

**T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

DERİN ÖĞRENME İLE METİN ÖZETLEME

YÜKSEK LİSANS TEZİ

Burak ERHANDI

**Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM
MÜHENDİSLİĞİ**

Tez Danışmanı : Dr. Öğr. Üyesi Fatih ÇALLI

Haziran 2020

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME İLE METİN ÖZETLEME

YÜKSEK LİSANS TEZİ

Burak ERHANDI

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM
MÜHENDİSLİĞİ

Bu tez tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

Jüri Başkanı

Üye

Üye

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Burak ERHANDI

26.12.2019

TEŞEKKÜR

Yüksek lisans tezim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğr. Üyesi Fatih ÇALLI'ya ve destekleri için sevgili aileme teşekkürlerimi sunarım.

İÇİNDEKİLER

TEŞEKKÜR	i
İÇİNDEKİLER	ii
ŞEKİLLER LİSTESİ	iv
TABLolar LİSTESİ	v
ÖZET	vi
SUMMARY	vii
 BÖLÜM 1.	
GİRİŞ	1
 BÖLÜM 2.	
LİTERATÜR ARAŞTIRMASI	5
 BÖLÜM 3.	
MATERYAL VE YÖNTEM	7
3.1. Metin Özetleme Kavramı	7
3.1.1. Metin özetlemede kullanılan çalışmalar	7
3.1.2. Metin özetlemede kullanılan yöntemler	8
3.2. Yapay Sinir Ağları	10
3.2.1. Yapay sinir ağlarının türleri	13
3.2.2. Yapay sinir ağları nasıl çalışır?	14
3.2.3. Yapay sinir ağlarının özellikleri	15
3.2.4. Yapay sinir ağlarının avantaj ve dezavantajları	16
3.2.5. Yapay sinir ağlarında öğrenme türleri	17
3.2.6. Yapay sinir ağlarında öğrenme kuralları	19
3.3. Derin Öğrenme Nedir?	20

3.3.1. Yapay zeka, makine öğrenmesi ve derin öğrenme ilişkisi.....	22
3.3.2. Derin öğrenme mimarileri	25
3.3.3. Derin öğrenme uygulamaları.....	32
3.3.4. Derin öğrenme kütüphaneleri	34
3.4. Yapay Sinir Ağları ve Derin Öğrenme İlişkisi	38
3.5. PageRank Algoritması ve TextRank Algoritmasının Gelişimi	38
3.6. Diğer Kavramlar	40
3.7. Kullanılan Sistem, Program, Kütüphane ve Teknolojiler	40
 BÖLÜM 4.	
ÖNERİLEN MODEL	42
4.1. Modelin Oluşturulması	43
4.2. Modellerin Eğitilmesi ve Özetleme Mekanizması	44
4.3. Kodlama Yapısı	45
4.4. Modelin Mimarisi	45
 BÖLÜM 5.	
DENEYSEL SONUÇLAR	52
5.1. Sonuçlar	52
5.2. Öneriler	61
 KAYNAKLAR	62
ÖZGEÇMİŞ	66

ŞEKİLLER LİSTESİ

Şekil 2.1. Antioksidanların sınıflandırılması	2
Şekil 3.1. Bir perceptron şekil yapısı	7
Şekil 3.2. Yapay zeka, makine öğrenmesi ve derin öğrenme karşılaştırması	19
Şekil 3.3. Makine öğrenmesi ve derin öğrenme farkı	22
Şekil 3.4. LSTM mimarisi [24]	24
Şekil 3.5. Birden fazla gizli katman içeren bir derin oto-kodlayıcı yapısı	27
Şekil 4.1. Modelin özeti	41
Şekil 4.2. Modelin yapısını gösteren diyagram	43
Şekil 5.1. Beş (5) epoch için grafik	41
Şekil 5.2. Yirmi (20) epoch için grafik	41
Şekil 5.3. Yüz (100) epoch için grafik	42
Şekil 5.4. İki yüz elli (250) epoch için grafik	42
Şekil 5.5. Yüz (100) epoch için grafik (İngilizce dili için)	45
Şekil 5.6. Yüz (100) epoch için grafik (Türkçe dili çok veri için)	47

TABLÖLAR LİSTESİ

Tablo 3.1. Kullanılan sistem ve program	42
---	----



ÖZET

Anahtar Kelimeler: Metin Özetleme, Derin Öğrenme, Yapay Sinir Ağları, Vektör, Keras, Yapay Zeka, Makine Öğrenmesi

Günümüzde teknolojinin gelişimi ile birlikte yapay zekanın uygulamaları hayatımızda yaygınlaşmaya başlamıştır. Sürücüsüz araçlar, sesli komut algılayan sistemler, değişik mimik ve tepkiler veren yardımcı robotlar, drone vb. benzeri araçlar hayatlarımıza girmekte ve daha pek çoğu giriş yapmaya devam etmektedir.

Yapay sinir ağı, beyinde bulunan nöronlardan yola çıkarak tasarlanmış basit işlemcilerdir. İnsan beyninin öğrenme şekli esas alınarak geliştirilmiştir. Belirli karmaşık problemlerin çözümü için aynen sinir hücrelerimizde olduğu gibi bu düğümler arasında da bilgi akışı ve öğrenmeyi sağlayan bağlantılar vardır. Beyindeki gibi eğitilerek testler sonucunda doğruluk oranı saptanmaktadır.

Metin özetleme, bir belgeyi girdi olarak alıp daha kısa, yalın ve anlaşılır hali ile kullanıcıya sunmaktır. Günlük hayat koşturmacasında gerekliliği tartışılmaz bir konudur. Dergi, gazete, makale gibi günlük uygulamalar özetlemenin faydasının üst düzeyde görülebileceği alanlardır.

Metin özetleme ile ilgili ilk çalışmalar yaklaşık elli yıl önce İngilizce için yapılmıştır. İlk zamanlar çeşitli istatistiksel yöntemler kullanılmış olup, bu yöntemler yüksek başarımlı ve düşük maliyet hedefini tutturduğu için geçerliliğini korumaktadır. Türkçe sondan eklemeli ve kurallı bir dil olduğu için, dil seçimi konusunda biçilmiş kaftan olmuştur. Bu tez çalışmasında, yapay zekanın alt dallarından biri olan makine öğrenmesinin dalı olan, derin öğrenme kullanılarak metin özetleme ele alınmış ve Tensorflow alt yapısını kullanan Keras kütüphanesi kullanılarak girilen metnin bir özeti elde edilmiştir.

Türkçe metin özetleme için hali hazırda çok az veri seti vardır. Genelde veri setleri İngilizce ağırlıktadır. Bu tez çalışmasında çeşitli verisetlerinden alınarak Türkçe ve İngilizce için veri setleri oluşturulmuştur ve ilgili diller için metin özetleme çalışmalarına yer verilmiştir. Yapılan tez çalışmasının literatüre önemli derecede katkılar sağlayacağı düşünülmektedir.

TEXT SUMMARIZATION WITH DEEP LEARNING

SUMMARY

Keywords: Text Summarization, Deep Learning, Artificial Neural Networks, Vector, Keras, Artificial Intelligence, Machine Learning

Today, with the development of technology, applications of artificial intelligence have started to become widespread in our lives. Driverless vehicles, systems that detect voice commands, auxiliary robots that give different gestures and responses, drone etc. similar vehicles enter our lives and many more continue to enter.

Artificial neural network are simple processors designed based on neurons in the brain. It has been developed based on the learning style of the human brain. Just as in our nerve cells, there are connections that provide information flow and learning between these nodes for the solution of certain complex problems. By training as in the brain, the accuracy rate is determined as a result of tests.

Text summarization is to take a document as input and present it to the user in a shorter, simpler and more understandable form. The necessity in daily life is an indisputable issue. Daily applications such as magazines, newspapers and articles are areas where the benefits of summarizing can be seen at a high level.

The first studies on text summarization were done in English about fifty years ago. Various statistical methods were used for the first time, and these methods remain valid because they meet the high performance and low cost targets. Since Turkish is an additive and canonical language, it has been cut out for language selection. In this thesis, text summarization, which is the branch of machine learning, which is one of the sub-branches of artificial intelligence, is discussed using deep learning and a summary of the text entered using the Keras library using the Tensorflow infrastructure is obtained.

Very few data sets are currently available for Turkish text summarization. Generally, data sets are in English. In this thesis study, data sets for Turkish and English have been created from various databases and text summarization studies for related languages are included. It is thought that the thesis study will contribute significantly to the literature.

BÖLÜM 1. GİRİŞ

Özet, bir veya aynı konu ile ilgili birden fazla dokümandan çıkarılan ve kaynağındaki en temel bilgiyi içeren metin parçasıdır [1]. Bir dokümandan bilgisayar ortamında otomatik olarak özet çıkarılmasına Otomatik Metin Özetleme (OMÖ) denilmektedir. OMÖ olarak adlandırılmaktadır. Tekli ve çoklu olmak üzere iki türü vardır. Tekli doküman özetlemede bir kaynak mevcuttur. Çokluda ise birbiri ile ilgili birden çok kaynak vardır.

Metin özetleme; cümle seçerek ve cümle yorumlayarak özetleme olmak üzere, iki farklı türde ele alınmaktadır. Bunlar:

Cümle Seçerek Özetleme (Extractive): Önemli cümleler üzerinden gidilerek, istatistiki ve sezgisel vb. yöntemler ile bir özet çıkarılmaktadır. Var olan cümle yapısı bozulmadan cümleler seçilerek çıkarılmakta ve bir özet elde edilmektedir.

Yorumlayarak Özetleme (Abstractive): Özet, belirli bir mantık çerçevesinde metnin yorumlanması ile yapılmaktadır. İlk metindeki ifadeler daha kısa hale getirilerek tekrardan yeni bir özet çıkarılmaktadır [2]. Örneğin “Ahmet elmadan, portakaldan ve armuttan nefret eder” ifadesi “Ahmet meyveden nefret eder” şeklinde özetlenmektedir. Bu tarz bir özetleme için geniş kelime dağarcığı gereklidir.

İnsanoğlunun kendi karar verme sistemini kopya ederek, bu şekilde çalışan modeller ve simülasyonlar üretme fikri geçmişten beri süregelen bir durumdur. Yapay zekanın temeli insanın düşünce, öğrenme ve karar mekanizmalarının taklit edilerek belli cihaz ve makineler üzerinde modellenmesidir. Yani insan beyninin çalışma sisteminin aynen aktarılmasıdır. Son yıllarda güzel gelişmeler olsa da, insan zekasının tamamen taklidi günümüzde oldukça uzak görünen bir kavramdır.

Ancak ilerleyen süreçte durum değişkenlik gösterecek gibi görünmektedir. İnsanların yirmi birinci yüzyılda kendisinden büyük ölçüde daha akıllı makineler yapacağı konusunda çok az şüphe vardır [3].

Yapay zekaya dayalı yöntemler 2000'li yıllarda önemli derecede sekteye uğradıktan sonra, derin öğrenme yaklaşımları ile bu yöntemler tekrar gündeme gelmiştir. Donanımsal gelişmelerin yetersizliği sebebiyle oluşan bu durgunluk dönemi, GPU benzeri hesaplama yöntemleri ile tekrardan normale dönmüştür. Bu hesaplama işleri yapay sinir ağlarındaki gizli katman ve düğümler üzerinedir.

Yapay zeka, mantıksal problemler konusunda başarılı olsa da; doğal dil işleme, konuşma tespiti, nesne tespiti, yüz tanıma vb. problemlerin çözümünde yeterli olmamıştır. Yapay zeka alanındaki gelişmelerle birlikte, özellikle makine öğrenmesi ve derin öğrenme gibi uygulamalarla bu tip problemlerin çözümünde önemli bir yol alınmıştır.

Makine öğrenmesi; çeşitli matematiksel yöntemler kullanarak mevcut verilerden bilgi edinen ve edindiği bu bilgi ile tahminler yapabilen yöntemler bütünü olarak ele alınmaktadır [4]. Büyük miktardaki verinin analiz ve işlenmesi mümkün olmadığı için, eldeki veriler ile yeni veriler elde etmek amacıyla makine öğrenmesi yöntemleri geliştirilmiştir. Makine öğrenmesi var olan verileri temel alarak yeni veriler için en iyi modeli bulmaya çalışmak üzere kurgulanmıştır. Makine öğrenmesi algoritmaları, ilgili deneyimi temsil eden verilerdir. Örnek olarak yüz tanıma, arama motorları, spam filtreleme, belge sınıflandırma, sosyal medya platformları, nesnelerin interneti vb. verilebilmektedir.

