ileriye bakıyor (peek ahead). Ve sonra, elbette, nereye varacağımıza göre düzeltme yapmak. Aşağıdaki görüntüde, standart momentum kullanıyor olsaydık mavi vektörlerin nereye gideceğimizi temsil ettiğini görebiliriz. Kahverengi vektör, tepe değerimizin değerini temsil eder. Mavi vektörle nasıl aynı yönde olduğuna dikkat edelim. Kırmızı vektör, ileriye bakma değerindeki gradyandır. Son olarak, yeşil vektör sadece kahverengi vektör ve kırmızı vektörün vektör toplamıdır.



Şekil 46. Nesterov Momentumlu gradyan inişi

Peki denklemler neye benziyor? İlk önce, bu ileriye atlayışımız (jump ahead) var, buna w'_{t-1} diyelim. $w+v$ ekleyerek tam sıçramayı yapmadığımızı, bunun yerine hızı μ kadar beklediğimize dikkat edelim. Sonra, bu noktadaki gradyanı hesaplıyoruz ve bunu v 'yi güncellemek için kullanıyoruz. Ve son adım, standart momentum, tümü w artı yeni v için olanla aynı olan w için güncellemedir.

Nesterov Momentum

$$\text{look-ahead value : } w'_{t-1} = w_{t-1} + \mu v_{t-1}$$

$$\begin{aligned} v_t &\leftarrow \mu v_{t-1} - \eta \nabla J(w'_{t-1}) \\ &= \mu v_{t-1} - \eta \nabla J(w_{t-1} + \mu v_{t-1}) \end{aligned}$$

$$w_t \leftarrow w_{t-1} + v_t$$

Şekil 47. Nesterov Momentumlu gradyan ifadeleri

İlginç olan, pratikte Nesterov momentumunun genellikle böyle uygulanmamasıdır. Bunun yerine, Nesterov momentumunun biraz yeniden formüle edilmesini yapacağız. Önce, w 'nin ileriye bakma değeri olan w' tanımımıza geri dönelim. Aynı tanıma kullanarak w'_t ve w'_{t-1} 'i tanımlayabilmemiz için gradyanı buradan hesaplamak istiyoruz. t yerine $t-1$ 'i takıyoruz. Başka bir deyişle, bunlar ardışık iki adımda bizim için ileriye dönük değerlerdir. Sonra, v ve w için güncellemelerimize tekrar bakalım, burada yapmak istediğim ilk şey, w_t ve w_{t-1} 'i yalnızca değerlere bakmayı içeren primli ifadelerle değiştirmek.

Yaygın Momentum

$$\begin{aligned} v_t &\leftarrow \mu v_{t-1} - \eta \nabla J(w_{t-1}) \\ w_t &\leftarrow w_{t-1} + v_t \end{aligned}$$

Nesterov momentum

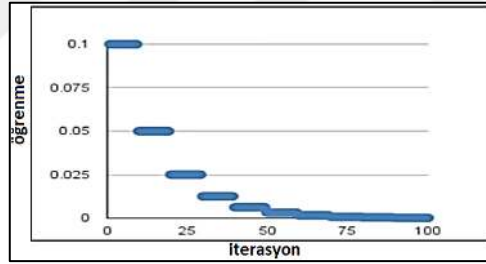
$$\begin{aligned} v_t &\leftarrow \mu v_{t-1} - \eta \nabla J(w_{t-1}) \\ w_t &\leftarrow w_{t-1} + \mu v_t - \eta \nabla J(w_{t-1}) \end{aligned}$$

Şekil 48. Nesterov ve Normal Momentumlular

Şimdiki tek fark, w 'yi güncellemek için kullandığımız denklemin biraz farklı olmasıdır. Yani tüm bunlara kolay bir bakış açısı istiyorsak, sadece şöyle düşünebiliriz, gerçekten zeki bir matematikçinin daha iyi olduğunu kanıtladığı biraz farklı bir momentum denklemi. Bu, uygulama açısından gerçekten güzel, çünkü her iki momentum türü için de pek bir değişiklik yok. Koda baktığımızda göreceğimiz şeyi etkili bir şekilde değiştirmesi gereken gerçekten sadece bir veya iki satır kod. Pratikte, Nesterov momentumu ve düz momentum aslında birbirine oldukça yakındır. Bu nedenle, veri kümelerimiz için neyin işe yaradığını hissetmek için her ikisini de denemeye teşvik edilmemize rağmen, çoğu zaman genellikle düz momentum kullanan insanları görmekteyiz.

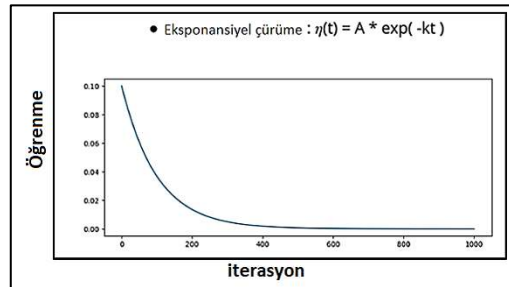
2.5.1.10. Değişken Öğrenme Oranları

Bu kısımda, değişken ve uyarlanabilir öğrenme oranlarını (adaptive learning rates) tartışacağız, şimdiye kadar momentuma, eğitimi büyük ölçüde hızlandırabilecek vanilla geri yayılımına (vanilla backpropagation) bir değişiklik olarak baktık. Pratik çalışmalar açısından, standart bir gradyan inişi için yineleme sonunda momentum ile dradyan inişi karşılaştırıldığında, en etkili olan momentumdur, fark çok büyüktür. Aynı zamanda, momentum güzel çünkü hiper parametrelerle çok fazla oynamak zorunda değilsiniz ve sadece 0.9 ile çalışabiliriz. Aynı zamanda, bu uyarlanabilir öğrenme hızı tekniklerinden bazıları çok güçlüdür. Öyleyse bir bakalım. Bahsedeceğimiz ilk şey, değişken öğrenme oranları veya başka bir deyişle, zaman içinde yinelemenin bir fonksiyonu olarak öğrenme oranlarıdır. Bazen insanlar buna öğrenme oranı planlaması (learning rate scheduling) diyor. Çok basitler, ama bize oynamamız için çok şey veriyorlar. Bu yüzden ilk bakacağımız şeye adım çürüme (decay) denir. Esasen yaptığımız şey, periyodik olarak her 100 adımda bir öğrenme oranını sabit bir faktörle azalttığımızı söylemektir.



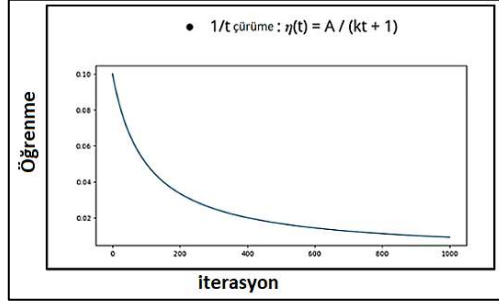
Şekil 49. Öğrenme oranı planlaması

Yani, örneğin, ikiye böleriz veya her seferinde alırız. Eğer onu çizersek, bir çeşit merdivene benzediğini görebiliriz. İkinci yöntem üstel azalma (exponential decay) denir, bu yöntemde öğrenme oranı üstel bir eğri izler.



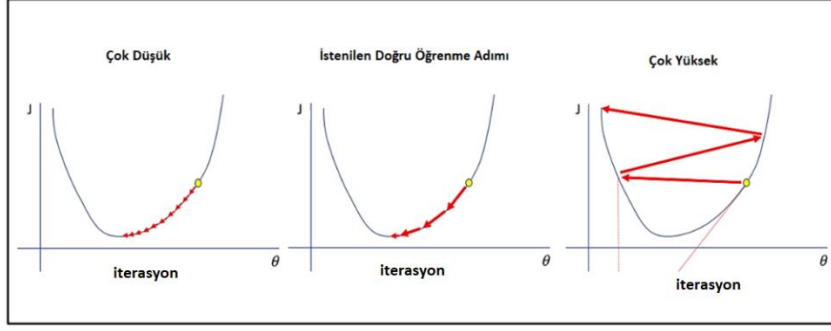
Şekil 50. Üstel azalma öğrenme oranı

Üçüncü yöntem, öğrenme hızının zamanla orantılı olarak bire düşmesi içindir, orantı sabitini değiştirerek öğrenme hızının ne kadar hızlı veya yavaş azalacağını kontrol edebilirsiniz.



Şekil 51. Üstel çürüme öğrenme oranı

Düşüşün (dropoff) üstel çürümeden nasıl daha yavaş olduğuna dikkat edin. Peki, bunların hepsinin zamanla öğrenme oranını azalttığını görebileceğiniz ortak noktası nedir? Bunu neden yapmak isteyelim ki? Genel olarak konuşursak, bir sinir ağının ağırlıklarını rastgele başlattığımızda, optimal ağırlıklardan çok uzak olacaklar. Bu nedenle, hedefe doğru daha büyük adımlar atabilmemiz için büyük bir öğrenme oranıyla başlamak iyidir. Momentumun arkasındaki motivasyon da budur. Geçmiş gradyanları biriktirerek hız kazanmak istiyoruz çünkü biliyoruz ki hedefimizden çok uzaktaysak, o zaman bu gradyanlar büyük olmalıdır. Ama hedefe yaklaştığımızda gradyan küçülecek. Aslında tanım gereği, bir fonksiyonun minimumu sıfır gradyanını gerektirir. Minimum fonksiyon hesabını bu şekilde çözersiniz. Hatırlarsanız türevi bulduktan sonra sıfıra alıyorsunuz ve ardından söz konusu parametreyi çözüyorsunuz. Peki minimuma yaklaştığımızda neden yavaşlamak isteyelim? Asgari seviyeye yaklaştığınızda ve çok büyük bir adım attığınızda, aşırıya kaçacaksınız. Ve sonunda olan şey, sadece ileri geri zıplamanız oluyor. Aslında, eğer öğrenme oranınız çok büyükse, hemen vadiden fırlarsınız ve kaybınız aslında artabilir. Yani tüm bu zıplamayı azaltmak için daha küçük adımlar atmak istiyoruz. İşte bu değişken öğrenme hızı teknikleri ile ilgili insanların kullandığı başka bir teknik.



Şekil 52. Optimal öğrenme adımlarının ayarlaması

Bazen araştırmacılar ve makine öğrenimi uygulayıcıları gerçekten orada oturup sinir ağı eğitim sürecine bebek bakıcılığı yapacaktır. Bu yüzden birkaç epoch iterasyonu yaparak eğitimin nasıl gittiğini görecekler. Öğrenme çok yavaş oluyorsa, öğrenme oranını artırabilirler. Öğrenme eşitlenmeye başlıyorsa, öğrenme oranını düşürebilirler. Ancak dikkatli olmalısınız çünkü güzel monoton bir şekilde azalan bir eğriye sahip olmanız garanti edilmez. Bazen hata yüzeyinin nispeten düz bir bölümünde sıkışıp kalabilirsiniz, ancak bu geçici olabilir. Ve yeterince sabırlıysanız, sonunda dik bir düşüş yaşayabilirsiniz. Ancak aynı zamanda bu garanti edilmez. Makine öğrenimindeki çoğu şeyde olduğu gibi, cevap, davranışın verilere bağlı olduğudur. Kaynaklarınız varsa daha uzun süre beklemek arzu edilir, çünkü o zaman öğrenme sürecinin tam bir resmini elde edersiniz. Bu nedenle manuel öğrenme oranı planlaması da aralarından seçim yapabileceğiniz bir seçenektir. Şimdi, tüm bu yöntemlerin başka ortak noktaları olduğunu düşünmek de önemlidir. Bu yöntemlerin tümü, optimize edilecek şeyler listenize yeni hiper parametreler ekler.

Aslında bu yöntemlerin her biri arasında seçim yapmak bir hiper parametre optimizasyonu olarak da görülebilir. Yani bir şekilde, bu teknikleri bilmek size yardımcı olabilir. Ancak başka bir açıdan, yükünüze daha fazla iş ekler. Bahsetmek istediğimiz sonraki birkaç teknik veya uyarlamalı öğrenme hızı teknikleri olarak adlandırmayı sevdiğimiz şey, çünkü bunlar şimdiye kadar gördüğünüz eğitim verilerine uyum sağlıyor. Tartışmak istediğimiz ilki AdaGrad olarak adlandırılıyor, temel fikir şu ki, maliyetlerin her bir parametreye olan bağımlılığının aynı olmasını bekleyemeyiz. Başka bir deyişle, bir yönde eğim gerçekten dik olabilir, ancak başka bir yönde eğim gerçekten düz olabilir. Bu nedenle, geçmişte ne kadar değiştiğine bağlı olarak her bir parametre için öğrenme oranını ayrı ayrı uyarlamak faydalı olabilir. AdaGrad'da yaptığımız şey, bellek (cache) adı verilen bir değişkeni tanıtmak. Sinir ağının her parametresinin kendi ön belleği vardır. Örneğin, üçe

dört boyutunda tek yönlü bir matrisiniz varsa, o zaman üçe dört boyutunda bir önbellek matrisiniz de olur. Aynı şey bias vektörlerimiz için de geçerli. Cache arkasındaki fikir, eğimlerin karesini aldığımız için kare eğimleri biriktirecek olmasıdır. Cache her zaman pozitif olacaktır. Ve her parametrenin kendi belleği olduğu için, o zaman eğer bir parametre geçmişte çok büyük gradyanlara sahipse, o zaman belleği çok büyük olacaktır ve etkili öğrenme oranı çok küçük olacaktır. Bu yüzden gelecekte daha yavaş değişecektir. Öte yandan, eğer bir parametre geçmişte çok sayıda küçük eğime sahipse, o zaman belleği küçük olacaktır. Bu nedenle, etkili öğrenme oranı yüksek kalacak ve gelecekte değişmek için daha fazla fırsatı olacaktır. Küçük bir ayrıntı, sıfıra bölmeyi önlemek için paydaya genellikle küçük bir epsilon sayısı eklememizdir.

$$\begin{aligned} \text{Bellek} &= \text{Bellek} + \text{Gradyan}^2 \\ \theta &\leftarrow \theta - \eta \frac{\nabla J}{\sqrt{\text{Bellek} + \epsilon}} \end{aligned}$$

Şekil 53. Geçmiş gradyanların önbellegi

Tipik olarak, bu on üzeri eksi sekiz veya 10 üzeri eksi 10 olarak ayarlanır. Üzerinde mutabık kalınan vurgulanması gereken önemli bir nokta, yaptığımız her şeyin öğeye dayalı bir işlem (element wise operation) olduğudur, bu nedenle her ölçek büyütme parametresi diğerlerinden bağımsız olarak etkin bir şekilde güncellenir, bu da mantıklıdır. Böylece tüm parametreler için geçerli olan skaler güncellemeler olarak sunduğumuz formüllere bakabilirsiniz. Veya tüm sinir ağı parametrelerini içeren devasa bir parametre vektörü düşünebilirsiniz. Ve sonra işlemlerin her biri bir y ögesinin işlemidir. Bu sonraki teknik, araştırmacıların, üzerinde anlaşmaya varılan aslında daha yapılacak daha çok öğrenme varken, öğrenme oranını agresif bir şekilde azalttığını ve dolayısıyla öğrenme oranının çok hızlı bir şekilde sıfıra yaklaşacağını gözlemledikleri gerçeğine dayanıyor. Bu tekniğe RMSProp denir ve Jeff Hinton ve ekibi tarafından icat edilmiştir, bu nedenle çalışma şekli, öğrenme oranını çok hızlı düşürmesinin nedeni, önbellegin çok hızlı büyümesidir. Bu nedenle, önbellegin daha az hızlı büyümesini sağlamak için, onu her güncellediğimizde aslında azaltıyoruz.

Bunu, eski önbellegin ve yeni kare gradyanın ağırlıklı ortalamasını alarak yapıyoruz. Buna k oranı diyoruz ve iki ağırlığın toplamının bir olduğunu görebilirsiniz. Konu boyunca

bu forma tekrar tekrar bakacağız, bu yüzden şu anda bilmemiz gereken her şeyi bilmek konusunda endişelenmeyelim, sadece yeni önbelleği almak için bunları bir araya ekleyerek eski önbelleğin ve yeni kare geçişin bir parçası olacağımızı bilelim. Çürüme oranı için tipik değerler 0.99 ve 0.999 vb. Yine, bu seçimlerin sezgisi daha sonraki kısımlarda tartışılacaktır. Bunu yaparak, önbelleği sızdırdığımızı söylüyoruz ve sızdırmasının nedeni, uzun bir süre boyunca sıfır eğime sahip olsaydık, önbelleğin eninde sonunda sıfıra ineceğini, çünkü her turda çürüme oranı ile azaltılacağını hayal edebiliyorsunuzdur.

2.5.1.11. Adaptif Öğrenme Oranları

Bu kısımda, ve 2.5.7. ve 2.5.8. kısımlarda uyarlamalı moment tahmini veya kısaca Adam adı verilen bir teknik kullanarak gradyan inişini değiştirmenin başka bir yolunu inceleyeceğiz. Adam kendi yaklaşımını hak eden bir tekniktir çünkü günümüzde derin sinir ağlarını optimize etme yöntemidir. Yani, insanlar, diğer seçenekleri gerçekten dikkate almadan, genellikle Adam'ı varsayılan seçenek olarak kullanırlar. Bunun nedeni Adam'ın öğrenme hızı, momentum parametresi vb. için varsayılan ayarlarla iyi çalışma eğilimindedir. Başka bir deyişle, Adam sağlamdır (robust). Bunu kullanabilir ve tam olarak doğru olan değerleri bulmak için parametreleri değiştirme konusunda endişelenmemize gerek kalmaz, çünkü varsayılan olarak sahip olduğumuz şey muhtemelen tamamdır. Ek olarak, Adam algoritması bir şekilde RMSprop'un halefidir. Her ikisi de Toronto Üniversitesi'nde geliştirildi RMSprop Geoffrey Hinton tarafından, Adam ise o sırada Hinton'ın doktora öğrencisi olan Jimmy Ba tarafından geliştirildi. Aslında, Jimmy Ba'nın üç doktora danışmanı vardı ve hepsi önde gelen makine öğrenimi araştırmacılarıydı ve şimdi kendisi bir profesör. Şimdi, Adam'ı düşünmenin bir yolu, RMSprop'unun momentumlu olmasıdır.

Bunun nedeni, uyarlanabilir bir öğrenme oranına sahip olma fikrini alması, ancak aynı zamanda ivme katmasıdır. Bu biraz kafa karıştırıcı olabilir çünkü TensorFlow gibi kütüphaneler bize RMSprop'a ivme ekleme seçeneği de sunar. Ancak bunlar biraz farklıdır. Yani kilit nokta, Adam'ın momentumlu RMSprop'a benzemesi, ancak tam olarak aynı değil. Şimdiye kadar gördüklerimizi gözden geçirerek başlayalım. Spesifik olarak, vanilya gradyan iniş yöntemini geliştirebileceğimiz iki yararlı yolu ele alalım. Birincisi momentum kullanmaktır. Hareket halindeki bir cisim hareket halinde kalır. İkinci yöntem, RMSprop gibi uyarlanabilir öğrenme hızı yöntemlerini kullanmaktır. Muhtemelen alışık olduğumuzdan biraz farklı semboller kullandığımızı unutmayalım. Bu sadece bizi,

geleneksel olarak farklı semboller kullanan Adam'ı öğrenmeye hazırlamak içindir. Momentum için parametre güncellemesi iki adımda ilerler.

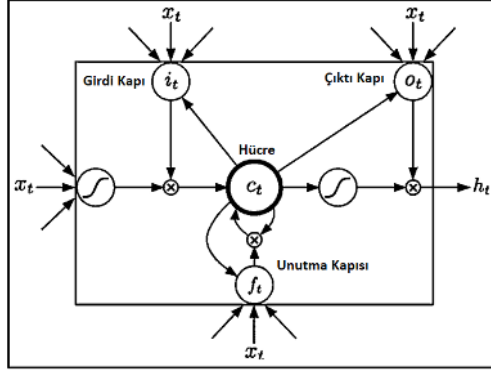
Momentumun önceki ifade şekli	Momentumun alternatif gösterimi
$m_t = \mu m_{t-1} - \eta g_t$ $\theta_t = \theta_{t-1} + m_t$	$m_t = \beta m_{t-1} + (1 - \beta)g_t$ $\theta_t = \theta_{t-1} - \eta m_t$

Şekil 54. Momentumlu Gradyan inişi notasyonları

İlk olarak, burda M olarak adlandıracağımız momentum değerini güncelliyoruz. Önceki momentumun bir oranına eşittir, eksi öğrenme oranı çarpı mevcut gradyan. Daha sonra θ dediğimiz parametreleri kendimiz güncelleriz. Güncelleme, teta'nın sadece eski değeri artı yeni momentumdur. Bu kısım için, momentumu biraz farklı bir şekilde sunmak faydalı olacaktır. Spesifik olarak, şimdi momentum değerini öncekinin bir tür negatifi olarak ele alıyoruz, m_t güncellemesi artık önceki m_{t-1} 'in bazı ağırlıklı ve θ güncellemek için mevcut gradyan g_t 'dir, ve onları öğrenme oranı η 'nin t katından çıkarıyoruz. Şimdi, tüm bu yeni semboller sadece sabitler olduğundan, momentumu sunmanın iki yönteminin eşdeğer olduğunu gösterebilirsiniz.

2.5.1. LSTM Mimarisi

Bu tezde makalelerde ortak kriter (benchmark) olarak kullanılan Uzun kısa süreli bellek (LSTM) mimarisi kullanılmıştır. MOYİ'de, ev aletlerinin yük sinyallerinde rastgele açma/kapama davranışı nedeniyle tüm cihazlar için yük imzası tanımlanır ve zaman serilerinin olayları olarak düşünülebilir. Bu nedenle, MOYİ için Tekrarlayan Sinir Ağında (RNN) LSTM'yi kullanıyoruz çünkü kaybolan gradyan probleminden kaçınırken uzun bir belleği idare edebiliyor.



Şekil 55. LSTM ağının mimarisi

Şekil 55, bir LSTM bellek hücrecini göstermektedir. Ağ tarafından uygulanan işlevler şunlardır:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (2)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (3)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (4)$$

$$h_t = o_t \tanh(c_t) \quad (5)$$

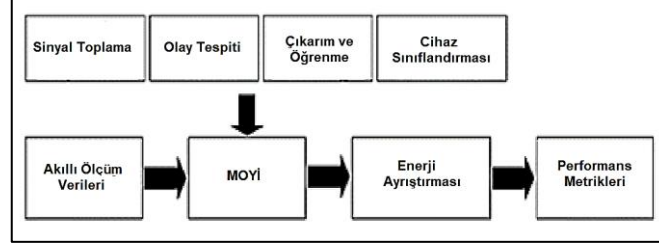
W_{ci} , W_{cf} , W_{co} , köşegen ağırlık matrislerinin gözetleme deliği bağlantılarıdır. τ ve θ hücre girdisinin ve hücre çıktısının aktivasyon işlevleridir ve i , girdi geçididir. σ sigmoid fonksiyonudur ve f unutma kapısıdır, o çıkış kapısıdır, a hücre giriş aktivasyonudur ve c hücre durum vektörleridir.

2.5.2. MOYİ Geçmişi

Bir Akıllı Sayaç verildiğinde, $T = \{1, 2, 3, \dots, t\}$ zamanı için toplam güç tüketimi serisi $P = \{p1, p2, p3, \dots, pt\}$, MOYİ problemi aşağıdaki gibi formüle edilebilir:

$$PT = \sum_{i=1}^M y_t^{(i)} + \sigma(t) \quad (6)$$

$\Sigma(t)$ hesaplanmamış gürültüyü gösterdiğinde, M aktif cihazların ev aletinin tüketim gücü $i \in \{1, 2, 3, \dots, M\}$ 'yi anlamayı amaçlamaktadır.

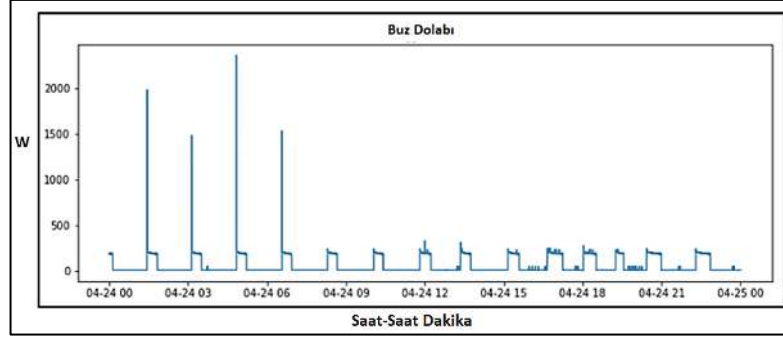


Şekil 56. MOYİ sürecinin blok diyagramı

Bir MOYİ sistemi, Şekil 56'de gösterildiği gibi dört aşama içerir, Ön işleme ile sinyal gücü toplama, özellik çıkarımı ile olay algılama, öğrenme ile çıkarım ve cihaz sınıflandırması. Enerji ayırıştırmanın ilk aşaması, sinyal gücü elde etmektir ve görevi, çeşitli örnekleme oranlarında toplu yük ölçümleri elde etmektir. Olay Algılama ve Özellik Çıkarma aşaması, geçici durumun veya sabit durum (steady state) durumunun önceden işlenmiş güç ölçümlerinde farklılık göstermesinin ayarlanmasını içerir. Çıkarılan olaylarla ilgili özellikler, her bir cihaz faaliyetiyle ilgili benzersiz tüketim kalıplarıdır. Öğrenme ve Çıkarım aşamasında, aletleri tanımlamak için temel denetimli/denetimsiz teknikler uygulandı. Son aşama olarak, sınıflandırma cihazları, toplam toplu kayıtların güç tüketimine ve bu cihaz durumuna ilişkin bireysel cihaz durumlarına bölünmesinden oluşur [59].

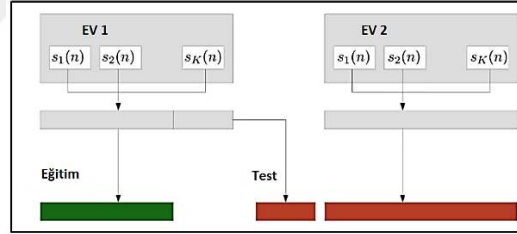
2.5.3. MOYİ Veri Kümesi

Bu çalışmada, enerji ayırıştırması için yaygın olarak kullanılan Referans Enerji Ayırıştırma Veri Kümesi (REDD) [60] veri kümesi kullanılmıştır. MOYİ veri kümesi, evler veya binalar gibi gerçek dünya ortamından ölçülen güç tüketimi verilerini içeren halka açıktır. Veri kümesi, akıllı ölçüm verilerini içerir ve ayrı bir cihaz temel gerçeğine (ground truth), veri kümesi hedeflerine dayalı güç tüketim verilerine sahip olabilir. MOYİ algoritmalarının performanslarını değerlendirmek için, ayırıştırmanın gerçekleştirildiği bir cihaz hakkında temel gerçeklere sahip olmak gerekir. Şekil 57, REDD evi 1'in 2011.04.24 gününün buzdolabı grafiğidir.