Derin öğrenme, makine öğrenmesi ailesindendir. Son senelerde dünya üzerinde oldukça bilinen bir hale gelmiştir. Derin öğrenme yapay sinir ağları üzerinde gerçekleştirilen öğrenme teknikleri felsefesidir [5]. Derin öğrenme insan beyninin nasıl çalıştığını simule etmek için yapay sinir ağlarını kullanmaktadır. Bu yapay sinir ağları, yüksek miktarda veriyi işlemek üzere birkaç nöronlu küçük ağlardan binlerce

nöronlu çok büyük olanlarına kadar bir aralıkta uzanmaktadır. Amaç öğrenme algoritmalarını daha iyi ve daha kolay olacak şekilde kullanılabilir yapmaktır [6].

Makine öğrenmesinde sadece bir ve sıfırdan oluşan sonuçlar elde edilmesine rağmen, çok katmanlı yapay sinir ağlarının yapay zeka alanına en önemli katkısı, bir ve sıfırdan başka ara değerlerde cevaplar üretmesidir. Derin öğrenme 0.3, 0.8 gibi sonuçlar döndürebilmektedir. Bunun bize en büyük katkısı problemi daha detaylı analiz edip, ayrıntılı sonuçlar elde edebilmemizdir. Örneğin, makine öğrenmesinde cevaplar “Hava güneşli”, “Hava bulutlu” şeklinde iken, derin öğrenmede “Hava öğleye kadar güneşli”, “Akşam saatlerinde sağanak yağış var” ya da “Bugün hava az bulutlu ve açık geçecek” şeklinde ayrıntılı ve problemin çözümü odaklı ifadeler olacaktır.

Bu çalışmada, makine öğrenmesinin alt dalı olan derin öğrenme kullanılarak metin özetleme konusu ele alınmıştır. Sistemde, Türkçe için farklı büyüklükte veri setleri kullanılmış ve ilgili sonuçlar karşılaştırmalı olarak incelenmiştir. İngilizce dili ile de anadilimiz Türkçe, dil karşılaştırması yapılarak ilerlenmiş ve ilgili sonuçlar elde edilmiştir.

1.1. Tezin Amacı

Bu tezin amacı; derin öğrenme kullanılarak, başarıyı yüksek metin özetleri elde edecek bir model geliştirmek ve geliştirilen model ile de akademik alana katkıda bulunmaktır.

1.2. Tezin Önemi

Daha yeni bir alan olan derin öğrenme konusunda, günlük hayatta da kullanılabilen bir çözüm geliştirmektir. Ayrıca anadilimiz Türkçe’yi temel alan bir çalışma olması da tezin önemini artırmaktadır.

1.3. Tezin Yöntemi

Bu amaca ulaşmak için:

- Metin özetleme yöntemlerinin incelenmesi,
- Yapay sinir ağlarının analizi,
- Derin öğrenme mimarileri ve uygulamalarının anlatımı,
- Derin öğrenme kütüphanelerinden bahsedilip, en uygun kütüphanenin seçimi,
- Modelin uygun veri setleri ile, uygun epoch sayısında eğitilmesi,
- Sonuç olarak grafiklerin ve ilgili özet metinlerinin elde edilmesi,

Şeklinde aşamalardan geçilmiştir.

1.4. Tezin Organizasyonu

Bu çalışma beş bölümden oluşmaktadır. Birinci bölümde teze ilgili temel kavramlar, tezin amacı, önemi, yöntemi ve organizasyonu açıklanmıştır. İkinci bölüm olan literatür araştırması başlığı altında, derin öğrenme ile metin özetleme hakkında literatüre yer verilmiştir. Üçüncü bölümde, metin özetleme kavramı, derin öğrenme, yapay sinir ağları ve yapay sinir ağları - derin öğrenme ilişkisi ele alınmıştır. Dördüncü bölümde ilgili matematiksel modelin eğitimi teorik olarak anlatılmıştır ve hesaplamaları sunulmuştur. Beşinci bölümde ise tez ile ilgili genel sonuçlar ve karşılaştırmalar verilmiş, bu çalışmanın akademik camialar için faydası ve literatüre katkısı ifade edilmiştir.

BÖLÜM 2. LİTERATÜR ARAŞTIRMASI

Doğal dil, insanların kendini ifadesi ve çevreyle iletişimi için bir araçtır. Chomsky, dilin çocukluk zamanındaki halinden, doğal dile dönüşümü sürecinin, genetik yapıyla bağlantılı olduğunu ifade etmektedir [7].

Zaman içinde, kural tabanlı yaklaşımlar ve istatistiksel yaklaşımlar gibi yöntemler, dil kullanımında gerçekleştirilmeye çalışılmıştır. Doğal Dil İşleme (DDİ), hedef doğal dilleri otomatik olarak oluşturma ve anlama konusunda oluşan problemlerle ilgilenmektedir [8]. DDİ, insan üretimi olan ses ve metinleri işlemekte ve insan-bilgisayar etkileşimine yardımcı olmaktadır.

Sosyal medya kullanımının her türlü yaş grubunu kapsamı vb. sebeplerle üretilen veri miktarı gün geçtikçe artmaktadır. Ancak doğal olarak oluşturulan bu veriler, doğrudan işlenememektedir. Bu verileri kullanabilmek için birçok alanın beraber çalışması gerekmiştir. Buradaki amaç insan-makine iletişimini sağlamaktır. Önceleri DDİ çalışmaları, tek tek sözcükler şeklinde iken, 19. yy. sonlarına doğru sözcüklerin ilişkisi ve bütünsellik çalışmaları yapılmıştır [9].

Metin özetleme de, doğal dil işlemenin bir alt başlığıdır. Geçen zamanla birlikte, metin özetlemede de, çeşitli yöntemler denenmiş ve kullanılmıştır. Derin öğrenme metin özetlemede kullanımı da son zamanlarda artış göstermiştir.

Derin öğrenme kavramı, bugüne gelene kadar literatürde farklı farklı kaynaklarda pek çok farklı şekilde tanımlanmıştır. Zhang ve arkadaşları derin öğrenmeyi, bilgisayarların deneyimlerden öğrenmelerini ve dünyayı kavramların hiyerarşisi bakımından anlamlandırmalarını sağlayan bir makine öğrenimi olarak nitelendirmiştir [10]. Deng ise, denetimli veya denetimsiz özelliklerin çıkarılması, dönüştürme, desen analizi ve sınıflandırma gibi doğrusal olmayan gizli katmandan

yararlanan bir makine öğrenmesi teknikleri sınıfı olarak ifade etmiştir [11]. Genel anlamda derin öğrenme, makinelerin dünyayı algılaması ve anlaması için kullanılmakta olan bir yaklaşımdır.

Derin öğrenmenin geçmişi, McCulloch ve Pitts'in, 1943 senesinde düşünce sürecini taklit adına sinir ağları için bir hesaplama modeli sunmasına kadar gitmektedir [12].

1969'da Bryson ve Ho, çok katlı dinamik sistem optimizasyonu olarak geri yayılımı tanımlamışlardır. Çok tabakalı yapay sinir ağları için öğrenme algoritmasının gücü, büyük ağların eğitimi için yeterli olduktan sonra, 2000 ve 2010'lu yıllarda derin öğrenmenin başarımına büyük ölçüde katkı sağlamıştır. 2006 senesinde ise Hinton, çoklu öğrenme tanımını ortaya atmıştır. Bu terim derin öğrenmeye gerçek anlamda ilk bakış olarak görülmektedir [13].

Doğal dil işlemede, özellikle yinelenen sinir ağları ve evrimsel sinir ağlarının başarılı sonuçlar verdiği bilinmektedir. Bahsi geçen bu iki yöntem dahil olmak üzere, diğer derin öğrenme yöntemlerinin de doğal dil işlemede başarılı sonuçlar verebileceği, literatürdeki çeşitli çalışmalarla ispatlanmıştır [14].

Bu tez çalışmasında, derin öğrenme mimarilerinden derin oto-kodlayıcılar ve LSTM kullanılmıştır. Temel yapı derin otokodlayıcı olmak üzere, LSTM yoluyla da performans artışı sağlanması hedeflenmiştir.

BÖLÜM 3. MATERYAL VE YÖNTEM

3.1. Metin Özetleme Kavramı

Günümüz dünyasında bilgisayar bilimlerinin hızla gelişimi, faydalarının yanında bir takım problemleri de beraberinde getirmektedir. Bunlardan biri de ilgili dökümanlara en yalın hali ile ulaşmaktır. Bu kapsamda metin özetleme çalışmalarının önemi ortaya çıkmaktadır.

Özet, bir konu hakkında bir ya da daha çok kaynaktan elde edilen en temel bilgi parçasıdır. Metin özetleme, günümüz bilgi çağında kullanılan ve hayatı kolaylaştırıcı önemli araçlardandır. Metin özetleme kavramı, doğal dil işleme başlığı altında incelenmektedir.

3.1.1. Metin özetleme üzerine yapılan çalışmalar

Doğal dil işleme çalışmaları ana ve ara uygulamalar olarak iki gruba ayrılmaktadır. Ana uygulamalar; bilgisayarla çeviri, otomatik özetleme, bilgi çıkarımı, bilginin yeniden elde edilmesi gibi kendi başına bir uygulama oluşturan örneklerdir. Ara uygulamalar ise; tümceyi öğelerine ayırma, çözümleme, biçimbilimsel analiz (sözcük ek ve köklerini bulma), sözcük anlamını belirginleştirme gibi ana uygulamalar için gerekli işlemleri gerçekleştirmektedir [15].

İnsanoğlu için manuel olarak büyük metinleri özetlemek çok zordur. İnternette metin materyali bolluğu vardır. Bununla birlikte, genellikle internet ihtiyaç duyulandan daha fazla bilgi sağlamaktadır. Fakat bu durum, beraberinde iki problemi de getirmektedir: varolan çok sayıda dökümanı araştırmak ve büyük miktardaki veriden ilgili bilgiyi edinmek [16].

Tüm bu sebepler sonucunda, sürecin hızlandırılması adına, Otomatik Metin Özetleme kavramı ortaya çıkmıştır. Otomatik Metin Özetleme, en önemli bilginin çeşitli amaçlar doğrultusunda çeşitli kaynaklardan damıtılma işlemidir. Hedef, kaynak metni kısa versiyona ilgili bilgi içeriğini ve nihai anlamını koruyarak dönüştürmektir.

Otomatik Metin Özetleme Sistemi, birçok evrak ve verinin işlenmesi gerektiği durumlarda son kullanıcıya çok yardımcı olmaktadır. Ayrıca birden fazla evrak üzerinde de çalışabilmektedir [17].

Genel anlamda, metin özetleme işleminde özet çıkarmak için kullanıcıdan belli veri girişleri (anahtar sözcük vb.) istenebilmektedir. Veya herhangi bir sınırlama işlemi yapılmamaktadır.

3.1.2. Metin özetlemede kullanılan yöntemler

Genel anlamda, metin özetleme işlemlerinde en sık kullanılan yöntem cümle puanlama yöntemidir. Belirli kıstaslar altında gerçekleştirilmektedir. Bütünden parçaya gidilmektedir. Metin paragraflara, paragraflar cümlelere, cümleler sözcüklere ayrılarak özetleme gerçekleştirilmektedir.

Otomatik metin özetlemenin, özetleyici ve çıkarımsal olmak üzere iki türü vardır.

Çıkarımsal metin özetlemenin okunabilirliği yüksektir. Çıkarımsal yöntemde, kelimeler, kelime öbekleri ve cümleler alınarak işlem yapılmaktadır. İstatistiksel ve sezgisel yöntemler kullanılmaktadır.

Çıkarımsal metin özetlemenin de iki türü vardır:

- Kelime Frekans Değerine Göre Özetleme
- TextRank Algoritmasına Göre Özetleme

Bir diğerk yöntem olan özetleyici yöntemde, metin kısaltılarak yeniden yorumlanmakta ve yeni bir özet eldesi gerçekleştirilmektedir. Örnek verecek olursak, “Ayşe sarı, turuncu ve kırmızı renkleri tablosunda kullandı” metni özet olarak “Ayşe sıcak renkleri tablosunda kullandı” şeklinde ifade edilmektedir. Bu yöntemde geniş kapsamda kelime bilgisi kullanılmaktadır.