Şekil 57. REDD veri setindeki 24.04.2011 tarihine ilişkin buzdolabı gününün zamana göre değişimi

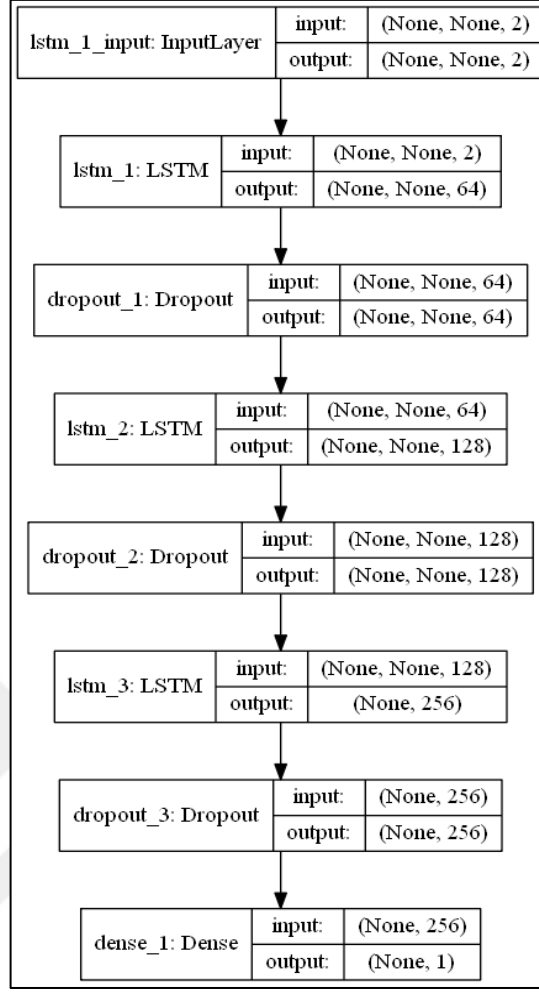
Şekil 58, eğitim ve test için bölünmüş verileri göstermektedir. Buzdolabı, MOYİ'nin LSTM ağını eğitmek için bilinen bir cihaz verisi olarak seçildi. Birinci evde, veriler, verilerin eğitimi ve test edilmesinin iki bölümüne bölündü. Eğitim verileri, 17 günlük güç verisi ve altı günlük test verilerini içerir. Enerji ayrıştırıcı ağını eğittikten sonra, bilinmeyen bir cihaz olan MOYİ ağına performans verimliliğini doğrulamak için ev 2 verilerini bilinmeyen bir veri olarak kullandık.



Şekil 58. Veri Kümesi Ayrımı

2.5.4. Cihaz Sınıflandırıcısının LSTM Model Yapısı

Keras with Tensor Flow [61] arka ucu kullanılarak uygulanan önerilen LSTM modeli Şekil 59'te gösterilmektedir. Ağ, bir bırakma (dropout) katmanıyla ayrılmış ve yoğun (dense) bir katmanla sonlandırılmış sekiz katmandan oluşur. Model yapısında, her LSTM'nin gizli 64, 128 ve 256 seviyeleri vardır. Ek olarak, bırakma katmanında, her katmanın düşme oranı değeri 0,2'dir.



Şekil 59. LSTM katmanlarla gösterimi

2.5.5. Hata Fonksiyonu

Bu tür modelleri değerlendirmenin bir başka parametresi de kayıp veya hata fonksiyonudur. Tahmin çıktısının gerçek hedef sonuçlardan çok farklı olması durumunda, hedef sonuçları optimize etmek için kayıp işlevi kullanılacaktır. Kayıp işlevi, optimizasyon algoritması uygulandığında tahminlerdeki hatayı en aza indirmeyi öğrenir. Bu çalışmada, hataları azaltmak için tahmin edilen ve hedef sonuçlar arasındaki mesafelerin karesi toplamı olan kayıp fonksiyonunu ve Ortalama Kare Hata'yı (MSE) [62] kullanıyoruz.

2.5.6. Optimizasyon Fonksiyonu

Pek çok derin öğrenme çalışmasında, optimizasyon işlevi olarak son teknoloji ürünü Uyarlanabilir Moment Tahmini (Adam) algoritmasını kullanır. Adam, eğitim verilerine göre

güncellenen yinelemeli sinir ağırlıkları sırasında Gradyan İnişini (GD) iyileştirmek için kullanılan bir optimizasyon yöntemidir. Adam, iyi sonuçları hızlı bir şekilde gerçekleştirebildiği ve hesaplamayı verimli bir şekilde artırabildiği için derin öğrenme için en sevilen algoritmadır [63]. Adam, birçok Makine Öğrenimi yaklaşımında hızlı yakınsama göstermesine rağmen, bu çalışma AdaX adlı yeni bir uyarlanabilir (adaptif) gradyan inişiyle Adam'ı geliştirmeyi amaçlamaktadır. Adam geçmiş gradyanları görmezden gelir, ancak AdaX, öğrenme oranını uyarlamalı olarak ayarlayarak (tuning) eğitim yaparken geçmişte uzun vadeli gradyan verilerini katlanarak biriktirir. Bu tezde AdaX'in tasarlanan LSTM ayrıştırıcı modelinin ayarlarında daha iyi yakınsamasını kanıtlayacağız.

2.5.7. ADAM Optimizasyon Algoritması

Adam [64] optimize edici, AdaDelta ve RMSprop [65] gibi önceki gradyan toplamalarının v_t üssel ortalama düşüşünü depolayarak verilere dayanarak öğrenme hızını (LR) hesaplar, bu hızlanma tekniği gibi m_t gradyanlarının üssel ortalama azalmasını tutar:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (7)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (8)$$

Burada, m_t ve v_t , sırasıyla gradyanların merkezi olmayan varyansı olarak ortalama ve ikinci moment olarak ilk an tahminleridir. M_t ve v_t vektörlerinin birincil durumu sıfır olduğundan, geliştiriciler sonuçların sıfıra doğru eğildiğini buldular. Bu, küçük azaltma oranına sahip ilk adımlarda daha belirgindi (yani, β_1 ve β_2 1'e yakındı). Sorunu çözmek için, birinci ve ikinci tahminlerin düzeltilmiş tahminlerini kullandılar:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (9)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (10)$$

“AdaDelta” ve “RMSprop” gibi bu iki formül, parametrelerdeki değişiklikleri hesaplamak için kullanıldı ve elde edilen değişiklik formülü aşağıdaki gibidir:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \quad (11)$$

β_1 , β_2 ve ϵ için önerilen varsayılan değerler sırasıyla 0.9, 0.999 ve 10^{-8} 'dir. Geliştiriciler deneysel olarak Adam yönteminin iyi bir performansa sahip olduğunu ve diğer uyarlanabilir öğrenme yöntemlerinden daha iyi performans gösterdiğini gösterdiler.

2.5.8. AdaX Optimizasyon Algoritması

Bu çalışma, uyarlanabilir öğrenme oranını ayarlayan yeni bir LSTM enerji ayırıştırıcısını tanıtmaktadır. Li ve ark. [66], küçük gradyanlar kararsız ikinci an üretebilir ve bu geçmiş bellek AMSGrad'daki maksimum işlem gibi vurgulanmalıdır [67]. Ayrıca, eğimlerde üssel azalmayı önlemek için vurgulanan işlem göreve bağlı olmamalıdır. Adam algoritmasının tersine, önerilen MOYİ-AdaX ayarlamasında, Li ve diğerleri tarafından Algoritma 2'de sunulduğu gibi, geçmiş gradyanların katlanarak daha fazla tartılması ve mevcut gradyanların uyarlamalı olarak kademeli olarak azaltılmasıyla gerçekleştirildi. AdaX ve Adam arasındaki en önemli farklar, üstel hareketli ortalama kullanmak yerine tasarımı $(\beta_2, 1 - \beta_2)$ $(1 + \beta_2, \beta_2)$ olarak değiştirildiği altıncı ve yedinci satırlardadır.

AdaX yöntemindeki algoritma 2

1: **Girdi:** $x \in F, \{\alpha_t\}_{t=1}^T$ $(\beta_1, \beta_2) = (0.9, 10^{-4})$

2: **Başlatma** $m_0 = 0, v_0 = 0$

3: **for** $t = 1$ **to** T **do**

4: $g_t = \nabla f_t(x_t)$

5: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$

6: $v_t = (1 + \beta_2) v_{t-1} + \beta_2 g_t^2$

7: $\hat{v}_t = v_t / [(1 + \beta_2) t - 1]$ and $V_t = \text{diag}(\hat{v}_t)$

8: $x_{t+1} = \Pi_F, \sqrt{V_t} (x_t - \alpha_t m_t / \sqrt{\hat{v}_t})$

9: **for döngüsünün bitişi**

Altıncı satır, geçmiş gradyanları birden büyük bir sabitle çarpılarak gösterir, bu da geçmiş verilerin gözden kaçırılmak yerine biriktiği anlamına gelir. Her g_t^2 küçük bir sayı ile çarpılır ve geçmiş belleğe eklenir. Algoritmanın arkasındaki fikir, parametreler θ ve diğerleri tarafından sunulan sentetik model gibi optimum değerlere yakın kalırken, ikinci anı en yeni gradyanlara uyarlamalı olarak kademeli olarak azaltmaktır. Mevcut çalışmada, önerilen tasarım, νt büyük tutulduğunda güncelleme adımlarında küçük gradyanların karışmamasını sağlar. Buna göre, önyargı düzeltme terimiyle, önerilen νt aşamalı olarak büyük ve istikrarlı hale gelecektir. Yedinci satırda, ikinci anın tarafsız bir tahminine ulaşmak için, νt , önyargı düzeltme terimine bölünmüştür. Kingma ve Ba [68] 'nın türetilmesi gibi, t zaman adımında g_t gradyanı ve $g_t \sim p(g_t)$ sabit dağılımından çekilmiş olduğu varsayılan s. Algoritma 2'deki altıncı satırın taraflarına göre,

$$E(v_t) = \sum_{i=1}^t = (1 + \beta_2)^{t-i} \beta_2 E(g_t^2) [(1 + \beta_2)^t - 1] E(g_t^2) \quad (12)$$

Dolayısıyla, ikinci bir anı doğru tutmak için νt , yedinci satırda $(1 + \beta_2)^t - 1$ 'e bölünmüştür. Bununla birlikte, Kingma ve Ba'da, ikinci neden olarak bir ilk an düzeltmesi terimi $(1 - \beta_1^t)$ içermeyen değerli atıfta bulunmaktadır. Stokastik Gradyan İnişindeki (SGDM) momentum ve Adam'ın ilk anı ile ilgili olarak,

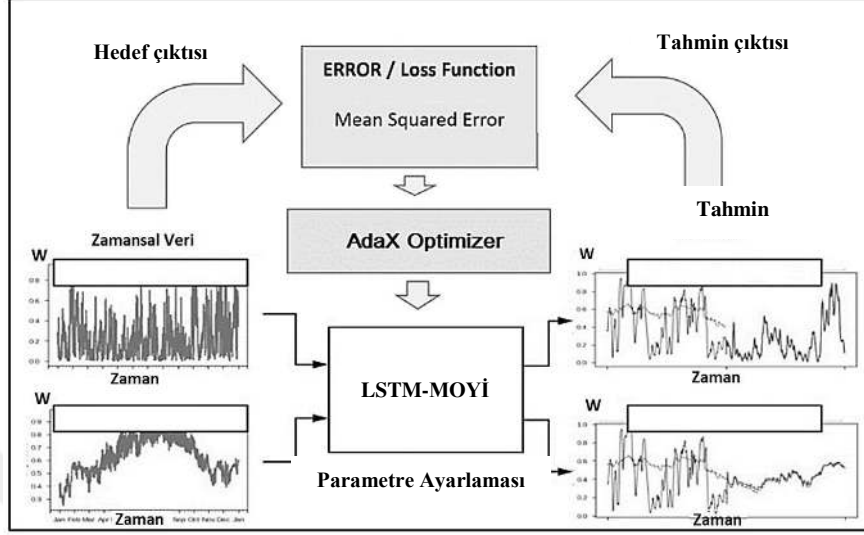
$$\text{SGDM: } m_t = \gamma m_{t-1} + g_t = \sum_{i=1}^t \gamma^{t-i} g_i \quad (13)$$

$$\text{Adam: } m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t = (1 - \beta_1) \sum_{i=1}^t \beta_1^{t-i} g_i \quad (14)$$

2.5.9. LSTM Sınıflandırıcının Öğrenme Yapısı

Ele aldığımız birinci MOYİ modelinde, minimum tahmin hatasını hesaplayarak bir sekanslı LSTM mimarisi ve verilen eğitim verileri için yinelemeli süreci kullanıldı. Hata genellikle modelin öngördüğü çıktılar ile eğitim verilerinin bir parçası olarak verilen gerçek/hedef çıktı arasındaki fark olarak ifade edilir. Şekil 60'te gösterildiği gibi genel bir taslakta, önerilen enerji ayrıştırıcı, bir Kayıp Fonksiyonu, bir optimizasyon yöntemi ve bir LSTM-MOYİ modelinden oluşur. Çimen ve ark. Tarafından sunulan kayıp fonksiyonu. (15)

ifadesi, bölüm 2.5.7.'de açıklanan optimizasyon algoritması ve Şekil 59'te gösterilen LSTM mimarisi tasarımı.



Şekil 60. AdaX ile LSTM-MOYI'nin Parametre Ayarlaması

2.5.10. Model Başarı Metrikleri

Model değerlendirme metrikleri [69], model performansının nicelleştirilmesine ihtiyaç duymaktadır. Bu çalışmada, Kök Ortalama Karekök Hata (MSE) ve Doğruluk kullandık. RMSE metriğinde, doğru birim terimlerinde mutlak yorumlanabilirlik gerekiyorsa, MSE'nin karekökünü alabilir ve RMSE ile bitirebiliriz. RMSE şu şekilde tanımlanmıştır:

$$RMSE = \sqrt{MSE} = \sqrt{1/n \sum_{i=0}^{n-1} (y - y_i^{\text{hat}})^2} \quad (15)$$

MOYİ yöntemlerini değerlendirmek için farklı metrikler kullanılabilir, bu da yük izleme için farklı yöntemler ve algoritmalar temelinde değerlendirmeleri karşılaştırmayı zorlaştırır. İlk başta, algoritmalar iki mod (açık/kapalı) ile uygulandığında, değerlendirme kriterleri, toplam kullanılan güçte dikkate değer bir değişiklik için doğru yük sınıflandırmalarının yüzdesine dayanmaktadır. Bu amaçla kullanılan çeşitli kriterler vardır ve bunları tanıtmadan önce aşağıdaki değişkenlerin tanımlanması gerekir:

TP (toplam gerçek pozitif sayısı): hem cihaz hem de gerçek durum (ground truth) açıkken.