Otomatik metin özetlemede kullanılan ve öne çıkan aşamalar genel manada şu şekildedir:

- Cümle içinde başlıktaki sözcükler aranmaktadır.
- Cümlede tarih içeriğı ve özel isim olup olmadığı incelenmektedir.
- Pozitif ve negatif sözcük taraması yapılmaktadır. Pozitif sözcük özetle, sonuç olarak gibi ifadeler, negatif sözcük ise çünkü, ancak gibi ifadelerdir. Pozitif ifadeler metni toparlayıcı cümlelerde, negatif ifadeler ayrıntılı bilgi içeren cümlelerde bulunmaktadır.
- Anahtar sözcük kontrolü yapılmaktadır.
- Cümle konumları göz önünde bulundurulmaktadır. İlk ve son kısımlardaki cümleler önceliklidir.
- Metin içindeki sözcüklerin frekansları bulunmaktadır. Frekans, sözcüğün metinde bulunma sıklığı olarak ifade edilmektedir. Bu sözcükler bir sıralamaya tutulmaktadır. Bu listenin %10’u kullanılmaktadır. Sözcükler sıklığına göre artı puan almaktadır.
- Anlamı pekiştiren sözcüklerin de kontrolü yapılmaktadır. Bunlar destekleyici ifadelerdir.
- Cümlelerin bitişindeki noktalama işaretleri de göz önünde bulundurulmaktadır. Ünlem vb gibi ekstra öneme sahip olan noktalama işaretleri içeriyorsa bu cümle artı puan almaktadır.
- Cümlelerin ortalama uzunluklarına bakılmaktadır. Ortalamadan sapmalara göre cümlelere puanlar verilmektedir.
- Cümle içindeki varlık adları seçilmektedir. Var ise bu ölçüde puanlamalar yapılmaktadır.

Genel hatları ile bu gibi yöntemler kullanılmakla beraber, yapılan çalışmanın niteliğine göre daha farklı kıstaslar kullanılarak daha verimli özetler de elde edilebilmektedir.

3.2. Yapay Sinir Ağları

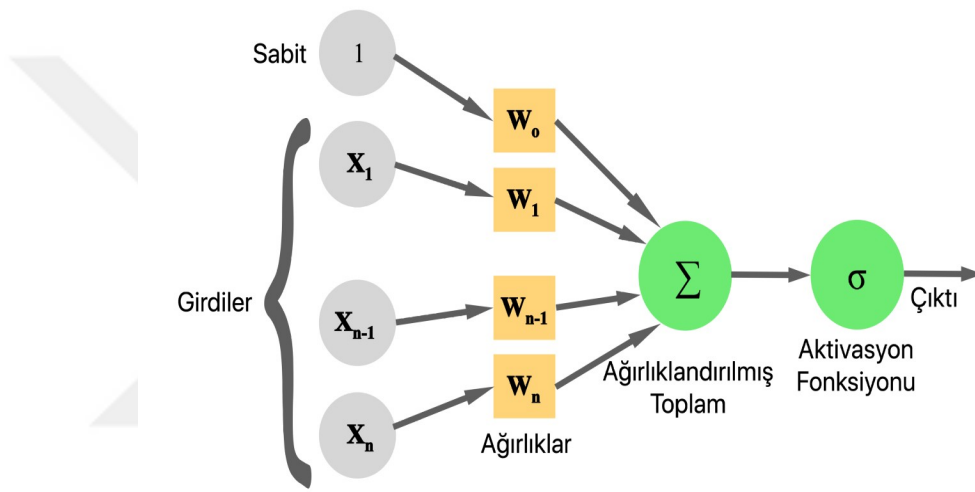
Bilindiği üzere; düşünme, hatırlama ve öğrenme vb. davranışların temel yapı taşı sinir hücreleridir. Beynimizde yaklaşık 10^{11} adet sinir hücresi olduğu düşünülmektedir ve bu hücreler arasında sonsuza yakın sayıda sinaptik bağlantı vardır.

İnsan beyni ve beynin nörofiziksel yapısı örnek alınarak matematiksel olarak modeli elde edilmiş ve beynin davranışsal özelliklerini modellemek adına fiziksel bileşenler oluşturularak yapay hücreler ve ağ modelleri ortaya çıkmıştır. Böylece yapay sinir ağları geliştirilmiştir.

Yapay sinir ağları, insan beyni ve sinir sisteminden ilham alınarak tasarlanmış bir yapay zeka algoritmasıdır. Yapay sinir ağları beynin herhangi bir işlevini gerçekleştirme amacı taşımaktadır. Sinir hücrelerinin birbiri ile çeşitli bağlantılar oluşturması ile sistem işler ve katman yapısı vardır. Modern bilgisayarlar sayısal hesaplamalarda doğruluğu yüksek sonuçlar verirken, beynimizin kalabalık içinden birini tanıması gibi algı yönlü problemlerde insanın çözme yeteneğinin oldukça gerisinde kalmaktadır.

Yapay sinir ağlarının en önemli özelliği, deneyimlerden (tecrübe) yararlanarak öğrenebilmesidir. Yapay sinir ağları, insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilmişlerdir. Öğrenmenin yanı sıra bilgiler arasında ilişkiler oluşturma yeteneğine de sahiptir [18].

Bir sinir ağı, bilgiyi depolamak ve onu kullanışlı hale getirmek için doğal eğilimi olan basit birimlerden oluşan paralel dağıtılmış bir işlemcidir [19]. Sinir ağı, işte bu işlemcilerden oluşmaktadır. Nöronlar, sinyalleri nörondan nörona aktaran ağırlıklandırılmış bağlantılar ile birbirine bağlanmaktadır. Hücreler arası iletişim sinapslar aracılığı ile gerçekleşmektedir. Sinir ağları bir özellikler vektörünü girdi almakta ve tek bir sonuç ya da ilgili verinin vektörünü sonuç olarak vermektedir. Bu sonuç ilgili bağlantılar üzerinden diğer hücelere yollanmaktadır.



Şekil 3.1. Bir Perceptron Şekil Yapısı.

Nöron yapısı incelendiğinde, girdilerin olduğu düğümler, girdilerin ağırlıkları, girdi ve ağırlıkların çarpılıp toplanması sonucu elde edilen toplama fonksiyonu yine bu ilgili fonksiyondan gelen değerlerin belirli bir aralıkta standartlaştırıldığı aktivasyon fonksiyonu ve çıktılarından meydana gelmektedir. Bir perceptron yapısı Şekil 3.1.'de gösterilmiştir.

Çok katmanlı yapay sinir ağları temel manada 3 katman içermektedir. Bu katmanlar sıralı bir model oluşturmaktadır. Bu üç katman şunlardır:

1. Girdi katmanı
2. Gizli katmanlar (Ara katmanlar)

3. Çıktı katmanı

Girdi katmanı, yapay sinir ağına gelen bilgileri tutmaktadır. Veri seti baz alınarak ilgili özelliklerin herbiri giriş katmanında bir düğüm olarak gösterilmektedir. İlgili girdilerin hepsinin belirli ağırlık değerleri vardır. Bu girdiler ara katmandaki düğümlere bu ağırlıklar yoluyla bağlanmıştır. Ağırlıklar ilgili özelliğin önem derecesi vermektedir.

Gizli (ara) katmanlarda, girdi katmanından gelen veriler işlenip bir çıktıya dönüşmektedir. Bu dönüşüm ağırlık değerleri kullanılarak yapılmaktadır. Problemin zorluk derecesi ara katman sayısında belirleyici etken olmaktadır.

Çıktı katmanı, çıktı değeri ya da değerlerinin tutulduğu yerdir. Ara katmandan gelen sonuç verisi çıktı değeri olarak ifade edilmektedir. Hesaplanan ya da beklenen değerler arası fark “hata fonksiyonu”na göre hesaplanmaktadır. İlgili hata değeri sonuca ne kadar yakın olup olmadığını göstermektedir. Hesap edilen değer baz alınarak “optimizasyon fonksiyonu” aracılığı ile ağırlıklar güncellenmektedir. Aslında bu bir öğrenme işlemidir. Yapay sinir ağlarında öğrenme işlemi işte bu şekilde ağırlıkları güncelleme yoluyla başlamaktadır. Devamında hata optimize edilmektedir. Eğer beklenen sonuçları veren ağırlıklara ulaşılır ise sistem eğitilmiş demektir. Ağırlıkların güncellemesi için iki temel yöntem vardır. Bunlar ileri beslemeli ağlar ve geri beslemeli ağlar olarak sınıflandırılmıştır. Bu kısma yapay sinir ağlarının türleri kısmında değinilmiştir.

Yapay sinir ağlarının başarısına etki eden pek çok parametre vardır. Başarı için ilgili parametrelerin düzgün bir şekilde optimize edilmesi gerekmektedir. Bunlardan başlıcaları şunlardır:

- Örnek sayısı
- Veri normalizasyonu
- Katman sayısı
- Girdi ve çıktı değerlerinin uygun temsili

- Doğru aktivasyon fonksiyonlarının seçimi
- Öğrenme oranı
- Hata hesaplama ve hata optimizasyonu fonksiyonlarının doğru şekilde seçimi

gibi parametreler ilgili başarıya etki etmektedir. Farklı problemler için farklı parametreler de mevcuttur.

Yapay sinir ağı oluştururken veri setinin doğru şekilde analizi yapılarak, ilgili verilerin nasıl temsil edileceği, ilgili yöntemler ve temel manada hangi ağ yapısında olacağı güzelce belirlenerek ağ oluşturulmalıdır.

Yapay sinir ağları, birçok başarılı metasezgisel gibi; tahmin etme, tanımlama ve sınıflandırma vb. gerçek dünya problemlerinde iyi sonuçlar vermektedir.

3.2.1. Yapay sinir ağlarının türleri

Tek katmanlı sinir ağları ve çok katmanlı sinir ağları olarak iki ana türe ayrılmaktadır.

3.2.1.1. Tek katmanlı sinir ağları

Tek katmanlı yapay sinir ağları yalnızca girdi ve çıktı katmanından oluşmaktadır. Doğrusal problemleri çözümlemeye yaramaktadır. Katmanlarda birden fazla nöron da bulunabilmektedir.

Tek katmanlı sinir ağlarında genel anlamda iki model vardır.

- Perceptron Modeli
- Adaline / Madaline Modeli

3.2.1.2. Çok katmanlı sinir ağıları

Tek katmanlı sinir ağıları doğrusal olmayan problemlerde başarısız olduktan sonra çok katmanlı algılayıcılar geliştirilmiştir. Bu sinir ağıları bir girdi, bir ya da birden çok ara katman ve bir çıktı katmanından oluşmaktadır. Katmanlar arasında geçişler vardır. İleri ve geri yayılım olarak adlandırılan iki türü vardır. İleri yayılımda ağıın çıktısı ve hata değeri bulunmaktadır. Geri yayılımda ise hesaplanan hata değeri azaltılmak üzere uğraşmakta ve bunun için ilgili bağlantıların ağırlık değeri güncellenmektedir.

Çok katmanlı sinir ağıları modeli doğrusal olan perceptron modelindeki en küçük kareler algoritmasının genelleştirilmiş hali olan geri yayılım (backpropagation) öğrenme algoritmasını baz almaktadır.

3.2.2. Yapay sinir ağıları nasıl çalışır?

Yapay sinir ağıları, insanlara benzer şekilde örnekler sayesinde öğrenmektedir. İnsanlarda sistem nasıl ki sinapslar arası boşluklardaki elektriksel öğrenmeler ile işliyorsa, yapay sinir ağıları da tekrarlı girdiler sayesinde yapılanma ve ağırlıklarını ayarlamaktadır. Aynı bizlerin sinir sistemi gibi ortama adapte olan yapıdadır. Yani çeşitli uyaranlara göre yapısını değiştirmekte ve bu sayede öğrenmektedir. Karar aşamasında bağlantıların ağırlıkları da hesaba katılmaktadır. Her işlem tek başına işleme izlenimi verse de arka planda bir çok yapay sinir ağı aynı anda işlemekte ve dağınık paralel hesaplama örneği göstermektedir.