FP (toplam sahte pozitif sayısı): cihaz açıkken ve gerçek durum kapalıyken.

TN (toplam gerçek negatif sayısı): hem cihaz hem de gerçek durum kapalı olduğunda.

FN (toplam sahte negatif sayısı): cihaz kapalıyken ve gerçek durum açıkken.

P: gerçek durumla ilgili toplam pozitif sayısı.

N: gerçek durum üzerindeki toplam negatif sayısı. Doğruluk: Her durumda gerçek sonuçların oranı.

$$\text{Doğruluk} = (TP+TN)/(P+N) \quad (16)$$

Kesinlik (precision) ve sınıflandırıcı ile ilgili olarak, bu ölçüm, bir cihaza atanan toplam enerjinin yüzde kaçının o cihaz tarafından gerçekten kullanıldığını gösterir.

$$\text{Kesinlik} = TP/(TP+FP) \quad (17)$$

F1 skoru, uyumlaştırılmış ortalama doğruluk ve hatırlamadır.

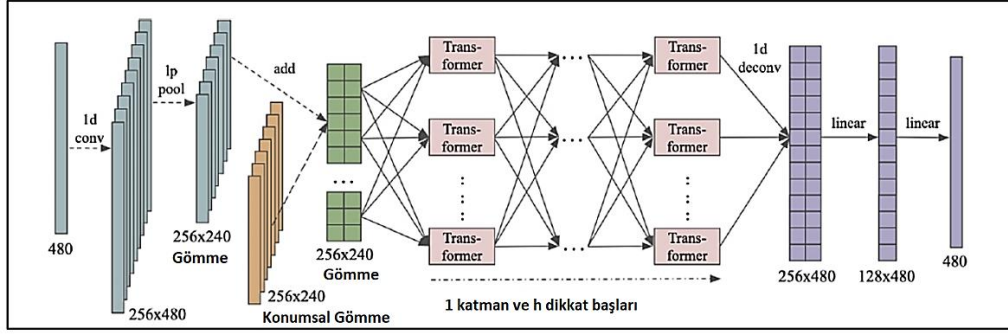
$$F1 = 2 \times (\text{doğruluk} \times \text{duyarlılık}) / (\text{doğruluk} + \text{duyarlılık}) \quad (18)$$

Sınıflandırıcı ile ilgili duyarlılık (Recall), doğru sınıflandırılmış ve ölçülen enerjinin bir parçasıdır.

$$\text{Duyarlılık} = (TP)/(TP+FN) \quad (19)$$

2.5.11. Önerilen BERT-MOYİ Yönteminin Mimarisi

Şekil 61'de gösterildiği gibi, önerilen BERT [70] mimarisi bir gömülü (embedded module) modülünden, transformer katmanlarından ve çok katmanlı algılayıcı çıktı katmanından (MLP) oluşur. Ağ, aynı şekil çıktısına sahip bireysel ev aletlerini kategorize etmek için sabit uzunlukta sekanslı (sequencial) verilerle sağlandı. Ayrıca elektrikli ev aletlerinin karşılaştırılması ile eşik değerleri hesaplanmıştır. Giriş verilerini transformer bloklarına beslemeden önce, tek boyutlu giriş dizisinin gizli boyutunu artırmak için bir evrişim (convolutional) katmanı adapte ederek özellikler (features) ilk olarak mimariden çıkarılır.



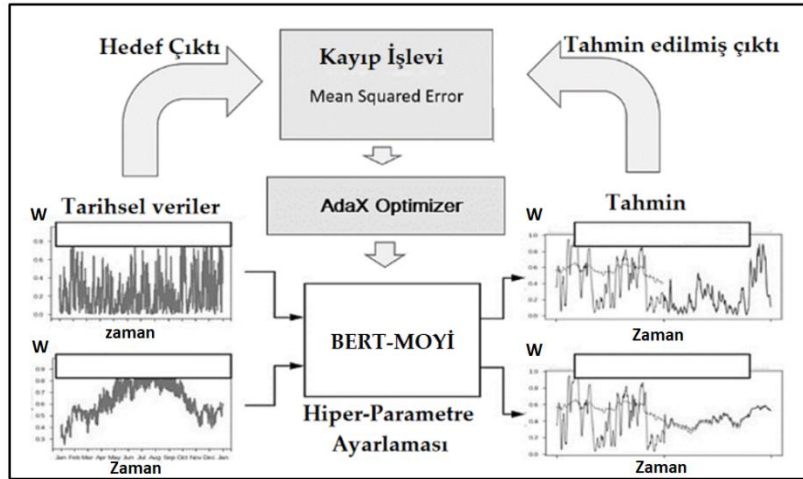
Şekil 61. BERT yapay zeka modelinin mimarisi

Daha sonra öğrenilen bir $L2$ norm havuzlama (pooling) işlemi kullanarak artan gizli boyutla evrimsel çıktı havuzlanır. İşlem, uzunluğu yarı yarıya azaltırken özellikleri daha iyi korumak için giriş dizisi üzerinde kare ortalama havuzlamayı zorlar. Ardından havuzlanmış girdiyi öğrenilebilir bir konumsal gömme (positional embedding) matrisine eklenir. Matris, dizi konumsal kodlamayı hesaba katar, burada $Embedding(X) = LPPooling(conv(X)) + epose$. Ayrıca, çift yönlü transformere nihai gömme matrisini sağlayarak analizi gerçekleştirilir. Transformer, l transformer katmanından ve her katmanda h dikkat (attention) başlığından oluşur. Tek başlı öz-dikkat (self-attention) (ölçeklendirilmiş nokta-çarpım dikkati) Q (query), K (Key) ve V (value) matrisleriyle (giriş matrisinin doğrusal transformer ile elde edilir) formüle edilebilir. Q ve K (başlangıçta gizli boyutun kareköküyle çarpılır ve bölünür), V ile çarpılmadan önce yumuşak dikkat oluşturmak için bir softmax işlemiyle işlenir ve ağırlıklı bir değer matrisi elde edilir. Benzer şekilde, çok kafalı dikkat (multi-head attention), gizli uzayı, aynı hesaplamayı gerçekleştiren parametre matrisleri ile birden çok alt uzaya böler ve birkaç Q , K ve V matrisiyle sonuçlanır. Bireysel dikkat ile faktörlerin her biri farklı alt uzaylardan bilgi alabilir. Sonuçlar, dikkat $(Q) = softmax(QKT/\sqrt{dk})$ V olan dikkatli çıktıyı oluşturmak için entegre edilir ve dönüştürülür.

Çoklu kafa $(Q, K, V) = concat(head_1, head_2, ..., head_h)$ WO, burada $head_i = dikkat(QWQ_i, KWK_i, VWV_i)$. Ek olarak, her transformer katmanındaki çok kafalı dikkatten sonra önceki matris ile konum bazında ileri beslemeli bir ağ (PFFN) sağlanır. Katmanın, girdi öğelerini doğrusal dönüşümler ve Gauss hatası doğrusal birimi (GELU) aktivasyonu ile işlediğine dikkat edilmelidir. UC Berkeley'den Dan Hendrycks ve Kevin Gimpel, GELU aktivasyon fonksiyonunu 2018'de Chicago'daki Toyota Teknoloji Enstitüsü'nden tanıttı [71]. Bir aktivasyon işlevi, nöron çıkışını tetikleyen “anahtar”dır ve ağlar derinleştikçe önemi artmıştır. Giriş özelliklerini korumak için dikkat ve ileri besleme modüllerini takiben artık

bağlantılar uygulanır. Ardından, çeşitli katmanlar arasındaki gizli durum dinamiklerini stabilize etmek için katman normalizasyonu (LayerNorm) gerçekleştirilir. İşlem şu şekilde formüle edilebilir: $LayerNorm(x + Dropout(Modül(x)))$, $PFFN(X) = GELU(0, XW1 + b1)W2 + b2$. Değerleri transformer katmanlarından geçirdikten sonra, bir ters evrişim (deconvolutional) katmanı ve iki doğrusal katman dahil olmak üzere MLP çıktısı bulunabilir. Ters evrişim katman, ilk önce çıktıyı transpose edilmiş evrişim ile önceki uzunluğuna geliştirir. Daha sonra, aralarında Tanh aktivasyonu olan iki katmanlı bir MLP, girdinin gizli boyutunu istenen çıktı boyutuna geri getirir. Çıkış değerleri (tercihen $[0, 1]$ aralığında) maksimum cihaz gücü ile çarpılır ve daha sonra $Ou(X) = Tanh(Deconv)$ eşiklerine karşılık gelen eşleşerek cihazın durumunu elde ederken, makul enerji tahmini oluşturmak için güvence altına alınır. $(X)W1 + b1)W2 + b2$.

Önerilen BERT-MOYİ modeli, verilen eğitim verileri için yinelemeli bir süreç kullanarak minimum tahmin hatasını hesaplamak için yarı denetimli bir öğrenme yöntemine sahiptir. Hata genellikle, eğitim verilerinin bir parçası olarak verilen model tahmini çıktısı ile gerçek/hedef çıktı arasındaki fark olarak ifade edilir. Önerilen ayrıştırıcı, bir kayıp fonksiyonu, bir optimizasyon yöntemi ve BERT-MOYİ modelinden oluşur. Çimen ve diğerleri tarafından önerilen kayıp fonksiyonu sunulur, optimizasyon algoritması Bölüm 2.5.7. ve 2.5.8.'de açıklanmıştır ve tasarlanan BERT yönteminin mimarisi Şekil 62'da gösterilmektedir.



Şekil 62. AdaX optimizasyon algoritması ile BERT-MOYİ yöntemine ilişkin Parametre Ayarlaması

3. BULGULAR

Tez çalışmasında önerilen BERT Transformer ile Sekans (Sequencial) modellerinin farkını ortaya koymak için önce sekans modellerinden bir LSTM mimarisi ve başarı incelemesi ele alınmıştır. RNN ve LSTM gibi sekans modellerinin problemleri şöyledir: Gerçek dünyada, girdi ve çıktı uzunlukları oldukça farklı olabilir. Basit sekans modelleri ise yalnızca sabit uzunluklu problemleri çözebilir. Bu sekanslı sinir ağları kademeli unutkanlıktan muzdariptir, bu da diziler yeterince uzun olduğunda model eğitiminin son derece zorlaştığı anlamına gelir.

Uyarlanabilir öğrenme oranının (LR) Sinir Ağı performansı üzerindeki bir etkisi olarak, Stokastik gradyan inişi yani SGD olarak bilinen performansını ilk olarak önerilen LSTM-MOYİ enerji ayrıştırma modelinde ADAM ve AdaX ile karşılaştırarak değerlendirdik. Daha sonra Adam [72], özellikle derin sinir ağların eğitimi için geliştirilmiş en yeni optimizasyon algoritması olarak değerlendirildi ve son olarak AdaX'in LSTM-MOYİ ayrıştırma probleminde aralarında en iyi performansa sahip olduğunu değerlendirdik. Verimli Geri Yayım (Backpropagation) bağlamında, LR, sinir ağının model ağırlıkları her güncellendiğinde tahmin edilen hataya yanıt olarak sinir ağı modelinin ne kadar değiştirileceğini kontrol eden bir hiper-parametredir.

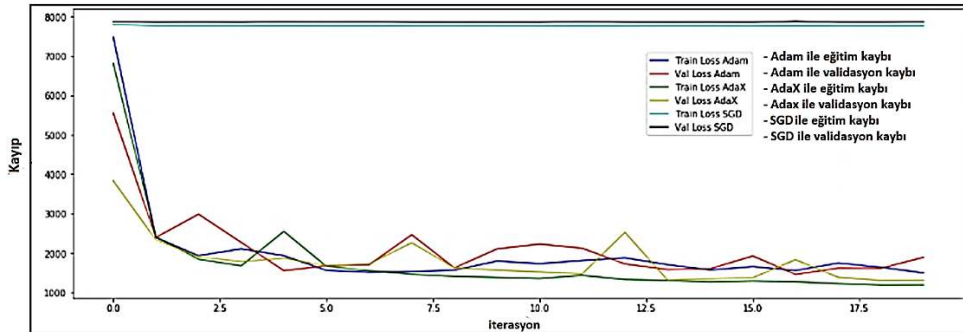
LR oranlarının seçilmesi zor konulardır [73], bu nedenle çok küçük LR oranlarının seçilmesi, takılıp kalabilecek uzun eğitim prosedürlerine neden olabilirken, çok büyük LR oranları, dengesiz bir eğitim prosedürüne veya bir alt optimal ağırlık setinin hızlı öğrenilmesine neden olabilir. LSTM hakkında, bu modeller yüksek hesaplama karmaşıklığına sahip derin tekrarlayan (recurrent) sinir ağı mimarileridir, LSTM mimarileri sürekli olarak gelişirken, eğitim algoritmaları oldukça sabit kalmıştır, yani SGD yöntemleri [74], Momentum tekniği doğal olarak SGD algoritmalarına daha sonra entegre edilmiştir ve bugüne kadar LSTM için standart eğitim rejimi olmaya devam ediyor. Uygulamadaki gelişmelere rağmen, SGD algoritmalarının eğitimde daha fazla iyileştirmeyi sınırlayan dezavantajları vardır. İlk dezavantajı, LR ve yakınsama kriterleri gibi manuel hiper parametre ayarlaması gerektirmesidir. İkinci verimsizlik, toplu öğrenmeden (batch learning) farklı olarak, SGD algoritmalarında ciddi optimizasyon için küçük bir alan kalmıştır çünkü yineleme başına gürültülü yapıya sahip bir örnek, çıktıyı daha fazla optimizasyon kullanmak için güvenilmez hale getirir.

Üçüncü eksiklik, SGD algoritmalarının doğası gereği sıralı olmaları ve GPU'ları kullanarak paralel hale getirmenin veya bir bilgisayar kümesiyle dağıtmanın son derece zor olmasıdır. Önerilen modeli doğrulamak için, REDD verilerini eğitim süreçleri için 0.3 validasyon oranı ile böldük. Ev 1'den LSTM-MOYİ eğitimi için seçilen buzdolabı verileri, 17 kullanım günü güç verisi ve altı günlük test verilerini içerir. Tablo 3, Adax optimizer tarafından sağlanan LSTM-MOYİ ağınnın, ev 1'den gelen verilerle en iyi performansı elde edebileceğini gösterdi. Ayrıca 1647.160 MSE'yi, 1191.1396'lık nihai kaybı ve 0.4794'lük doğruluğu da başarabilir. Adam tarafından LSTM-MOYİ, 1724.310 MSE, 1497.9198 nihai kayıp ve 0.4293 doğruluk elde etti. Buna karşılık, SGD modelinin LSTM-MOYİ modelinde yalnızca 8500.840 MAE ve 7770.7919 nihai kayıp vardı. Ayrıca en yüksek MSE ve en düşük doğruluk değeri 0,0000e + 00 ile, yukarıda belirtilen SGD'nin özyineli sinir ağları ve LSTM modellerinde belirtilen verimsizlikleri nedeniyle modelin eğitilmediği anlamına gelir.

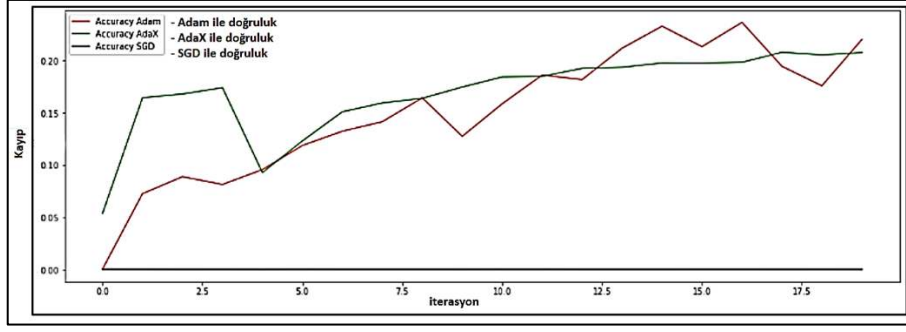
Tablo 3. Eğitim sırasında görülen evlerdeki enerji ayrıştırma performansları

Yöntemler	MSE	Final Hata	Doğruluk	LR	Steps
LSTM-MOYİ SGD	8500.840	7770.7919	0.0000e+00	0.001	223188
LSTM-MOYİ ADAM	1724.310	1497.9198	0.5293	5e-3	223188
LSTM-MOYİ AdaX	1647.160	1191.1396	0.5794	0.001	223188

Deneyler, önerilen modelin tüm testlerdeki hiper parametreleri baştan sona ayarladığını ve adil karşılaştırmalar sağlamak için tüm algoritmalar için en iyi sonuçları rapor ettiğini doğruladı. Tüm deneylerimiz Nvidia 2070 RTX Süper GPU'da çalışmıştır.

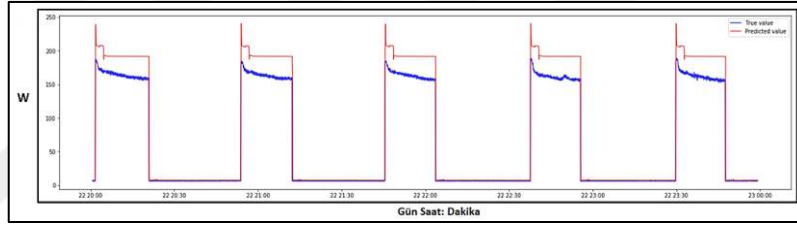


Şekil 63. Ev 1'de belirlenen LSTM-MOYİ yöntemine ilişkin eğitim ve doğrulama hataları (losses)

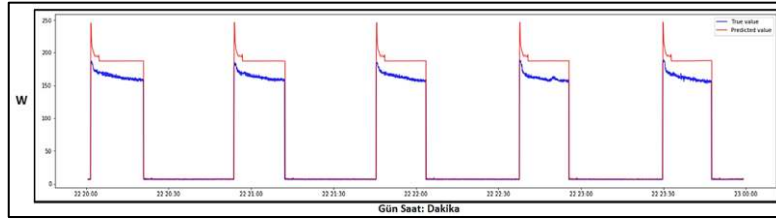


Şekil 64. Ev 1'deki validasyon seti ile LSTM-MOYİ yöntemi ile eğitimin doğruluğu (acc)

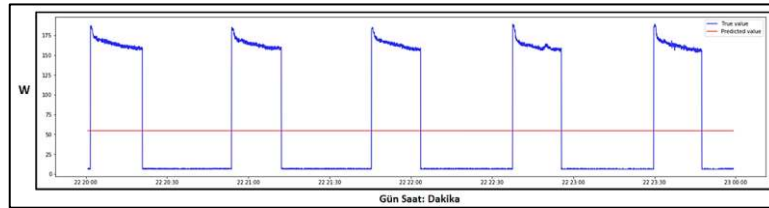
Şekil 63, LSTM-MOYİ eğitim hatalarını ve buzdolabı verileri için ev 1'deki validasyon setini göstermektedir. Şekil 64, LSTM-MOYİ eğitiminin Doğruluklarını ve buzdolabı verileri için ev 1'deki validasyon setini gösterir.



Şekil 65. Adam algoritması ile ev 1'in altıncı test gününde gerçek ve tahmini buzdolabı gücünün zamana göre değişimi



Şekil 66. AdaX algoritması ile ev 1'in altıncı test gününde gerçek ve tahmini buzdolabı gücünün zamana göre değişimi



Şekil 67. SGD algoritması ile ev 1'in altıncı test gününde gerçek ve tahmini buzdolabı gücünün zamana göre değişimi

Şekil 65, Adam tarafından ayarlanan ev 1'in altı test gününde buzdolabının gerçek ve tahmini sinyalıdır. Şekil 66, AdaX algoritması tarafından ayarlanan ev 1'in altı test gününde buzdolabının gerçek ve tahmini sinyalıdır. Ve şekil 67, SGD tarafından ayarlanmış ev 1'in altı test gününde buzdolabının gerçek ve tahmini sinyalıdır.

3.1. REDD Üzerinde BERT-MOYİ Yöntem Doğrulaması

Önerilen sınıflandırıcı modelindeki yaklaşımını doğrulamak için, REDD (benchmark) verilerini, elde edilen modeli eğitmek ve test etmek için bir doğrulama setine bölünmüştür. Birinci evden BERT-MOYİ'yi eğitmek için seçilen veriler, 17-günlük güç verisini içerirken, test için 6 günlük veri kullanılmıştır.

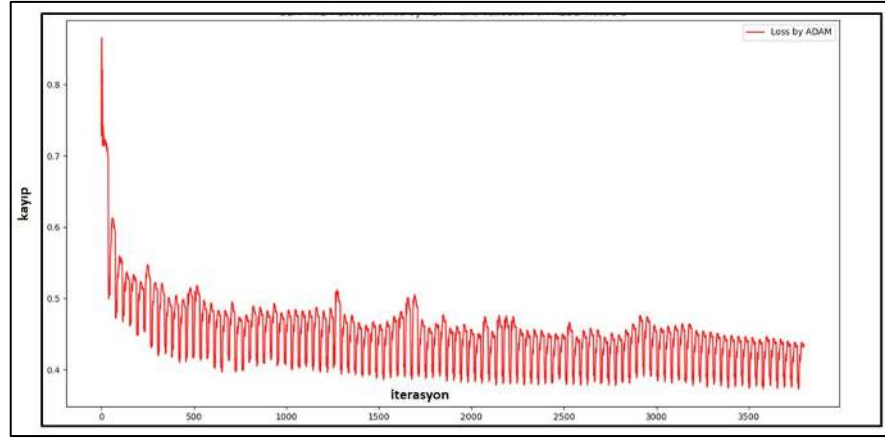
Tablo 4'de REDD'deki tüm cihazları için, Ortalama Doğruluk (MA), Ortalama Kesinlik (MP), Ortalama Duyarlılık (MR), Ortalama F1 puanı (MF1), Ortalama Göreceli Hata (MRE) ve Ortalama Mutlak Hatadır (MAE) sunulan metriklerdir, Ayrıca sınıflandırıcının Regresyon çıktıları 0 ve 1 aralığında gösterilmektedir.

Tablo 4. REDD test seti ile BERT-MOYİ metoduna ilişkin ortalama başarı performansı

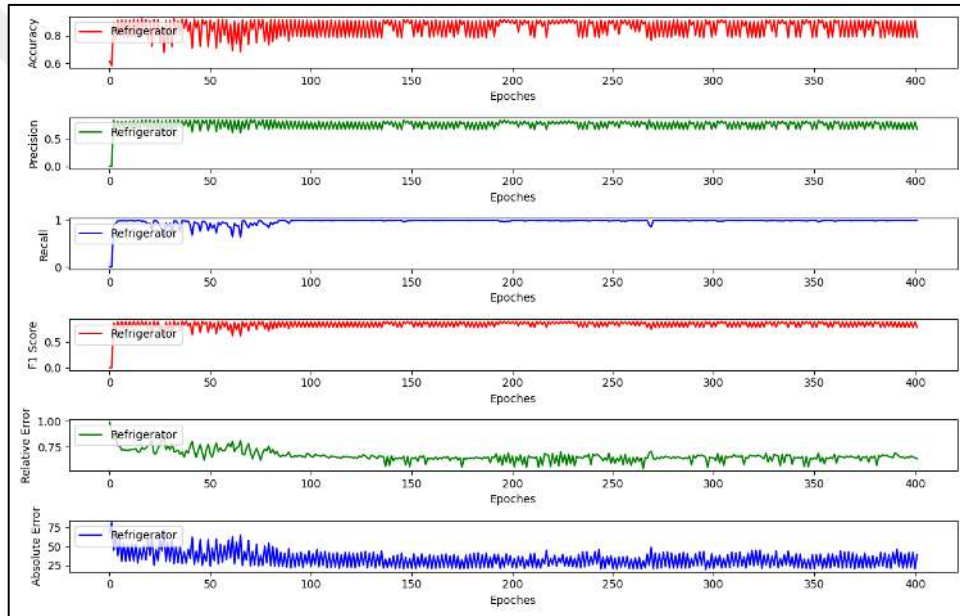
BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.95	0.54	0.74	0.58	0.24	26.49
Adam Algoritması	0.94	-	-	0.57	0.23	26.35

Tablo 5. REDD veri setindeki ev 1'deki BERT-MOYİ metoduna ilişkin buzdolabı başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.89	0.71	0.98	0.82	0.04	25.16
Adam Algoritması	0.84	-	-	0.75	0.80	32.35



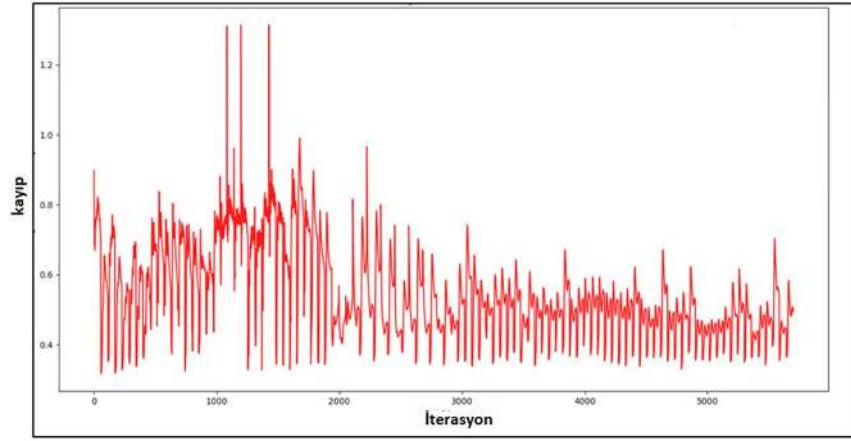
Şekil 68. BERT-MOYİ metodu ile buzdolabına ilişkin kayıplar



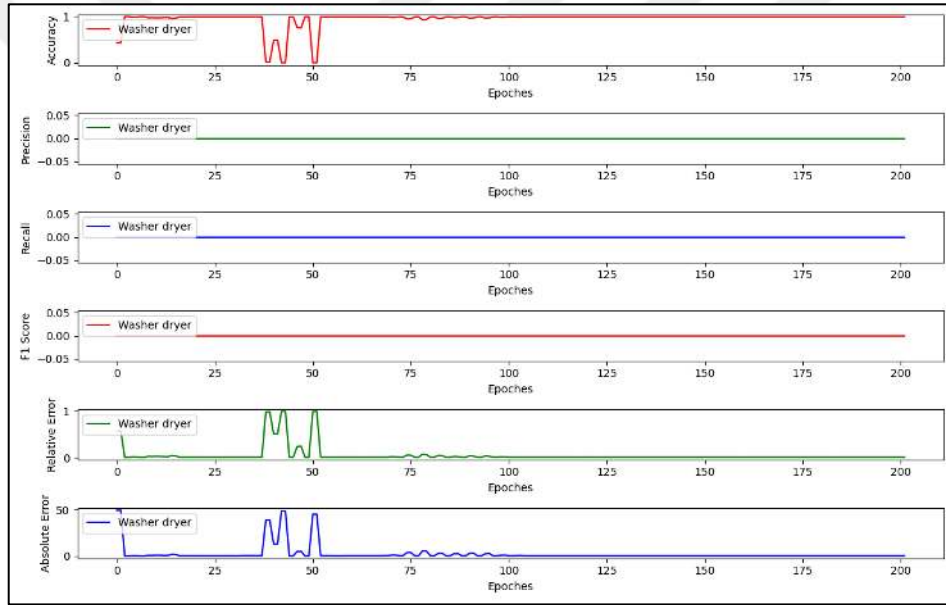
Şekil 69. REDD veri setindeki ev 1'de BERT-MOYİ metoduna göre buzdolabı başarı metrikleri