Ancak hesaplamadaki durum bazı değişim çeşitli sorunlara yol açabilmektedir. Bilgisayar sistemleri ne yapılması söylenirse onu yaparken, burda kendi çözümünü kendisi bulduğu için sistemin bazı durumlarda nasıl davranacağı bilinmemektedir. Bu da sisteme bir bilinmezlik ve tahmin edilemezlik özelliği kazandırmaktadır.

3.2.3. Yapay sinir ağlarının özellikleri

Yapay sinir ağları gücünü, paralel yapısından, öğrenebilme ve genelleme yapma yeteneklerinden almaktadır. Genelleme yapma yeteneği, karşılaşılmayan girişler için de uygun ve işe yarar tepkiler üretmeye yaramaktadır. İşte bu sebeptir ki yapay sinir ağları karmaşık problemlerin çözümü konusunda başarılı profil çizmektedir. Yapay sinir ağlarının özellikleri genel anlamda şunlardır: doğrusal olmama, öğrenme, genelleme, uyarlanabilirlik.

3.2.3.1. Doğrusal olmama

Yapay sinir ağlarının yapı taşı olan hücre doğrusal olmadığından bu hücrelerin birleşmesinden olan yapay sinir ağları da doğrusal olmayan yapıdadır. Bu yüzden doğrusal olmayan problemlerdeki en önemli çözüm yöntemidir.

3.2.3.2. Öğrenme

Yapay sinir ağlarının istenen şekilde davranması için amaca uygun şablonda ayarlanmış olması gerekmektedir. Bu da ancak hücrelerin birbirine doğru bağlantılar ile bağlı olması ve bu bağlantıların doğru ağırlıklara sahip olması ile olabilmektedir. Tüm bu ayarlamalar var olan karmaşık yapı nedeni ile önceden belirlenememektedir. Bu sebeple yapay sinir ağları alınan örnekler kullanılarak problemi öğrenmelidir.

3.2.3.3. Genelleme

Yapay sinir ağları, ilgili problemi öğrendikten sonra daha önce karşılaşmadığı örnekler için de beklenen tepkileri verebilmektedir. Örneğin; karakter tanımlama işleminde, bozuk karakterlerin girişlerinde doğru karakterlerin eldesi.

3.2.3.4. Uyarlanabilirlik

Yapay sinir ağıları, ilgili problemin yapısı ölçüsünde değişimlere göre ağırlıklarını da değiştirmektedir. Bu tekrar eğitim şeklinde olabilir veya değişimler sürekli devinim halinde ise gerçek zamanlı şekilde eğitime devam edilmektedir. Bu özellik yapay sinir ağlarının örnek tanıma, sinyal işleme vb alanlarda etkin olarak kullanılmasını sağlamaktadır.

3.2.4. Yapay sinir ağlarının avantaj ve dezavantajları

Yapay sinir ağlarının insan beynini taklidi bilinmekle beraber, çeşitli avantajları ve halen üstesinden gelinmeye çalışılan dezavantajları vardır.

Avantajları şu şekildedir:

- Bilgiler ağına tamamına yayılarak depolanmaktadır.
- Bir bilgi eksikliği olduğu zaman bu durum ağına çalışmasını engellememektedir. Ağına performansı eksik bilginin önemine göre değişmektedir.
- Yapay sinir ağlarının bir ya da birden fazla hücresinin bozulmaya uğraması çıktıyı engellememektedir.
- Sistem dağıtık hafızaya sahiptir. Sistemin işlemesi için belirli örnekler baz alınmaktadır. Eğer bütün yönler ilgili örneklerde gösterilmezse ağ yanlış çıktılar verebilmektedir.
- Ağına bozulması zamanla yavaş ve göreceli olarak gerçekleşmektedir. Problem ortaya çıktığı gibi bir bozulma gerçekleşmemektedir.
- Yapay sinir ağları olayları öğrenmekte ve benzer olaylar ile karşısında yorumda bulunarak bir karar vermektedir.
- Paralel işlem yeteneği vardır. Yapay sinir ağları çok çekirdekli işlemci gibi çalışarak, birden fazla işi senkronize şekilde gerçekleştirecek sayısal güce sahiptir.

Dezavantajlar da şu şekilde sıralanabilir:

- Çıkan sonucun açıklaması yapılamamaktadır. Bu yüzden sistem “karakutu” olarak ifade edilmektedir.
- Uygun ağ yapısı deneyim ve deneme yanılma yolu kullanılarak belirlenmekte olduğundan belirleme işlemi belirli bir kural içermemektedir.
- Yapay Sinir Ağları paralel işlem gücüne sahip işlemciler üzerinden performanl çalıştığı için donanıma bağımlı bir yapı göstermektedir.
- Yapay sinir ağları yapısı gereği nümerik değerler üzerinden çalışmakta olduğundan, problemler ağlara tanıtılmadan önce nümerik değerlere çevrilmektedir.
- Ağın eğitiminin süresinin ne zaman bitirileceği konusunda belirlenmiş bir yöntem bulunmamaktadır.
- Ağın parametre değerleri ve eğitim örneklerinin belirlenmesi konusunda belirli ve genel bir kural bulunmamaktadır.

3.2.5. Yapay sinir ağlarında öğrenme türleri

Yapay sinir ağlarının belirli girdileri alarak çıktı üretmesi ağın öğrenmesi ile olmaktadır. Bu öğrenmenin birden fazla yöntemi vardır. Bunlar danışmanlı, danışmansız ve takviyeli öğrenme olarak üç türdür.

3.2.5.1. Danışmanlı öğrenme

Danışmanlı öğrenmede, yapay sinir ağı kullanılmadan eğitilmesi gerekmektedir. Bu işlem sinir ağına giriş ve çıkış bilgileri sunmakla başlamaktadır. Bu bilgiler genel manada eğitme kümesi olarak adlandırılmaktadır [20].

Ağa verilen giriş değerlerine karşılık olarak çıktı değerleri de bilindiği zaman danışmanlı öğrenme ihtiyacı doğmaktadır [21]. Ağ ağırlıkları güncelleyerek, verilen girdiler için istenen çıktıları oluşturmaktadır. Ağın çıktıları ile beklenen çıktılar arası hata hesap edilerek ağın yeni ağırlık değerleri bu hata payına göre ayarlanmaktadır. Yine benzer şekilde aradaki fark hesabı ile ilgili farka göre nöron başı hata payı bulunmaktadır. Daha sonra her nöron kendine gelen ağırlıkları güncellemektedir.

3.2.5.2. Danışmansız öğrenme

Danışmansız öğrenmede ağa öğrenme esnasında örnek girdiler sunulmaktadır. Herhangi bir beklenen çıktı değerimiz yoktur. Bu yönde bir girdi de yapılmamaktadır. Girişteki bilgilere göre ağ her bir örneği kendi içinde sınıflandırmaya tabii tutmakta ve buna göre kurallar oluşturmaktadır. Ağ ilgili ağırlık değerlerini aynı özellikteki dokuları ayırarak düzenlenmekte ve öğrenme işlemi tamamlanmaktadır.

Bu metod, yapay sinir ağlarında öğrenme için sürekli araştırılan ve gelişen bir metottur. Gelecekte makinelerin, insan yardımı olmadan öğrenebileceğinin göstergesidir. Ancak günümüzde kullanım alanı sınırlıdır ve hala yoğun araştırmalar devam etmektedir [22].

3.2.5.3. Destekleyici öğrenme

Destekleyici öğrenme, daha önce birçok açıdan tartıştığımız öğrenme metotlarından farklıdır. O, destekleyici öğrenmedeki bir danışman ile öğrenmenin karşısı olarak, tartışma yoluyla öğrenme olarak ifade edilmektedir. Tartışma, danışmandan ne yapacağımızı değil, geçmişte nasıl iyi şekilde yaptığımızı söylemek yönüyle ayrılmaktadır. Tartışma peşinen bilgilendirme yapmamaktadır. Tartışma ile gelen geri dönüş, seyrektir ve geldiği zaman geç gelmektedir. Bu kredi atama problemine yönlendirmektedir [23].

Bu yaklaşımda ağın her iterasyonunda ilgili değerin olumlu ya da olumsuz olup olmadığına bakılmaktadır. Bu olumluluk ya da olumsuzluk istenen değerin elde olup olmadığı ile ilgilidir. Ağ bu bilgilere göre ilgili yapıyı düzenlemektedir. Ağ devinim halinde bir girdi dizisi ile hem öğrenme ile hem de sonuçlar elde ederek işlenmektedir.

Örneğin bir satranç örneği verilebilmektedir bu kısım için. Sinir ağı yaptığı hamlenin iyi ya da kötü olmasını anlık ayırt edememesine rağmen hamleyi

gerçekleştirmektedir. Ancak oyun kazanılır ise hamle iyiler hanesine yazılmaktadır. Daha sonraki oyunlar için hamleler iyi olarak değerlendirilecek ve öğrenme işlemi gerçekleşmiş olacaktır.

3.2.6. Yapay sinir ağlarında öğrenme kuralları

Literatürde çok sayıda öğrenme algoritması vardır. Genel anlamda var olan algoritmalar matematiksel tabanlıdır ve kullanım amacı ağırlıkların güncelleştirilmesidir. İlgili algoritmalar şu kurallardan esinlenerek geliştirilmiştir:

- Hebb Kuralı
- Delta Kuralı
- Kohonen Kuralı
- Hopfield Kuralı

3.2.6.1. Hebb kuralı

1949'da Kanadalı psikolog Donald Hebb bu algoritmayı biyolojik temele dayalı şekilde geliştirmiştir. En eski ve en meşhur öğrenme algoritmasıdır. Diğer kuralların temeli niteliğindedir.

Eğer A nöronu, B nöronundan girdi alıyorsa ve ikisi de aktifse yani aynı işaretli ise, her iki hücrenin arasındaki bağlantı kuvvetlendirilmelidir.

3.2.6.2. Delta kuralı

Delta Kuralı, Widrow ve Hoff tarafından geliştirilmiş mühendislik temelli bir algoritmadır. Bu kural, nöronun istenilen ile gerçek çıkış değerleri arasındaki farkı azaltan, giriş bağlantılarını kuvvetlendiren ve bunları sürekli devinim halinde tutan bir kuraldır.

Bu kural, ortalama karesel hatayı, bağlantı ağırlık değerlerinin değiştirilerek hata oranını düşürmeyi amaçlamaktadır. Diğer adı en küçük kareler kuralıdır. Hata aynı anda bir katmandan öncekine doğru geriye yayılarak azaltılmaktadır. Bu hata düşürülmesi işlemi, çıkıştan girişe ulaşınca kadar sürmektedir.

3.2.6.3. Kohonen kuralı

Bu kural, biyolojik sistemlerdeki öğrenme yapısı esas alınarak geliştirilmiştir. Bu kuralda nöronlar öğrenme yarışı yapmaktadır. Kazananın ağırlıkları güncellenmektedir. Bu kural “kazanan hepsini alır” olarak da bilinmektedir. En büyük çıkışı olan işlemci nöron kazanmaktadır. Bu nöronun komşularını uyarma ve yasaklama gibi özellikleri vardır. Kural herhangi bir hedef çıkış gözetmemektedir. Bu sebeple danışmansız öğrenme sınıfına girmektedir.

3.2.6.4. Hopfield kuralı

Bu kural Hebb Kuralı’na benzer şekilde işlemektedir. Zayıflatma ve kuvvetlendirme büyüklüğü ile Hebb Kuralı’ndan ayrılmaktadır. Eğer istenilen çıkış ile girişin ikisi de aynı işaretli ise öğrenme oranı yoluyla bağlantının ağırlığı artırılmaktadır. Aksi durumda ise azaltılmaktadır.

3.3. Derin Öğrenme Nedir?

Yapay sinir ağları kapsamında ilk “Derin Öğrenme” ifadesi 2000 senesinde Igor Aizenberg ve arkadaşları tarafından ortaya atılmıştır. 2006’da ise Geoffrey Hinton bir makalesinde çok katmanlı ileri beslemeli yapay sinir ağının her kademedede ilgili katmanı etkili şekilde nasıl eğittiğini ve sonrasında geri yayılım ile nasıl ince ayar yapıldığını anlatmıştır.