Tablo 6. REDD veri setindeki ev 1'deki BERT-MOYİ metoduna göre çamaşır makinesi başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.971	0.68	0.36	0.47	0.04	25.16
Adam Algoritması	0.969	-	-	0.52	0.039	20.49



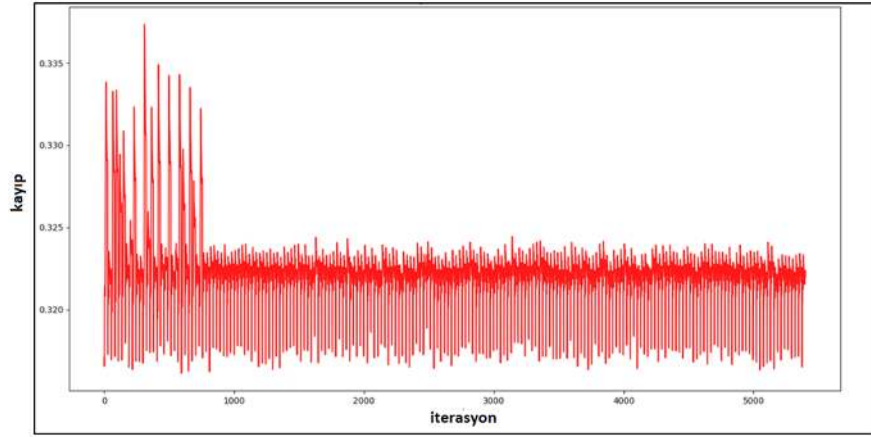
Şekil 70. BERT-MOYİ metoduna göre Çamaşır Makinesine ilişkin kayıplar



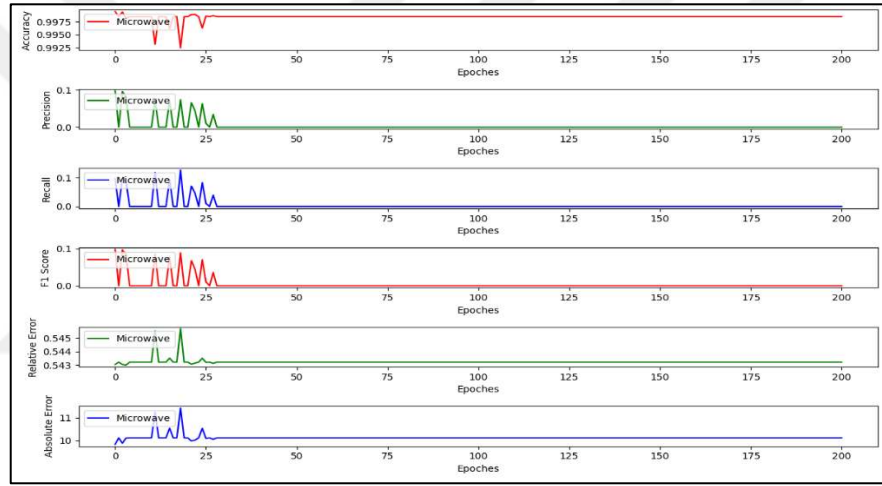
Şekil 71. REDD veri setindeki ev 1'de BERT-MOYİ metoduna göre Çamaşır Makinesi başarı metrikleri

Tablo 7. REDD veri setindeki ev 1'deki BERT-MOYİ metoduna göre mikrodalga başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.97	0.32	0.71	0.45	0.07	20.56
Adam Algoritması	0.98	-	-	0.47	0.05	17.58



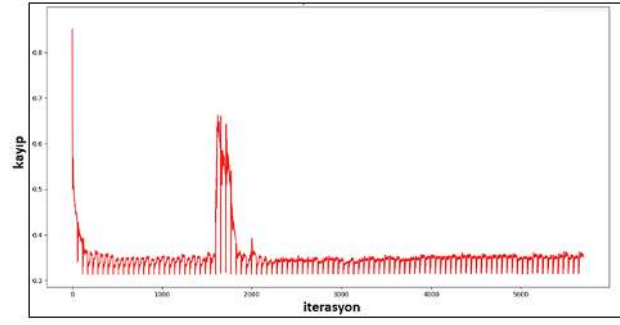
Şekil 72. BERT-MOYİ metoduna göre mikrodalga cihazına ilişkin kayıplar



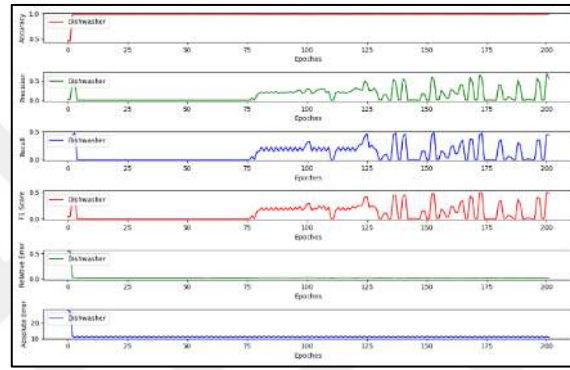
Şekil 73. REDD veri setindeki ev 1'de BERT-MOYİ metoduna göre mikrodalga başarı metrikleri

Tablo 8. REDD veri setindeki ev 1'deki BERT-MOYİ metoduna göre bulaşık makinesinin başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.96	0.68	0.36	0.47	0.04	25.16
Adam Algoritması	0.96	-	-	0.52	0.03	20.49



Şekil 74. BERT-MOYİ metoduna göre bulaşık makinesine ilişkin kayıplar



Şekil 75. REDD veri setindeki ev 1'de BERT-MOYİ metoduna göre bulaşık makinesi başarı metrikleri

4. İRDELEME

4.1. TEEAVS ile BERT-MOYİ Eğitimi ve Başarı Metrikleri

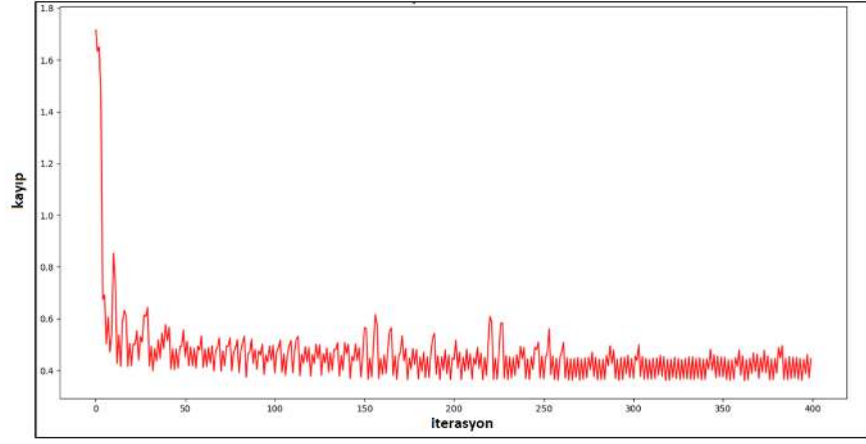
Bulgular kısmında REED veri seti ile tasarlanan sınıflandırıcı yönteminin doğrulaması yapılmıştır. Şimdi ülkemiz için çalışmayı yerli veri setimiz ile başarı metriklerini irdeliyoruz. Tasarlanan BERT-MOYİ modeli önerilen TEEAVS [75] veri setinin birinci evdeki veriler ile eğitilmiş ve ikinci evdeki veriler ile test edilmiştir.

Tablo 9. TEEAVS veri setindeki ev 2'deki BERT-MOYİ metoduna göre tüm cihazların ortalama başarı metrikleri

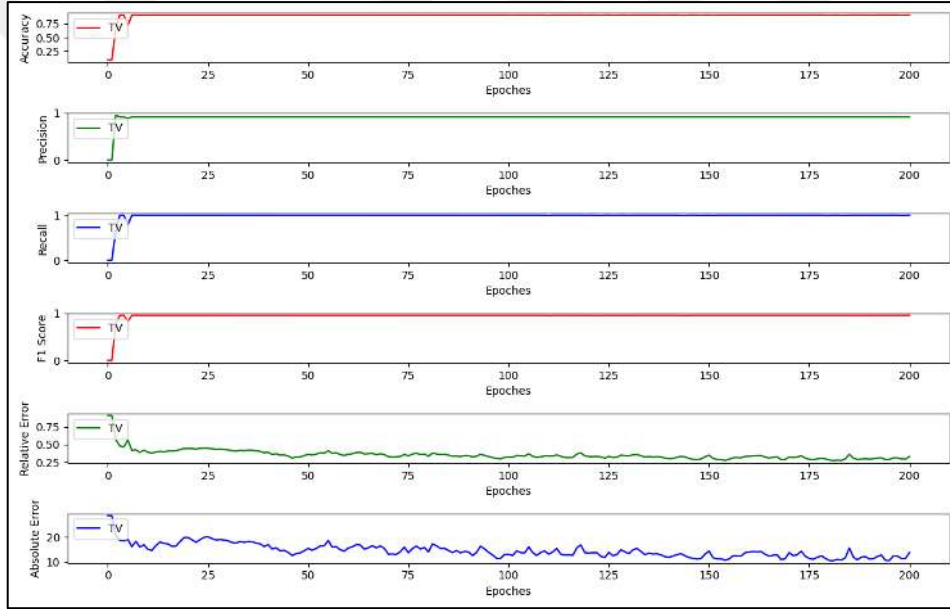
BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.86	0.86	0.93	0.89	0.35	23.42
Adam Algoritması	0.84	0.84	0.97	0.89	0.27	16.35

Tablo 10. TEEAVS veri setindeki ev 2'deki BERT-MOYİ Metoduna göre TV başarı Metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.92	0.92	0.99	0.96	0.18	5.47
Adam Algoritması	0.92	0.92	1	0.96	0.17	4.99



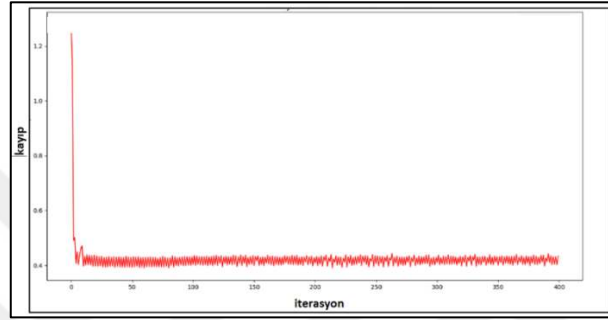
Şekil 76. BERT-MOYİ metoduna göre TV cihazına ilişkin kayıplar



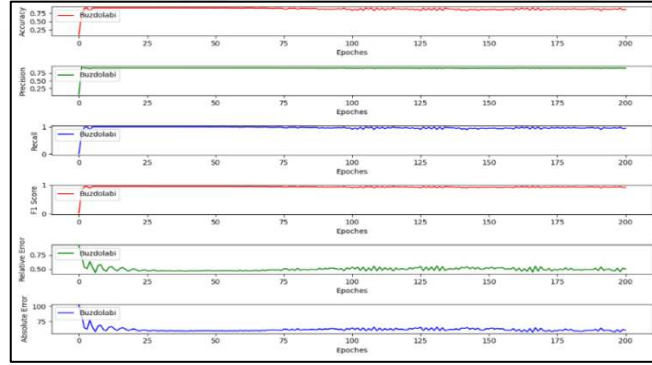
Şekil 77. TEEAVS veri setindeki ev 2'de BERT-MOYİ metoduna göre TV başarı metrikleri

Tablo 11. TEEAVS veri setindeki ev 2'deki BERT-MOYİ Metoduna göre Buzdolabı başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.92	0.92	1	0.96	0.43	54.22
Adam Algoritması	0.92	0.92	0.98	0.95	0.27	28.66



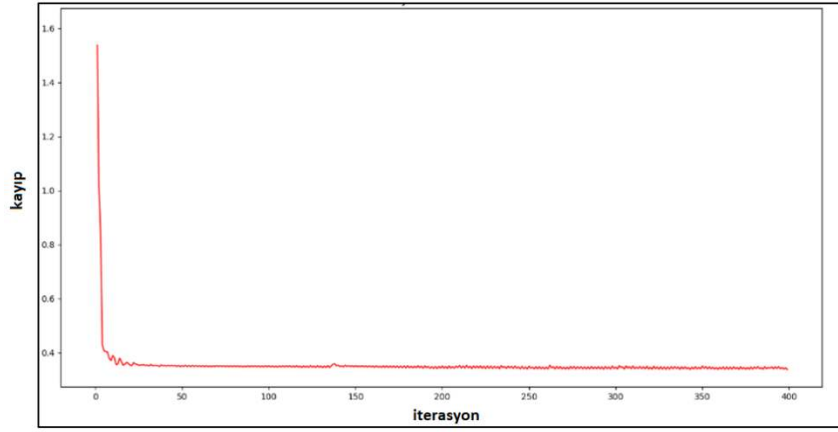
Şekil 78. BERT-MOYİ metoduna göre buzdolabına ilişkin kayıplar