Krizhevsky, Sutskever ve Hinton 2012’de GPU kullanılarak ön eğitim olmadan eğitilen mimariler tasarlamışlardır. Özellikle 2012-2017 yılları arasında çalışma

grupları ve başlangıç (startup) şirketlerin büyük firmalar tarafından satın alınması ile derin öğrenme alanındaki gelişmeler hız kazanmıştır.

Derin öğrenme, makine öğrenmesinin bir alt dalıdır. Özellik çıkarma ve dönüştürme işlemleri için doğrusal olmayan işlem birimi katmanları kullanılmaktadır. Herbir ardışık katman, kendinden önce gelen katmanın çıktısını girdi olarak almaktadır.

Derin öğrenme temelinde, verilerin özellik seviyeleri ya da temsil ettiklerinin öğrenilmesi vardır. Üst düzeydeki özellikler alt düzeydekilerden çıkarım yapılarak ve ilgili veri türetilerek yapının hiyerarşik bir gösterimi oluşturulmaktadır. Bu gösterim soyutlanan modelin farklı seviyelerinde karşılık bulan çok sayıda seviyeyi öğrenmektedir. Temel manada ilgili gösterimler yoluyla öğrenmeye dayanmaktadır.

Derin öğrenmede doğru bilgiyi doğru biçimde (formatta) elde etmek çözülmesi gereken en büyük zorluklardan biridir. Doğru verileri elde etmek, elde edilmeye çalışılan sonuçlarla ilişkili verileri toplamak veya tanımlamak anlamına gelmektedir. Derin öğrenme için doğru format genellikle bir tensör veya çok boyutlu bir dizidir. Bu nedenle, derin öğrenme için oluşturulan veri kümeleri - görüntüler, video, ses, metin ya da zaman serileri - doğrusal cebir işlemlerinin uygulanabileceği vektörlere ve tensörlere dönüştürmektedir [24].

Derin Öğrenme Ağlarının düzgün işlemesi, iyi bir eğitim kümesi ile mümkündür. Bu iş için öncelikle problem iyi anlaşılmış olmalıdır. Bir sinir ağının eğitiminde eğitim ve doğrulama için iki ayrı veri kümesi gereklidir. Eğitim verileri ağı eğitmeye, doğrulama verileri ise ağı ezberleme ve diğer olumsuz durumlarına karşı olan durumların ölçümü için kullanılmaktadır.

Bu eğitimde kullanılmak üzere açık kaynak kodlu ücretsiz veri kümeleri vardır. En popüler olanlar MNIST, CIFAR ve IMAGENET'tir.

Karakter tanıma uygulamasında kullandığımız MNIST veri kümesi, farklı kişiler tarafından rakamların el yazısıyla yazılmış halde bulunduğu ve farklı kenar belirleme

yöntemlerinin eğitiminde kullanılan çok geniş bir veri tabanıdır. Bu veri tabanı aynı zamanda makine öğrenme yöntemlerinin eğitimi ve testi için de literatürde geniş bir biçimde kullanılmaktadır. Veri tabanı, 60.000 eğitim görüntüsü ve 10.000 adet test görüntüsüne sahiptir. Bu eğitim görüntülerinin yarısı ve test görüntülerinin yarısı NIST veri setinden alınmıştır. Bazı araştırmacılar, yapay sinir ağlardan oluşan bir komite kullanarak MNIST veri seti üzerinde insan performansına yakın bir başarı elde etmişlerdir [25].

CIFAR, CIFAR-10 ve CIFAR-100 adları ile seksen milyon ufak resmin olduğu veri setidir. CIFAR-10 altmış bin 32x32 renkli resimden oluşmaktadır. On sınıfı vardır ve bu sınıflarda altı bin resim bulunmaktadır. Elli bini eğitim için, on bini test için kullanılmaktadır. CIFAR-100 ise yüz sınıfa ve bu sınıflarda altıyüz resme sahiptir. Sınıfın içerisinde, beş yüz adet eğitim resmi ve yüz adet test resmi vardır. Bu yüz sınıf, yirmi tane de süper sınıfa ayrılmaktadır. Bu kütüphaneler görüntü tanıma işlemlerinde kullanılmaktadır.

IMAGENET, WordNet düzenini temel alan bir resim kütüphanesidir. Her düğüm yüzlerce, hatta kimi düğümde binlerce resim tarafından tanımlanmaktadır. Son verilere göre, her düğüm beşyüz ve üzeri resim içermektedir.

3.3.1. Yapay zeka, makine öğrenmesi ve derin öğrenme ilişkisi

İnsan zekasını anlama merakı milattan önce 4. yüzyıla kadar dayanmakla birlikte; zekanın bir benzetiminin makineler üzerinde sağlanması fikri 18. yüzyıla kadar geri gitmektedir. 1796 yılında Wolfgang von Kempelen “Türk” ü icat etmiştir. Türk, mekanik bir satranç oyuncusu olan sözde otomatondur (pseudo automaton) ve sözde otomaton da olsa zekanın makineler üzerinde bir benzetimini sağlamaya yönelik ilk girişimdir [26].

Bu otomatona ithafen, Edgar Allen Poe bir makale yazmıştır. Eserin adı “Maelzel’in Satranç Oyuncusu” dur. Bu makale Türk adlı otomatonun hareketlerini açıklamak üzerinedir. Bu sistemin sözde olarak adlandırılmasının sebebi, masanın altına

gizlenen bir adamın gerekli satranç hamlelerini gerçekleştirmesidir. Bu sistemin gerçek satranç hamleleri yapan versiyonunu ilk defa, Torres y Quevedo geliştirmiştir. Sistem şah ve kale ile oyun sonunu otomatize eden bir yapıdadır. Tüm bunlara rağmen Yapay Zeka kavramı 1950'lere gelene kadar ismen ifade edilmemiştir.

Alan Turing'in bilgisayarların geleceği hakkında şöyle bir ifadesi olmuştur: Yaklaşık elli yıl içinde, yaklaşık 10^9 depolama kapasitesine sahip bilgisayarları taklit etme oyunlarını oynatmalarını sağlayacak şekilde programlamanın mümkün olacağına ve ortalama bir sorgulayıcının beş dakikalık sorgulamadan sonra doğru tanımlamanın yapılmasında, yüzde 70'den daha fazla şansı olmayacağına inanıyorum [27].

Bu sözden öncesinde, Alan Turing'in makineler düşünebilir mi? sorusu, kendisi tarafından asıl soru olarak ifade edilmiş ve bu sorgulama ilgili kavramın doğmasına sebep olmuştur ve 1956 senesinde McCarthy Dartmouth konferansında yapay zeka, yeni bir araştırma alanı olarak adlandırılmıştır.

Yapay zeka, makinelerin işlemleri insanlar gibi belirli kabiliyetler ölçüsünde gerçekleştirmesidir. Zayıf yapay zeka sistemleri sadece programlanı yapabilmekteyken; kuvvetli olanlar algoritmik hesaplar yoluyla, programlanı iyileştirip hatalardan öğrenen sistemlerdir.

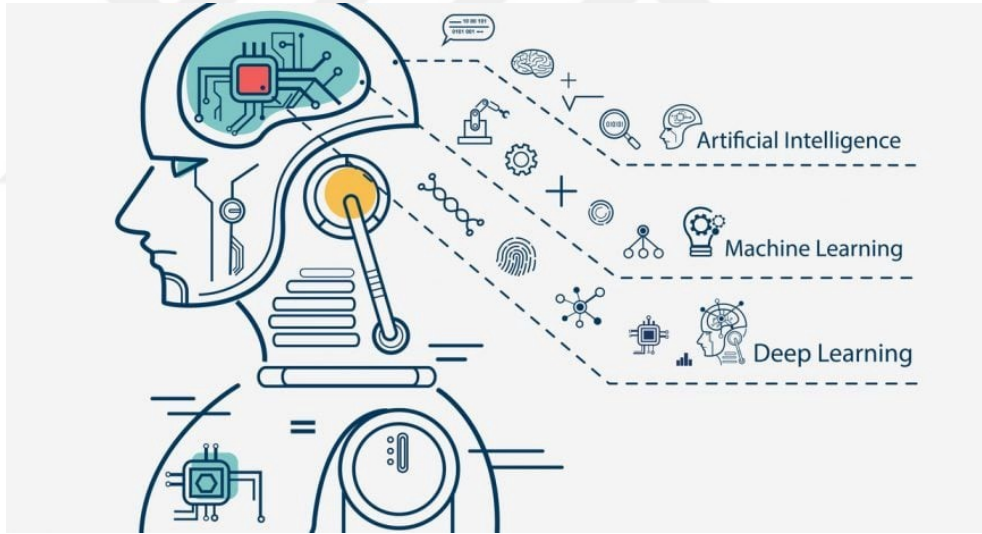
Makine öğreniminin ilk olarak isminin geçtiği zamanlar 1980'leri göstermektedir. Popülerliği veri madenciliği ile beraber artmaya başlamıştır. Sisteme sunulan veriler ve parametreler ile benzetimler gerçekleyerek, insandan daha iyi tespitler yapabilen, programlananları açığa çıkarabilen ve kendini eğitebilen bir yapıdadır.

Derin öğrenme kavramı ise ilk olarak 2010'da gündeme gelmiştir. Yüksek veri olanağı ile tek katman temelinde değil, birçok katman temelinde çalışabilen bir yapıdadır. Yani makine öğrenimi tek katman üzerinden ilerlerken, derin öğrenme birçok katmanda aynı anda işlemektedir. Bu çoklu katman çalışmasında makine öğrenmesindeki hesapları bir kerede gerçekleştiren, tanımlanması gereken

parametreleri bile kendisi belirleyen ve daha iyi parametreler ile değerlendirmeler yapan bir yapıya sahiptir.

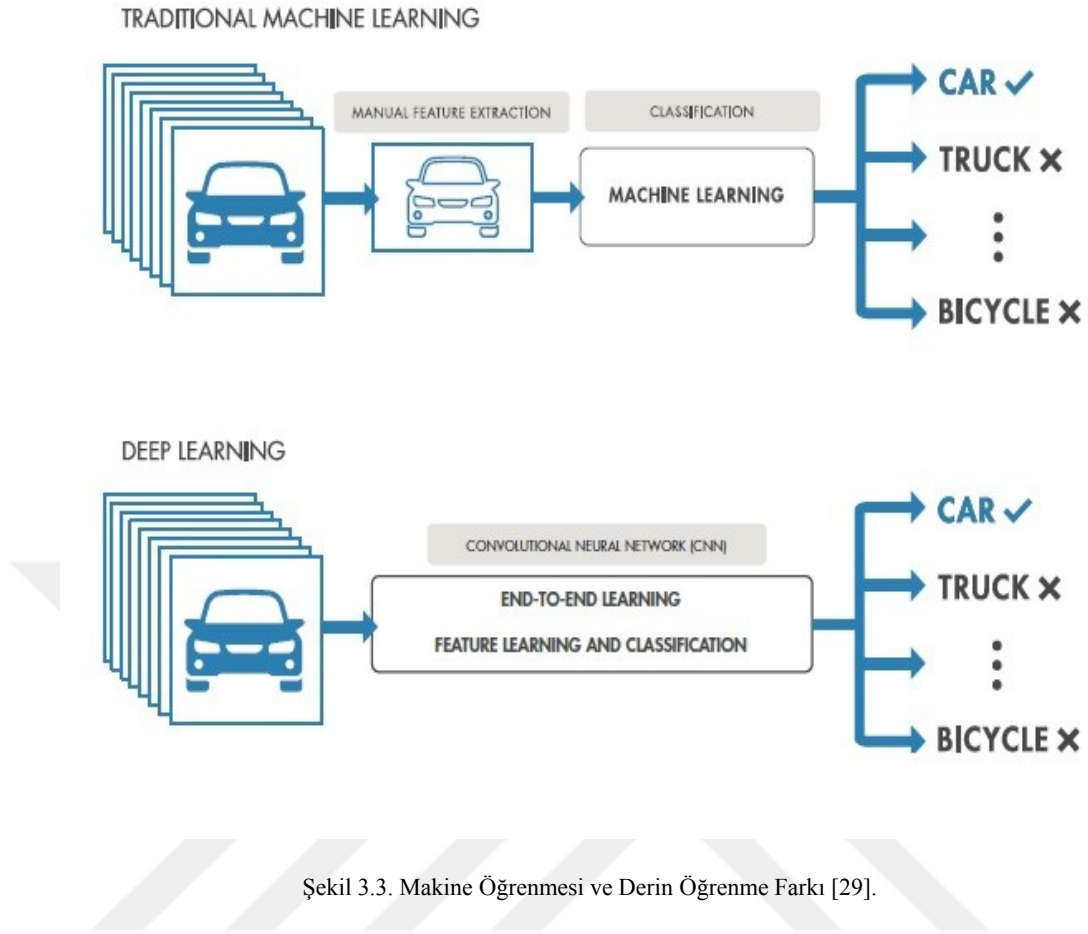
İşin aslına bakarsak makine öğrenimi; istatistik, veri madenciliği vb. alanlar ile yakın olduğundan birçok uzman, yapay zeka ile ayrı tutulması görüşünde birleşmişlerdir. Çünkü genel anlamda, makine öğrenmesinin yapay zekadaki özelliklere ihtiyacı yoktur. Ancak günümüzde ilgili kavramlar iç içe geçtiği için, günümüzde birlikte kullanılmaya başlamıştır. Örneğin; asistanlık uygulamaları ve botlar gibi.