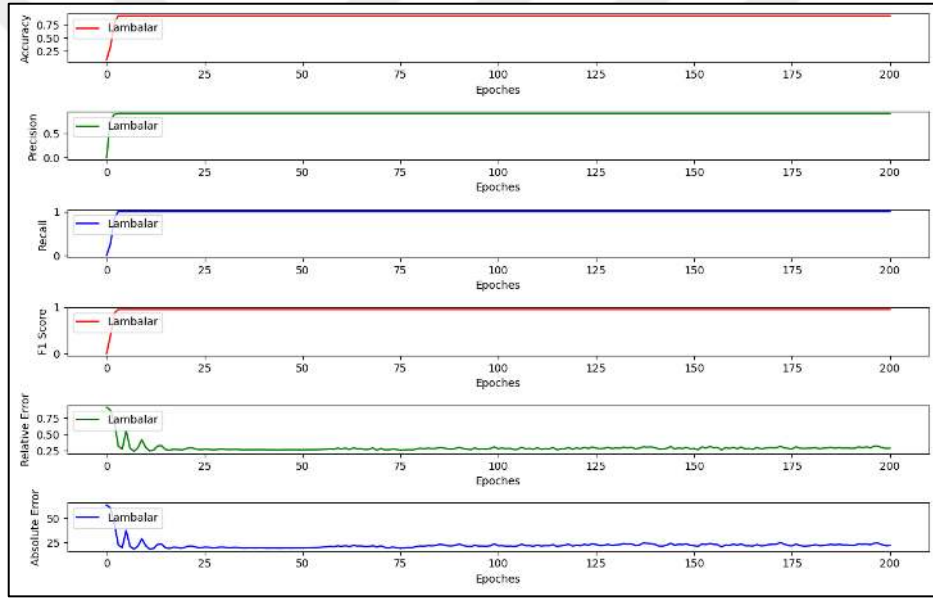
Şekil 79. TEEAVS veri setindeki ev 2'de BERT-MOYİ metoduna göre buzdolabı başarı metrikleri

Tablo 12. TEEAVS veri setindeki ev 2'deki BERT-MOYİ Metoduna göre lambaların başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.99	0.99	1	0.99	0.29	20.69
Adam Algoritması	0.99	0.99	1	0.99	0.27	18.32



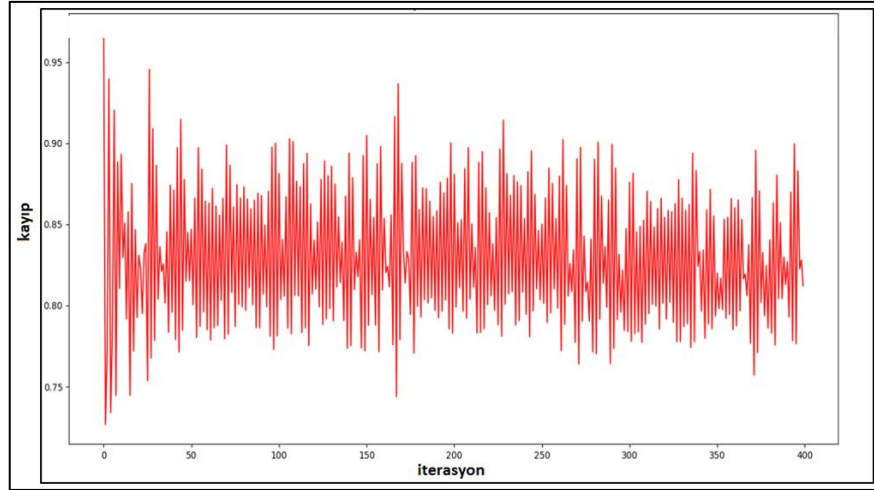
Şekil 80. BERT-MOYİ metoduna göre lambalara ilişkin kayıplar



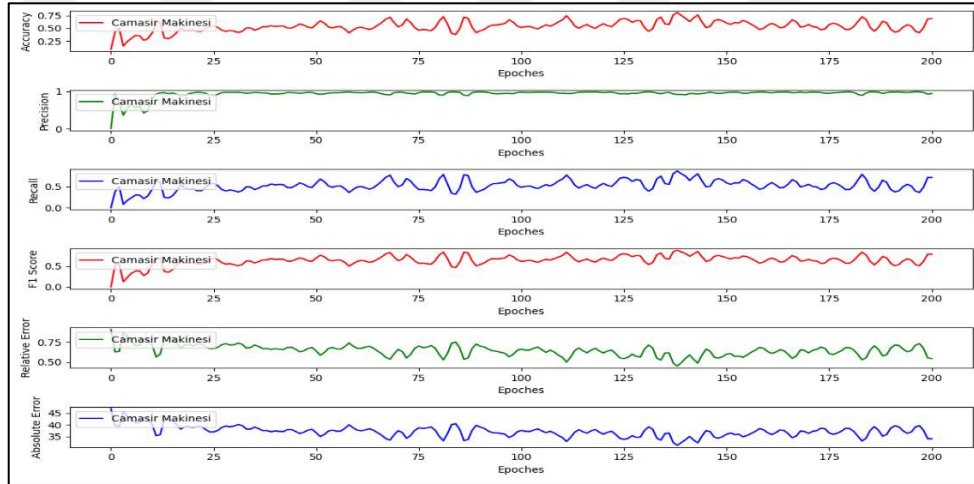
Şekil 81. TEEAVS veri setindeki ev 2'de BERT-MOYİ metoduna göre lambaların başarı metrikleri

Tablo 13. TEEAVS veri setindeki ev 2'deki BERT-MOYİ metoduna göre çamaşır makinesi başarı metrikleri

BERT-MOYİ Metodu	MA	MP	MR	MF1	MRE	MAE
AdaX Algoritması	0.60	0.61	0.72	0.66	0.51	13.32
Adam Algoritması	0.54	0.54	0.91	0.68	0.40	13.44



Şekil 82. BERT-MOYİ metoduna göre çamaşır makinesine ilişkin kayıplar



Şekil 83. TEEAVS veri setindeki ev 2'de BERT-MOYİ metoduna göre çamaşır makinesi başarı metrikleri

Elden edilen sonuçlara göre en başarılı çıktı Lambalar olduğunu göstermektedir, lambalar rezistif yük tipi kategoride yer aldıkları için model karmaşıklığının az olduğundan dolayı anazili daha kolay olmuştur. Elden edilen sonuçlara göre en düşük cihaz tahmini çamaşır makinesi olduğunu göstermektedir, çamaşır makinesi indüktif yük tipi kategoride yer aldıkları için model karmaşıklığının fazla olduğundan dolayı anazili daha zor olmuştur.

5. SONUÇLAR

Bu tezde, ilk önce bir LSTM sekans modeline dayanan geleneksel MOYİ yönteminin performansını iyileştirmek için AdaX optimizasyon algoritması ile bir Enerji Ayırıştırma modeli tasarlanmıştır. Öncelikle, bir LSTM-MOYİ probleminde SGD'nin hatayı minimum noktaya yakınsaması sorunları incelenmiştir, Adam ve AdaX yöntemlerinin daha etkili ve hızlı yakınsamalarının olabilmesi dikkate alınarak performansları ele alınmıştır. Daha sonra AdaX'in LSTM-MOYİ'e uygulanması önerilmiştir, yakınsamaları analiz edilmiş ve Enerji Ayırıştırma görevinin öğrenme sinyalleri üzerindeki başarı performansı değerlendirilmiştir. Analizimiz ve deneysel sonuçlarımız, AdaX tarafından ayarlanmış ayırıştırıcı modelin, LSTM'ye dayalı Enerji Ayırıştırma görevinde son Adam optimize ediciden daha iyi performans göstermiştir. Sekans modeline dayanan MOYİ başarı performans incelemesinden sonra bir Transformer modeli MOYİ tasarımına adapte edilmiştir ve en iyi cihaz sınıflandırıcı modeli olarak önerilmiştir.

Özetle, bu çalışma derin yapay sinir ağları tabanlı MOYİ ile ilgili literatürü gözden geçirdi. İnceleme, düşük frekanslı veriler, yani alternatif akım (AC) frekansından daha düşük örnekleme oranları verileri aracılığıyla ev aletlerinin enerji sınıflandırması için derin öğrenme ve verimli optimizasyon yöntemlerini kullanan birkaç çalışmayı içerir. Bu çalışma için motivasyonumuz, düşük frekanslı verilerin yakın gelecekte uygun ölçekte mevcut olacağı ve diğer uygulama alanlarında muazzam bir derin öğrenme başarısı olduğu için, IoT geliştirmeleriyle entegre iyi eğitilmiş (well-trained) bir MOYİ'den birçok makalenin faydalanabilmesiydi. Enerji ve sürdürülebilirlik sorunları, veri madenciliği ve makine öğrenimi yaklaşımları kullanılarak ele alınabilir. Ancak, kamuya açık veri kümelerinin olmaması nedeniyle bu tür sorunlar yavaş çözülmüştür. Bu çalışmada, akıllı şebekelerde enerji ayırıştırma araştırmalarını ilerletmeyi amaçlayan evlerden toplanan elektrik kullanım bilgilerini içeren TEEAVS sunulmuştur.

Akıllı ölçüm bağlamında, ev aletlerini TEEAVS bilgilerinin bir fonksiyonu olarak yerli cihaz sınıflandırma için bir MOYİ modeli önerilmiştir. MOYİ sistemi, talep tarafındaki ağır yükler sırasında ayrıntılı geri bildirim ve farkındalık sağlayarak kullanıcıların enerji talebini azaltarak üretim varlıklarının daha verimli kullanılmasını sağlar. Böylece, son kullanıcılar, pahalı akıllı soket sensörlerine ihtiyaç duymadan, talep üzerine son derece hassas denetimli MOYİ ile ayrıntılı faturalandırma elde edebilecektir.

Önerilen MOYİ'nin faydaları, halihazırda kurulu akıllı sayaçlara yerleştirilebilen MOYİ ile bir enerji farkındalığı sistemi aracılığıyla enerji tüketimini azaltmak için kullanılan enerji verimliliği ve ayrıntılı enerji verileridir. Dezavantajları ise sürekli değişken cihazın algılanamaması, elektriksel olarak aynı (her detayda benzer) cihazların ayırt edilememesi, daha büyük bir tespit edilemeyen hata potansiyeli olması ve olağandışı cihazların tanınmasının zor olmasıdır. Bu yazıda, geleneksel MOYİ yöntemlerinin performansını iyileştirmek için AdaX optimizasyon algoritması ile ayarlanmış bir BERT modeline sahip verimli bir enerji ayrıştırıcısı sunulmuştur. Ardından, BERT-MOYİ'ye uygulanan AdaX'i ("uzun süreli belleğe sahip" bir optimize edici) önerilmiştir.



6. ÖNERİLER

Analizimiz ve deneysel sonuçlarımız, AdaX ile ayarlanmış (tune) transformer bazlı ayrıştırıcı modelinin, BERT'ye dayalı enerji ayrıştırma görevinde son (state-of-the-art) Adam optimize ediciden daha iyi performans gösterdi. MOYİ'nin toplam performansının değerlendirilmesi için daha fazla araştırmaya ihtiyaç duyulduğu not edilmelidir. Ek olarak, mevcut çalışma, basit MOYİ yaklaşımlarının ötesinde verimli enerji ayrıştırma yöntemleri tasarlanmanın ilk adımını temsil etmektedir. Ancak, somut bir ifade için diğer yeni tasarımlar da incelenmelidir. Modern derin öğrenme ve verimli geri yayılım bağlamında, uyarlanabilir öğrenmeyi gerçekleştirmek için uzun vadeli gradyan önbelleğine sahip yeni optimizasyon algoritmalarının MOYİ görevlerinde AdaX'tan daha iyi performans gösterebileceği düşünülmektedir; ancak bunların yakınsaması ve performansının kapsamlı bir şekilde araştırılması gerekir.

7. KAYNAKLAR

1. Fichman, B. T., Annual Energy Review, USDOE Energy Information Administration (EIA), Office of Energy Markets and End Use, (2009) 0384
2. Çavdar, İ. H. ve Faryad, V., New Design of a Supervised Energy Disaggregation Model Based on the Deep Neural Network for a Smart Grid, *Energies*, 12(7), (2019) 1217
3. Kolter, J. Z. ve Johnson, M. J., REDD: A public data set for energy disaggregation research, In *Workshop on Data Mining Applications in Sustainability (SIGKDD)*, (2011) 59-62.
4. Paltoo, D. N., Rodriguez, L. L., Feolo, M., Gillanders, E., Ramos, E. M., Rutter, J. L. ve Green, E. D., Data use under the NIH GWAS data sharing policy and future directions, *Nature genetics*, 46(9), (2014) 934-938.
5. Marcus, M., Santorini, B. ve Marcinkiewicz, M. A. Building a large annotated corpus of English: The Penn Treebank. (1993).
6. LeCun, Y., Bottou, L., Bengio, Y. ve Haffner, P., Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, 86, (1998) 2278-2324.
7. Fei-Fei, L., Fergus, R. ve Perona, P., Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories, *Conference on Computer Vision and Pattern Recognition Workshop*, (2004) 178-178.
8. Everingham, M., Van Gool, L., Williams, C. K., Winn, J. ve Zisserman, A., The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88, (2010) 303-338.
9. Pérez-Lombard, L., Ortiz, J. ve Pout, C., A review on buildings energy consumption information, *Energy and buildings*, 40, (2008) 394-398.
10. Faustine, A., Mvungi, N. H., Kaijage, S. ve Michael, K., A survey on non-intrusive load monitoring methodologies and techniques for energy disaggregation problem, (2017) 1703.00785.
11. Batra, N., Singh, A. ve Whitehouse, K., If you measure it, can you improve it? Exploring the value of energy disaggregation, *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, (2015) 191-200.