Verinin çoğalması, yapay zekanın özelliklerini iyi bir şekilde açığa çıkartmasını sağlamaktadır. İşler karmaşıktıkça; yapay zekadan makine öğrenmesine, daha da karmaşıktıkça da; derin öğrenmeye geçişler başlayacaktır. Durumun anlatımı Şekil 3.2.'de gösterilmiştir.



Şekil 3.2. Yapay Zeka, Makine Öğrenmesi ve Derin Öğrenme Karşılaştırması [28].

Örneğin makine öğrenmesinde, bir araba ile kamyon resmini ayırmamız gerekirse; ilgili parametreler yardımı ile makineye tanıtım yapılması gerekmektedir. Kasası varsa kamyon, yoksa araba gibi. Ancak derin öğrenmede bunu kendisi öğrenmektedir. Yaptığımız tek şey, resimleri derin öğrenme sistemine göstermektir. Kuralları, farkları ve ayırıcı özellikleri sistem kendisi bulmaktadır. Şekil 3.3.'te bu durumun bir benzetimi görülmektedir.



Şekil 3.3. Makine Öğrenmesi ve Derin Öğrenme Farkı [29].

3.3.2. Derin öğrenme mimarileri

Literatürde altı mimari yapı vardır. Bu mimariler kullanıldıkları yerlere ve amaca göre farklılık göstermektedir. Sınıflandırma, tespit, tanı vd. sonuçların eldesi için veya ilgili modelin yapısına göre değişik modeller kullanılmaktadır. Verinin görüntü, ses, sinyal, metin gibi çeşitli türleri de yine ağ modelinin belirlenmesinde etkili olmaktadır.

3.3.2.1. Konvolüsyonel sinir ağları (CNN)

Konvolüsyonel sinir ağları çok katmanlı algılayıcıların bir çeşididir. Hayvanların görme merkezinden yola çıkarak geliştirilmiştir. Buradaki işlem, bir nöron hücresinin kendi bölgesindeki uyarılara verdiği cevaptır. Konvolüsyonel sinir ağları, bir ya da birden çok konvolüsyonel katman, altörnekleme katmanı ve sonrasında standart çok katmanlı sinir ağı gibi bir ya da daha fazla bağlı katmandan meydana gelmektedir.

İlk konvolüsyonel ağ 1988 yılında Yann LeChun tarafından ortaya atılmıştır. LeNet isimli mimaridir. 1998'lere kadar iyileştirmeler devam etmiştir. Bu ağ sisteminde, alt katmanlar konvolüsyon ve maksimum havuzlama katmanlarından meydana gelmektedir. Daha sonra gelen üst katmanlar ise tamamen bağlı geleneksel MLP'den oluşmaktadır.

Uygulama alanı olarak bu algoritmalar, görüntü ve ses işleme başlıca olmak üzere NLP (doğal dil işleme) ve biyomedikal gibi farklı alanlarda da kullanılmaktadır. Özellikle görüntü işleme alanında hata oranını %2'lere düşürebilmektedir. 2014 yılında, ImageNet Yarışması'nda, nesne sınıflandırması ve algılaması dalında üst seviyede yer alan takımlar bu algoritmanın modifiye edilmiş hallerini kullanmışlardır.

Ayrıca NLP (Doğal dil işleme) için de bu algoritmalar kullanılabilmektedir. NLP için kullanımlar şu şekildedir: anlamsal ayrıştırma, arama sorgusu elde etme, cümle modelleme, sınıflandırma, tahmin problemleri vb. Farklı alanlarda da kullanılmakta olan bir algoritmadır. Mesela kimyasal tepkimeler 3 boyutlu olarak temsil edilip sistem eğitilerek, Ebola ve Skleroz gibi hastalıklarda yeni biyomoleküllerin bulunması için kullanılmıştır. Bir diğer örnek şu şekildedir: Google DeepMind tarafından geliştirilen AlphaGo konvolüsyonel tabanlıdır ve ilk kez profesyonel bir oyuncuyu (insanı) yenmiştir.

3.3.2.2. Tekrarlayan sinir ağları (RNN)

Basit tekrarlayan ağın ilk tasarımı Jeff Elman tarafından yapılmıştır. Tasarlanan bu simülasyonda isim ve fiil kategorileri düzgün bir biçimde ayrılmıştır. Bununla beraber isimler canlı-cansız ve insan-hayvan, hayvanlar ise avcı-yırtıcı gibi alt başlıklara ayrılmıştır.

Tekrarlayan sinir ağı, ilgili bağlantıların yönlü döngü oluşturduğu bir modeldir. Bu döngü dinamik davranış sergilemektedir. Burdaki amaç sıralı bilgileri kullanmak üzerinedir. Geleneksel sinir ağında girdi ve çıktılar birbirinden bağımsız olduğu

düşünülmektedir. Fakat bu durum NLP için uygun görünmemektedir. Örneğin, bir cümle içinde sonraki kelimenin tahmini, hangi kelimenin o kelimeden önce geldiğini bilmekle olmaktadır. Bu yapının tekrarlanan olarak isimlendirilmesi, ilgili dizideki her öge için aynı görevi çıktılara göre gerçekleştirmesidir.

Alex Graves'in ele aldığı çalışma, ses verilerini doğrudan metne çeviren tekrarlayan sinir ağı tabanlı konuşma sistemidir. Bir diğer çalışma ise tekrarlayan sinir ağı ile konvolüsyonel sinir ağı birlikte, etiketlenmemiş görüntülere tanımlayıcı üreten bir modelin parçası olarak ele alınmıştır. Birleşik model, resim içindeki nesneleri tanımlamakla birlikte, bu tanımlayıcıların görüntü içindeki konumlarını bile saptamıştır.

Tekrarlayan sinir ağlarında yaklaşım önceki bilgileri kullanmaya yöneliktir. Örneğin “Kırmızı araba görüldü” cümlesinde, araba kelimesi kolay tahmin edilirken, ilgili bağlamlar arası boşluk artışıyla önceki bilgilerin kullanılması oldukça zor olmaktadır. Örneğin, “İtalya’da büyüdüm. Akıcı bir şekilde İtalyanca konuşurum” metnini ele alacak olursak; “İtalyanca” kelimesini tahmin ederken, cümleden dil adı için tahminde bulunulabilir ancak doğru kelimenin “İtalyanca” olduğunu tahmin etmek adına yer bilgisi de tutulmalıdır.

3.3.2.3. Uzun ve kısa vadeli hafıza ağları (LSTM)

Teorik olarak uzun vadeli bağımlılıklar uygulamada çeşitli problemlere yol açmaktadır. Bu durumu çözümlemek adına uzun vadeli bağımlılıkları öğrenen bir tekrarlayan sinir ağı türü olan Uzun ve Kısa Vadeli Hafıza ağları 1997’de bulunmuştur.

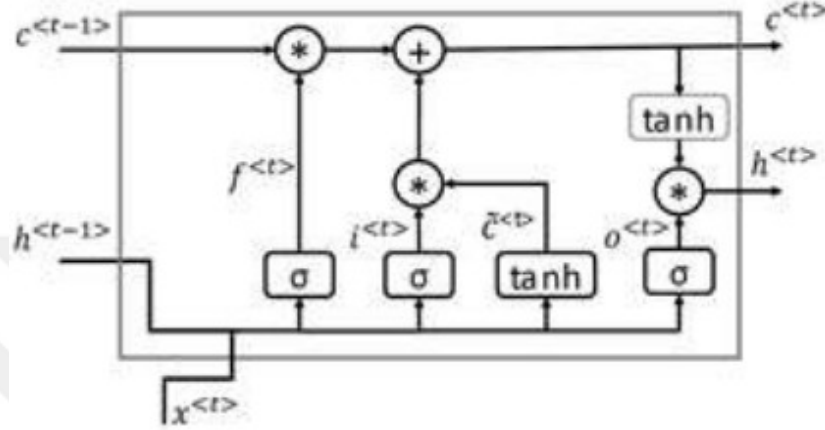
Uzun ve Kısa Vadeli Hafıza Ağları, uzun vadeli bağımlılık problemini çözmek için tasarlanan özel bir tekrarlayan sinir ağı türüdür [30]. Bu mimari (Bknz. Şekil 3.4.); giriş (Denklem 3.1), çıkış (Denklem 3.4), unutma kapıları (Denklem 3.2), hafıza hücreleri (Denklem 3.3) olmak üzere 4 temel kısımdan oluşmaktadır [31].

$$it = \sigma(w_{xixt} + w_{hi} h_{t-1} + w_{ci} c_{t-1} + b_i) \quad (3.1)$$

$$ft = \sigma(w_{xfxt} + w_{hf} h_{t-1} + w_{cf} c_{t-1} + b_f) \quad (3.2)$$

$$ct = ft \odot c_{t-1} + it \odot \tanh(w_{xcxt} + w_{hc} h_{t-1} + b_c) \quad (3.3)$$

$$ot = \sigma(w_{xoxt} + w_{ho} h_{t-1} + w_{co} c_{t-1} + b_o) \quad (3.4)$$



Şekil 3.4. LSTM Mimarisi [32].

Çalışma sistemi şu şekildedir: Kapılardan gelen veriler belirli bir aktivasyon fonksiyonundan (tanh veya sigmoid) geçmektedir. Sonra gelen veriler de girdi içeriği ile belirli işlemlerden (çarpma, toplama vs.) geçerek, çıktı olarak düğümden çıkmaktadır [33].

LSTM yapısı, RNN hücresinin hafıza ile birlikte işletildiği bir yapıdır. Hafıza aracılığı ile önceki zamanın verisi alınıp, bir sonraki zamana iletilmektedir. Bilginin değerlendirilip değerlendirilmeme durumuna ise eğitim aracılığı ile karar verilmektedir [34].

Uzun ve Kısa Vadeli Hafıza Mimarisi konuşma ve metin işleme konusunda oldukça başarılıdır. Çerçeve tabanlı ses sınıflandırma, konuşma verileri üzerinden anahtar kelime tespiti vb. uygulamalarda oldukça başarılı sonuçlar vermiştir. Ayrıca insan aktivitelerini öncül bir veri kullanmadan sınıflandırma gibi işlemleri öğrenen sistemlerde de yine konvolüsyonel sinir ağları ve uzun - kısa vadeli sinir ağları başarılı sonuçlar vermiştir.

Yine ilgili model ile farklı bir deneme müzik bestesi üretimidir. Tekrarlayan sinir ağında, nota tahmini yapılabilmektedir ancak dinlenmeye uygun bir beste yapılamamıştır. Uzun vadeli öğrenme yeteneği sayesinde bu modelde, blues türünde müzikler başarılı öğrenmeler yoluyla yeni besteler şeklinde oluşturulmuştur.

Diğer kullanım örnekleri de şu şekilde sıralanmaktadır: hizadan bağımsız protein homolojisinin algılanması, robotikte düğümleri bağlayan sistemler, düzensiz dillerde öğrenme yolları, çevrimdışı olarak el yazısının tanınması gibi.

3.3.2.4. Boltzmann makineleri

Sınırlı Boltzman Makineleri'nin ilk ismi 1986'da Harmonium olarak ortaya atılmıştır. Ancak daha sonraları 2006 senesinde Geoffrey Hinton ve arkadaşları hızlı öğrenme algoritması şeklinde ön plana çıkmasını sağlamıştır. Yapısı girdi setindeki olasılık dağılımını öğrenebilen üretken rassal şekildedir. Bu yapı Boltzmann Makinaları'nın bir alt türüdür. Görünür ve gizli olmak üzere aralarında simetrik bağlantı olan 2 parçadan oluşan graf yapısıdır. Bir graf içinde düğümler arası bağlantı bulunmamasına rağmen kısıtlamasız olan şekilde gizli birimler arasında da bağlantı mevcuttur. İlgili kısıtlama Boltzman Makineleri'nin normal yapısına göre daha etkili eğitim algoritmalarına olanak tanımaktadır.

Sınırlı Boltzmann Makineleri; boyut indirgeme, sınıflandırma işlemleri, özellik öğrenimi vb. konular için kullanışlıdır.

3.3.2.5. Derin inanç ağları

Geoffrey Hinton, Derin İnanç Ağları'nı Kısıtlı Boltzmann Makinelerinin bir yığını olarak ifade etmektedir. Kısıtlı Boltzmann Makineleri'nde her katman önceki ve sonraki katmanlar ile birbirine bağlıdır. Ancak bu katmanların düğümlerinin birbiri ile yatay iletişimi bulunmamaktadır.

Derin İnanç Ağları genel manada görüntü tanıma ve üretme gibi alanlarda kullanılmıştır. Kelime sayısı vektörleri elde etme, insan hareketi tanımlama gibi kullanım şekilleri olmuştur.

3.3.2.6. Derin oto-kodlayıcılar

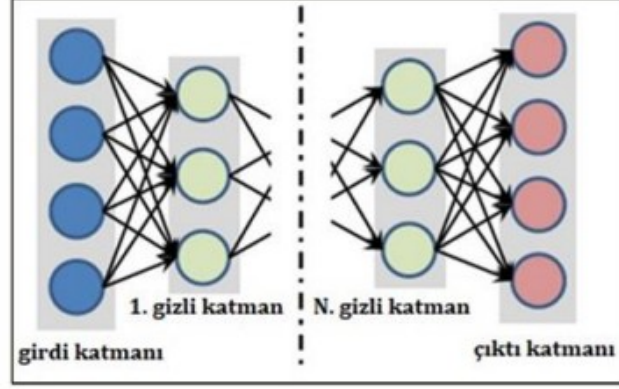
Diğer bir adı Diabolo Ağı olan Oto-kodlayıcılar denetimsiz öğrenme amacıyla kullanılan özel bir yapay sinir ağıdır. Temel manada 3 adet katmandan meydana gelmektedir. Bunlar; girdi katmanı, gizli katman ve çıktı katmanıdır. Otomatik kodlayıcıyı oluşturmada önce ayarlamamız gereken 4 parametre vardır. Bunlar kod büyüklüğü, katman sayısı, katmanlardaki düğüm sayısı ve kayıp fonksiyonudur [35].

Buradaki amaç bir veri seti için boyut indirgeme ve özellik çıkarımı hedefiyle bir kodlama yapmaktır. Kabaca sıkıştırılmış verideki en iyi özelliklerin öğrenilmesi beklenmektedir. İleri beslemeli bir sinir ağıdır. Bu modelde, ilgili işlemlerden sonra çıktı olarak yine aynı girdi denk gelene kadar ağırlıklar sürekli değiştirilmektedir. Hedefe ulaşıncaya, gizli katmandaki düğüm sayısı bize girdi verisini temsilen vermektedir.

Derin veya yığılı oto-kodlayıcılar ise, her bir katmanın çıktıları ile ardışık katman girişlerinin birbirine bağlandığı çok katmanlı bir sinir ağıdır. Hinton ve Salakhutdinov tarafından, 2006 yılında geliştirilmiştir [36]. Uzun yıllar boyunca, sinir ağlarının temelinde yer alan bir kısım olmuştur [37]. Derin öğrenme mimarilerinin doğuşu ile derin öğrenme mimarilerinde yer tutmaya başlamıştır [38].

Derin oto-kodlayıcılar, birden fazla gizli katmanla ilişkilidir. Bir Derin Oto-kodlayıcı yapısı Şekil 3.5.'te verilmiştir. Genel anlamda, tek katmanlı bir oto-kodlayıcı, özgün verileri temsil eden bir özelliğe sahip olmadığından, derin(yığılı) oto-kodlayıcı kavramı ortaya çıkmıştır. Eklenen katmanlarla, oto-kodlayıcı daha karmaşık kodlamalar öğrenebilmektedir. Ancak bu uzmanlaşmanın çok olmaması da önemlidir. Çok uzmanlaşma sonucunda, eğitim verileri düzgün şekilde yeniden yapılandırılabilir ancak ilgili süreçle ilgili herhangi bir faydalı genel veri

oluşturamayacak ve yeni durumlar için genelleme yapma olasılığı da düşük olacaktır [39].



Şekil 3.5. Birden Fazla Gizli Katman İçeren Bir Derin Oto-kodlayıcı Yapısı [40].

Derin oto-kodlayıcılarda; kod çözücü katmanlar, sıkıştırılmış öznitelikten girdi benzeri çıktılar üretmeye çabalamaktadır. Çıktı girdi ile aynı olmayabilmektedir. Bu ikisi arasındaki benzerlik ne kadar fazla ise, algoritmadaki fonksiyon o derecede başarılıdır denilmektedir [41].

Bu aynı veriyi elde etme çabası bazı sistemlerde, ilgili verinin ezberlenmesine ve test verisi için kötü sonuç değerleri elde edilmesine yol açmaktadır. Bu durumu önlemek adına gürültü giderici (denoising) geliştirilmiştir. Girdi katmanındaki veriye gürültüler eklenmekte, çıktı katmanında gürültüsüz halde sonuçlar elde edilmektedir. Böylece sistemin eğitim verisinde bulunmayan, farklı örüntülerde verileri de öğrenmesi sağlanmaktadır [42].

Derin oto-kodlayılar kumaş hata tespiti için denenerek, iki adet oto-kodlayıcı ile oluşturulan derin ağ ile oldukça ayırt edici sonuçlara ulaşılmıştır. Mail spam yakalama için yapılan bir diğer örnekte ise Bayes, Karar Ağaçları vb yöntemler ile kıyaslanarak daha iyi sonuçlar verdiği görülmüştür.

Derin oto-kodlayıcılar özellikle anormallik ve aykırı değerleri bulmada iyi sonuçlar veren bir mimaridir.

3.3.3. Derin öğrenme uygulamaları

Bu kısımda konu bazında bir sınıflandırma yapılarak, popüler olan yöntemler incelenecektir. Metin Özetleme başlığı, tezin temel konusu olduğu için bu kısımda ele alınmamış olup, ayrıntılı şekilde başlık 2.1 ve alt başlıklarında işlenmiştir.

3.3.3.1. NLP (Doğal dil işleme)

Doğal dil işleme, asıl işlevi bir doğal dili çözümleme, anlama, yorumlama ve üretme olan bilgisayar sistemlerinin tasarımı ve gerçekleştirilmesini içermektedir [43].

NLP'nin çalışma alanı insanların doğal dillerinin bilgisayar tarafından nasıl işlendiği konusudur. Buradaki amaç doğal dillerin kurallı yapısının çözümlenip bir anlam çıkarılarak yeniden bir yapı oluşturulmasıdır.

Doğal Dil İşleme'nin bir alt başlığında, Belge Erişimi vardır. Bunun sebebi Belge Erişimi'nin, Doğal Dil İşleme'nin özel uygulama alanları olan konuşma, görüntü, video erişimi, yazılı metin vb ile uğraşmasıdır. Belge erişimi konusunun kapsamı, belge ambarlarından algoritma ve model çıkarımıdır.

Ancak Belge Erişimi ile Doğal Dil İşleme arasında sınırlı bir etkileşim vardır. Bunun sebebi Belge Erişimi'nin beklentilerinin Doğal Dil İşleme alanında karşılık bulmaması ve yine Belge Erişimi'nde çok kullanılan istatistiksel yöntemlerin Doğal Dil İşleme'de kullanılmasının tercih edilmemesidir.

Bahsi geçen etkileşim Doğal Dil İşleme'de nicel yöntemlerin yeniden güçlenmesi ile artmıştır. İlgili etkileşim örnekleri şu şekildedir: belgelerdeki terim dağılımının olasılıklı modelleri, hitap bölümleri, vektör uzayı modeli ve gizli anlambilimsel dizinleme.

3.3.3.2. Görüntü işleme

Derin öğrenmenin görüntü işleme alanındaki popüler olan kullanımı görüntü sınıflandırma üzerinedir. Bu konuda ortak değerlendirme seti olarak MNIST kullanılmaktadır. Bu set eğitim amacıyla 60000 ve test amacıyla 10000 el yazısı rakamından meydana gelmektedir. Bugüne kadar yapay sinir ağlarından, istatistiksel yöntemlere kadar birçok yöntem bu veri seti üzerinde test edilmiştir.

2011'de Konvolüsyonel Sinir Ağları görsel desen tanımda ilk defa süper insan performansı vermiştir. Ekim 2012'de, Alex Krizhevsky, George Hinton ve ekibi tarafından oluşturulan bir sistem büyük ölçekli bir ImageNet yarışmasını, sıg makine öğrenme yöntemleriyle belirgin bir farkla kazanmıştır. Yine 2014'te derin öğrenmeyi kullanan ImageNet yarışmasındaki hata oranı, benzer derin öğrenme modelleri ile daha da aşağı seviyelere çekilmiştir. Bunlarla birlikte derin öğrenme tabanlı denetimsiz öğrenme algoritmalarından da etiketli olmayan veriler üzerinden sınıfa özgü nitelikler çıkarılmaktadır. Derin öğrenme nesne tanıma problemlerinde de oldukça başarılı sonuçlar vermektedir.

3.3.3.3. Diğer uygulamalar

Bahsi geçen uygulama alanları dışında, yine farklı alanlarda derin öğrenme yöntemleri uygulanmaktadır. Bunlar: uyku kalite tahmini, insan aktivite tanıma, insan hareket tespiti gibi konulardır.

Ayrıca CRM (Müşteri İlişkileri Yönetimi) otomasyonu konusunda da derin takviye öğrenme yöntemi denemeleri olmuştur. RFM (güncellik-frekans-tutar) değişkenleri olarak tanımlanan müşteri durumu kısımlarındaki, muhtemel doğrudan pazarlama eylemlerinin değerinin yaklaşık olarak hesabı sinir ağları ile yapılmıştır.

Kapalı alan yönlendirmesi de önemli bir problemdir. Örneğin; bir quadcopterin bağımsız olarak iç mekanda dolaşıp tek kamera ile bir hedefi bulması için bir sistem düşünülmüştür. Konvolüsyonel Sinir Ağları ile uzman bir pilotun hareketleri

benzetim yapılmış, performans ölçütleri çeşitli kapalı mekanlardaki gerçek zamanlı deneyler kullanılarak sonuç olarak sunulmuştur.

Bir diğer örnek kullanım Google Lens'tir. Google Lens, yapay zekayı ve derin öğrenme modelini kullanarak fotoğraflarda bulunan objeleri belirleyen ve tespit ettiklerine bağlı olarak önerilerde bulunan bir uygulamadır [44].

Google Lens ile şu tip işlemler yapılabilir: akıllı metin seçimi, akıllı metin araması, giyim ve dekor araması vb.'dir. Akıllı metin seçiminde, aynı kart okuyucular gibi ilgili metni kopyalayıp alma imkanı vardır. Akıllı metin araması, ilgili metni yakalayarak arama yapma üzerinedir. Giyim ve dekor araması, hoşta giden bir kıyafeti kameraya taratarak yorum ve alışveriş seçeneklerine bakmaya ve ev dekorunu tanımlamaya yaramaktadır.