12. Batra, N., Singh, A. ve Whitehouse, K., Gemello: Creating a detailed energy breakdown from just the monthly electricity bill, In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, (2016). 431-440.
13. Zhong, M., Goddard, N. ve Sutton, C., Interleaved factorial non-homogeneous hidden Markov models for energy disaggregation, (2014)1406.7665.
14. Reyes Lua, A. R., Location-aware energy disaggregation in smart homes, Repository. tudelft. nl (2015).
15. Froehlich, J., Larson, E., Gupta, S., Cohn, G., Reynolds, M. ve Patel, S., Disaggregated end-use energy sensing for the smart grid, IEEE Pervasive Computing, 10, (2010) 28-39.
16. Hart, G. W., Nonintrusive appliance load monitoring, Proceedings of the IEEE, 80(12), (1992) 1870-1891.
17. Barsim, K. S. ve Yang, B., Toward a semi-supervised non-intrusive load monitoring system for event-based energy disaggregation, IEEE Global Conference on Signal and Information Processing (GlobalSIP), (2015) 58-62.
18. Makonin, S., Popowich, F., Bajić, I. V., Gill, B. ve Bartram, L., Exploiting HMM sparsity to perform online real-time nonintrusive load monitoring, IEEE Transactions on Smart Grid, 7, (2015) 2575-2585.
19. Stankovic, V., Liao, J. ve Stankovic, L., A graph-based signal processing approach for low-rate energy disaggregation, IEEE symposium on computational intelligence for engineering solutions (CIES), (2014) 81-87.
20. Do Nascimento, P. P. M., Applications of deep learning techniques on NILM. Diss, Universidade Federal do Rio de Janeiro (2016).
21. Lowe, D. G., Object recognition from local scale-invariant features, In iccv, 99(2), (1999) 1150-1157.
22. Krizhevsky, A., Sutskever, I. ve Hinton, G. E., Imagenet classification with deep convolutional neural networks, In Advances in neural information processing systems, (2012) 1097-1105.
23. Graves, A. ve Jaitly, N., Towards end-to-end speech recognition with recurrent neural networks, In International conference on machine learning, (2014) 1764-1772.
24. Roos, J. G., Lane, I. E., Botha, E. C. ve Hancke, G. P., Using neural networks for non-intrusive monitoring of industrial electrical loads, In Conference Proceedings, 10th Anniversary, IMTC/94, Advanced Technologies in I & M, Instrumentation and Measurement Technolgy Conference, 94CH3424-9, (1994) 1115-1118.

25. Yang, H. T., Chang, H. H. ve Lin, C. L., Design a neural network for features selection in non-intrusive monitoring of industrial electrical loads, 11th International Conference on Computer Supported Cooperative Work in Design, (2007) 1022-1027.
26. Gelazanskas, L. ve Gamage, K. A., Demand side management in smart grid: A review and proposals for future direction, Sustainable Cities and Society, 11, (2014) 22-30.
27. Behrangrad, M., A review of demand side management business models in the electricity market, Renewable and Sustainable Energy Reviews, 47, (2015), 270-283.
28. Athanasiadis, C., Doukas, D., Papadopoulos, T. ve Chrysopoulos, A., A Scalable Real-Time Non-Intrusive Load Monitoring System for the Estimation of Household Appliance Power Consumption, Energies, 14(3), (2021) 767.
29. Garcia, F. D., Souza, W. A., Diniz, I. S. ve Marafão, F. P., NILM-based approach for energy efficiency assessment of household appliances, Energy Informatics, 3(1), (2020) 1-21.
30. Gopinath, R., Kumar, M., Joshua, C. P. C. ve Srinivas, K., Energy management using non-intrusive load monitoring techniques-State-of-the-art and future research directions, Sustainable Cities and Society, (2020) 102411.
31. Moradzadeh, A., Sadeghian, O., Pourhossein, K., Mohammadi-Ivatloo, B. ve Anvari-Moghaddam, A., Improving res-idential load disaggregation for sustainable development of energy via principal component analysis, Sustainability, 12(8), (2020) 3158.
32. de Souza, W. A., Garcia, F. D., Marafão, F. P., Da Silva, L. C. P. ve Simões, M. G., Load disaggregation using micro-scopic power features and pattern recognition, Energies, 12(14), (2019) 2641.
33. Massidda, L., Marrocu, M. ve Manca, S., Non-intrusive load disaggregation by convolutional neural network and multilabel classification, Applied Sciences, 10(4), (2020) 1454.
34. Kalluri, B., Kamilaris, A., Kondepudi, S., Kua, H. W. ve Tham, K. W., Applicability of using time series subsequences to study office plug load appliances, Energy and Buildings, 127, (2016) 399-410.
35. Zhao, B., He, K., Stankovic, L. ve Stankovic, V. Improving event-based non-intrusive load monitoring using graph signal processing, IEEE Access, 6, (2018) 53944-53959.
36. Meziane, M. N., Picon, T., Ravier, P., Lamarque, G. ve Le, J., A new measurement system for high frequency nilm with controlled aggregation scenarios. In Workshop on Non-Intrusive Load Monitoring (NILM), 2016 Proceedings of the 3rd International. (2016).
37. Sadeghianpourhamami, N., Ruysinck, J., Deschrijver, D., Dhaene, T. ve Develder, C., Comprehensive feature selection for appliance classification in NILM, Energy and Buildings, 151, (2017) 98-106.

38. Wang, A. L., Chen, B. X., Wang, C. G. ve Hua, D., Non-intrusive load monitoring algorithm based on features of V–I trajectory, *Electric Power Systems Research*, 157, (2018) 134-144.
39. Ruano, A., Hernandez, A., Ureña, J., Ruano, M. ve Garcia, J., NILM techniques for intelligent home energy management and ambient assisted living: A review, *Energies*, 12(11), (2019) 2203.
40. Çimen, H., Çetinkaya, N., Vasquez, J. C. ve Guerrero, J. M., A microgrid energy management system based on non-intrusive load monitoring via multitask learning, *IEEE Transactions on Smart Grid*, 12(2), (2020) 977-987.
41. Elahe, M. F., Jin, M. ve Zeng, P., Review of load data analytics using deep learning in smart grids: Open load datasets, methodologies, and application challenges, *International Journal of Energy Research*, 45, (2021) 14274–14305.
42. Hernández, Á., Ruano, A., Ureña, J., Ruano, M. G. ve Garcia, J. J., Applications of NILM techniques to energy management and assisted living, *IFAC-PapersOnLine*, 52(11), (2019) 164-171.
43. de Paiva Penha, D. ve Castro, A. R. G., Home appliance identification for NILM systems based on deep neural networks, *International Journal of Artificial Intelligence & Applications*, 9, (2018) 69-80.
44. Salerno, V. M. ve Rabbeni, G., An extreme learning machine approach to effective energy disaggregation, *Electronics*, 7(10), (2018) 235.
45. Yu, J., Zhang, C. ve Wang, S., Multichannel one-dimensional convolutional neural network-based feature learning for fault diagnosis of industrial processes, *Neural Computing and Applications*, 33(8), (2021) 3085-3104.
46. Bai, Y., Xie, J., Liu, C., Tao, Y., Zeng, B. ve Li, C., Regression modeling for enterprise electricity consumption: a com-parison of recurrent neural network and its variants, *International Journal of Electrical Power & Energy Systems*, 126, (2021) 106612.
47. Himeur, Y., Alsalemi, A., Bensaali, F. ve Amira, A., Smart power consumption abnormality detection in buildings us-ing micromoments and improved K-nearest neighbors, *International Journal of Intelligent Systems*, 36(6), (2021), 2865-2894.
48. Singh, M., Kumar, S., Semwal, S., Prasad, R.S., Residential load signature analysis for their segregation using wavelet—SVM, In *Power Electronics and Renewable Energy Systems*; Springer: New Delhi, India, (2015) 863–871.
49. Chowdhury, D. ve Hasan, M. M., Non-intrusive load monitoring using ensemble empirical mode decomposition and random forest classifier. In *Proceedings of the International Conference on Digital Image and Signal Processing (DISP)*, Oxford, UK, (2019) 29-30.

50. Yang, C. C., Soh, C. S. ve Yap, V. V., A non-intrusive appliance load monitoring for efficient energy consumption based on Naive Bayes classifier, *Sustainable Computing: Informatics and Systems*, 14, (2017) 34-42.
51. Saha, D., Bhattacharjee, A., Chowdhury, D., Hossain, E. ve Islam, M. M., Comprehensive NILM framework: device type classification and device activity status monitoring using capsule network, *IEEE Access*, 8, (2020), 179995-180009.
52. Bonfigli, R., Principi, E., Fagiani, M., Severini, M., Squartini, S. ve Piazza, F., Non-intrusive load monitoring by using active and reactive power in additive Factorial Hidden Markov Models, *Applied Energy*, 208, (2017) 1590-1607.
53. Jazizadeh, F., Becerik-Gerber, B., Berges, M. ve Soibelman, L., An unsupervised hierarchical clustering based heuristic algorithm for facilitated training of electricity consumption disaggregation systems, *Advanced Engineering Informatics*, 28(4), (2014) 311-326.
54. Nalmpantis, C. ve Vrakas, D., Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparison, *Artificial Intelligence Review*, 52(1), (2019) 217-243.
55. Machlev, R., Belikov, J., Beck, Y. ve Levron, Y., MO-NILM: A multi-objective evolutionary algorithm for NILM classification, *Energy and Buildings*, 199, (2019) 134-144.
56. Lin, Y. H., Trainingless multi-objective evolutionary computing-based nonintrusive load monitoring: Part of smart-home energy management for demand-side management, *Journal of Building Engineering*, 33, (2021) 101601.
57. Yang, Z., Ghadamyari, M., Khorramdel, H., Alizadeh, S. M. S., Pirouzi, S., Milani, M., ve Ghadimi, N., Robust mul-ti-objective optimal design of islanded hybrid system with renewable and diesel sources/stationary and mobile energy storage systems, *Renewable and Sustainable Energy Reviews*, 148, (2021) 111295.
58. Çavdar, İ. H. ve Faryad, V., New design of a supervised energy disaggregation model based on the deep neural network for a smart grid, *Energies*, 12(7), (2019) 1217.
59. Nguyen, M., Alshareef, S., Gilani, A. ve Morsi, W. G., A novel feature extraction and classification algorithm based on power components using single-point monitoring for NILM. In *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)* (pp. 37-40). IEEE. (2015).
60. Kolter, J. Z. ve Johnson, M. J., REDD: A public data set for energy disaggregation research, In *Workshop on data mining applications in sustainability (SIGKDD)*, San Diego, CA (Vol. 25, No. Citeseer, (2011) 59-62.
61. Ahmed, S. ve Bons, M., Edge computed NILM: a phone-based implementation using MobileNet compressed by Tensorflow Lite, In *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*, (2020) 44-48.

62. Kelly, J. ve Knottenbelt, W., Neural nilm: Deep neural networks applied to energy disaggregation, In Proceedings of the 2nd ACM international conference on embedded systems for energy-efficient built environments, (2015) 55-64.
63. Zhang, Y., Yin, B., Cong, Y. ve Du, Z., Multi-state Household Appliance Identification Based on Convolutional Neural Networks and Clustering, *Energies*, 13(4), (2020) 792.
64. Jais, I. K. M., Ismail, A. R., ve Nisa, S. Q.. Adam optimization algorithm for wide and deep neural network. *Knowledge Engineering and Data Science*, 2(1), (2019) 41-46.
65. Zhuang, J., Tang, T., Ding, Y., Tatikonda, S., Dvornek, N., Papademetris, X., ve Duncan, J. S. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *arXiv preprint arXiv:2010.07468*, (2020).
66. Li, W., Zhang, Z., Wang, X., ve Luo, P. AdaX: Adaptive Gradient Descent with Exponential Long Term Memory. *arXiv preprint arXiv:2004.09740*, (2020).
67. Wang, J. K., Li, X., Karimi, B., ve Li, P. An optimistic acceleration of amsgrad for nonconvex optimization. *arXiv preprint arXiv:1903.01435*, (2019).
68. Kingma, D. P., ve Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, (2014).
69. Atmaja, S. T., & Halim, A. Steady State Modification Method Based On Backpropagation Neural Network For Non-Intrusive Load Monitoring (NILM). In *MATEC Web of Conferences* (Vol. 218, p. 02013). EDP Sciences, (2018).
70. Liu, H., Zhang, Z., Xu, Y., Wang, N., Huang, Y., Yang, Z., ... ve Chen, H. Use of BERT (Bidirectional Encoder Repre-sentations from Transformers)-based deep learning method for extracting evidences in chinese radiology reports: develop-ment of a computer-aided liver cancer diagnosis framework. *Journal of medical Internet research*, (2021) 23(1), e19689.
71. Hendrycks, D., ve Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, (2016)..
72. Xiao, P., ve Cheng, S. Neural network for NILM based on operational state change classification. *arXiv preprint arXiv:1902.02675*, (2019).
73. Kim, J., & Kim, H. (2017, February). An effective intrusion detection classifier using long short-term memory with gradient descent optimization. In *2017 International Conference on Platform Technology and Service (PlatCon)* (pp. 1-6). IEEE.
74. Principi, E., Bonfigli, R., Fagiani, M., Severini, M., Squartini, S., & Piazza, F. DEEP NEURAL NETWORKS PER LA DISAGGREGAZIONE DEL CARICO.

75. Turkey Electrical Appliances Dataset (TEAD) Containing Household Electricity Usage Data. Available online: (https://github.com/vahit19/smart_grid).



ÖZGEÇMİŞ

Lise öğrenimini Tebriz Kharazmi (harezmi) lisesinde tamamladı, 2010 yılında Zanjan Rouzbeh Üniversitesi Bilgi ve İletişim Mühendisliği Bölümünden ICT Mühendisi ünvanı ile mezun oldu, 2013 yılında Karadeniz Teknik Üniversitesi Elektrik-Elektronik Mühendisliği Bölümü'nden Elektronik Yüksek Mühendisi ünvanı ile mezun oldu. 2014 yılında KTÜ Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda Doktora öğrenimine başladı. Vahit FERYAD iyi derecede İngilizce, Türkçe, Farsça ve Azerice bilmektedir. Tezden üretilmiş SCI kapsamındaki dergilerde yayınlanmış makaleler:

Çavdar, İ. H., & Faryad, V. (2019). New design of a supervised energy disaggregation model based on the deep neural network for a smart grid. *Energies*, 12(7), 1217.

Çavdar, İ. H., & Feryad, V. (2021). Efficient Design of Energy Disaggregation Model with BERT-NILM Trained by AdaX Optimization Method for Smart Grid. *Energies*, 14(15), 4649.