3.3.4. Derin öğrenme kütüphaneleri

3.3.4.1. Keras

Tensorflow'u arka planda araç olarak kullanan yüksek seviye bir yapay sinir ağı API'sidir. Keras ile sıralı katmanlardan sinir ağı oluşturulurken ilgili model için sarmalayıcı sınıf (wrapper) kullanılmaktadır. Modeli eğitme ve çalıştırma amacıyla kullanılan compile(), fit(), evaluate() gibi metotlar Keras arayüzüne uygulanmaktadır.

```
from keras.models import Sequential  
model.Sequential()
```

Keras katman sınıfı, tam bağlı katmanlar, maksimum havuz katmanları ve aktivasyon katmanları vb için ortak bir arayüz imkanı sunmaktadır. Modele yeni bir katman eklemek add() fonksiyonu ile yapılmaktadır. Bir adet gizli katmanı olan basit bir model örneği verecek olursak:

```
Import numpy as np
```

```

from keras.models import Sequential
from keras.layers.core import Dense, Activation
X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]],
dtype=np.float32)
y = np.array([[0], [1], [1], [0]], dtype=np.float32)
model=Sequential()
model.add(Dense(32,input_dim=X.shape[1]))
model.add(Activation('softmax'))
model.add(Dense(1))
model.add(Activation('sigmoid'))

```

İlk gizli katman, 32 düğümden meydana gelmektedir. Bu düğümler iki elemanlı vektörü girdi olarak almaktadır. Her bir katman, bir önceki katmanın çıktısını girdi kabul etmektedir. Bu zincir son katman olan çıktıya kadar sürmektedir. Model oluşturulduktan sonra çalışma öncesi derlenmelidir. Keras yapısındaki model, derleme aşamasında TensorFlow kütüphanesini kullanmaktadır. Modeli çalıştırmadan önce istenen girdi verisi üzerinde optimizier, hata fonksiyonu ve diğer parametreler tanımlanması gerekmektedir. Derleme için şu şekil bir örnek verilebilmektedir:

```

model.compile(loss="categorical_crossentropy",
optimizer="adam", metrics = ["accuracy"])

```

İlgili parametreler şu şekilde seçilmiştir:

Hata fonksiyonu: Çapraz entropi

Algoritma: Adam algoritması

Metrik: Doğruluk

İlgili sonuçlar model.summary() komutu ile görüntülenebilmektedir. Epoch sayısını içeren fit() metodu ile model eğitilmekte ve değerlendirme için ise evaluate() metodu kullanılmaktadır.

Aşağıdaki maddeleri içeren ihtiyaçlarınız olması durumunda Keras kullanımı tavsiye edilmektedir: [45]

- Kolay ve hızlı prototip oluşturma (kullanıcı dostu olma, modülerlik ve genişletilebilirlik.),
- Konvolüsyonel (evrişimli), tekrarlayan ya da ikisinin kombinasyonu olan ağ gereksinimi,
- CPU ve GPU üzerinde sorunsuz çalışma arayışı.

3.3.4.2. Tensorflow

2015 senesinde Google tarafından çıkartılan açık kaynaklı bir kütüphanedir. Destek verdiği diller Java, C++, Go ve Python'dur. Veri akış grafikleri aracılığı ile sayısal hesaplama için kullanılmaktadır.

TensorFlow başlangıçta, makine öğrenimi ve derin sinir ağları araştırması amacı ile geliştirilmiştir. Başlarda amaç Google içindeki bir kullanım iken, sistem daha sonraları diğer alanlarda geniş bir şekilde kullanılmaya başlanmıştır.

Esnek mimarisi, hesaplamayı sadece tek API kullanılarak ilgili cihazlardaki bir veya daha fazla CPU veya GPU gibi birimlere dağıtmayı sağlamaktadır. Tüm işlemler düğüm olarak tanımlanmakta ve bu düğümler arasındaki kenarlara da tensor ismi verilmektedir. TensorFlow'da veri integer, float, string vb şekilde tanımlanamamaktadır. Bu değerler tensor adlı nesnede tutulmaktadır. Tensorler, çok boyutlu veriler tutan dizi yapılarıdır. Yaygın olarak kullanılanları şunlardır: skalar(0B), vektör(1B), matris(2B), 3B, 4B, 5B tensorler.

#Tensorflow kütüphanesi şu şekilde yüklenmektedir

```
>>import tensorflow as tf
```

#a, 0-boyutlu int32 türünde tensor

```
>> a = tf.constant(31)
```

#b, 0-boyutlu string türünde tensor

```
>> b = tf.constant ("tensorflow")
```

#c, 1-boyutlu int32 türünde tensor

```
>> c = tf.constant ([123, 456, 789])
```

#d, 2-boyutlu int32 türünde tensor

```
>> d = tf.constant ([[123, 456, 789],
                      [444, 555, 666]])
```

#e, 3-boyutlu int32 türünde tensor

```
>> e = tf.constant ([[[123, 456, 789],
                      [444, 555, 666]],
                      [[123, 456, 789],
                      [444, 555, 666]]])
```

TensorFlow ağ içindeki gradyanları otomatik hesap ve hata fonksiyonunu optimize etmektedir. İlgili hesaplar bir ya da birden çok GPU ile yapılarak büyük ölçüde hızlandırılabilir.

TensorFlow; Windows, Linux, Mac vb bilgisayar ve Android işletim sistemli cihazlarda çalışabilmektedir. Taşınabilir yapıdadır. TensorFlow'un topluluğu gelişmiş durumdadır. Derin sinir ağı modelleri için geliştirme, analiz etme, iyileştirme gibi işlemler için TensorBoard adlı aracı vardır. Kullanışlılık derecesi yüksek bir platformdur.

3.3.4.3. Digits

NVIDIA'nın Derin Öğrenme GPU Eğitim Sistemi (DIGITS), görüntü sınıflandırma, segmentasyon ve nesne algılama gibi hassas işler için derin sinir ağını hızlı eğitmek vb. amaçlarla kullanılmaktadır.

Digits tamamen etkileşimli olduğundan veri bilimcileri programlama ve hata ayıklama yapmak yerine ağ tasarımı ve eğitimi gibi konulara öncelik tanıma imkanına sahip olmaktadır. Yangqing Jia tarafından Caffe isimli derin öğrenme kütüphanesi (programlama yapmadan çalışan bir kütüphane) web arayüz desteğine kavuşturularak geliştirilmiş ve Digits oluşturulmuştur.

Digits şu özellikleri ile öne çıkmaktadır:

- Çoklu GPU kullanılarak sinir ağları tasarımı ve eğitimi.
- Gelişmiş görsel ekran çıktıları ile performansı gerçek zamanlı olarak takip etme.
- Tamamen etkileşimli yapı.

3.4. Yapay Sinir Ağları ve Derin Öğrenme İlişkisi

Aslında bu iki kavramın, genel anlamda hiçbir farkları yoktur. Ancak, derin öğrenme algoritmaları, yapay sinir ağlarının karmaşık halidir. Derin öğrenmede yeni olarak şunlar sıralanabilmektedir;

- Daha çok veri ve daha çok işlem gücü
- Yeni doğrusal dışı (İng. non-linear) aktivasyon fonksiyonları
- Yeni ilkleme (İng. initialization) yöntemleri
- Yeni düzenlileştirme (İng. regularization) yöntemleri [46]

3.5. PageRank Algoritması ve TextRank Algoritmasının Gelişimi

TextRank Algoritması, çok popüler ve etkili bir metin özetleme algoritmasıdır. TextRank Algoritması'nı incelemeden önce PageRank algoritmasını anlamamız gerekmektedir. Çünkü TextRank, PageRank'tan türetilmiştir.

PageRank Algoritması, Google tarafından yazılmış, arama motorunda kullanılan, popüler bir algoritma olduğu için, genel manada duyulmuş olma oranı yüksektir. Web sayfalarının değerlerini hesaplama üzerine kurulmuş bir yapıdadır.

“PageRank” sözcüğü aynı zamanda bir kelime oyunudur. Bu, web sayfalarını önem sırasına sokmada kullanılan bir algoritmadır fakat aynı zamanda başlıca mucidi olan Larry Page’in yarattığı ve kendi ismini taşıyan bir sıralama algoritmasıdır. Page ve Brin buldukları algoritmayı 1998 yılında bir akademik konferans bildirisi ile yayınladılar: “Büyük Boyutlu bir Hipertekst Web Arama Motorunun Anatomisi.” Başlığından da anlaşılacağı gibi bu bildiri PageRank (Sayfa Sıralaması) sistemini açıklamanın da ötesine gitmektedir [47].

Yani bu algoritmanın Google versiyonu, orijinalinden oldukça uzaklaşmıştır. Bunun ardında yatan sebep, Google’ın algoritmayı çok büyük ölçekte kullanmasıdır. Temel manada ele alırsak, özellikler şu şekilde sıralanmaktadır:

- Önemli sayfalar, önemli sayfalarla bağlıdır.
- Bir sayfanın PageRank değeri, esasen o sayfanın bir kullanıcı tarafından ziyaret edilme olasılığıdır.

Basitçe, herhangi bir u internet sayfasının sırasını bulmak için aşağıdaki eşitlik (Denklem 3.5) kullanılmaktadır.

$$PR(u) = (1 - d) + d \cdot \sum_{v_i \in B_u} \frac{PR(v_i)}{L(v_i)} \quad (3.5)$$

Denklem (3.5)’de belirtilen B_u , u ’nun bağlantılarının bulunduğu sayfaları, d ise sönümleyici faktörü belirtmektedir. İlk olarak her bir sayfanın sırası, diğer sayfalarda bulunan bağlantılarının sayısı olarak belirlenmektedir. Daha sonra (3.5) kullanılarak yinelemeli bir şekilde sayfaların sırası hesaplanmaktadır [48].

Bu algoritmanın metin üzerinde uygulanması ile ortaya TextRank algoritması çıkmıştır. Temelinde, cümleler arası bağlantılar kontrol edilerek önem sırasına göre sıralamak vardır. Herhangi bir puanlama işlemi yoktur. Daha sonra ise istediğimiz en önemli cümleler özet olarak kabul edilmektedir.

3.6. Diğer Kavramlar

3.6.1. Özellik

Bir gözlem ya da ölçüm ele alındığında, bunların bir sıralaması ya da ilgili ayarı özellik olarak tanımlanmaktadır. Bu tez kapsamında özellik kavramı, metinden çıkarılan verinin değişik tipleridir. Yine bu özellikler genel anlamda bir yapay sinir ağında beslenmek üzere bir verinin vektörüne birleştirilerek kullanılmaktadır.

3.6.2. Belirteçlere ayırma (tokenization)

Belirli bir veri kitlesinin, marka (jeton, token) isimli parçalara ayrılması işlemine denilmektedir. Yine tez kapsamındaki metin özetleme işlemi ile ilgili veri paragraf, ilgili paragraf cümle, ilgili cümle ise kelime şeklinde ayrılmaktadır.

3.6.3. Anahtar kelime çıkarma

Bu işlemde dökümandaki en çok geçen kelime, başlıkta geçen kelimeler vb. ilişkiler hesaplanarak en ilgili bilgiler bulunacak ve bunlardan anahtar kelime ve kelime öbekleri ayrılacaktır. Bulunan anahtar yapılar özellik çıkarma işlemi için kullanılacaktır.

3.7. Kullanılan Sistem, Program, Kütüphane ve Teknolojiler

Bu tez kapsamında yapılan deneysel çalışmalar Tablo 3.1.'de verilen özelliklere sahip bilgisayar üzerinde gerçekleştirilmiştir.

Anakart	MSI Gaming M5
İşlemci	Intel Core i7-6700K CPU @4.00 Ghz
Ram	Gskill 16gb 3000 Mhz
Disk	Samsung SSD 850 EVO
İşletim Sistemi	Windows 8.1 Embedded
IDE	PYCharm 2019.3.3 Community Edition

Tablo 3.1. Kullanılan Sistem ve Program.

Yapılan incelemeler sonucunda Keras kütüphanesinin kullanımına karar verilmiştir. Bu kütüphanenin seçilmesinin sebepleri maddeler halinde şu şekildedir:

- Arkaplanda TensorFlow, Theano, CNTK vb kütüphanelerin kullanımına izin vermesi.
- Modülerlik, minimalizm, genişletilebilirlik, Python-doğallık gibi ilkelere sahip olması.
- Hızlı denemeler yapmak için ideal olması.

Tez çalışmamız, Keras'ın son sürümünde olduğu üzere birinci dil olarak Python kullanmaktadır ve tüm NLP kütüphaneleri bu ölçüde hazırlanmıştır. Python, yorumlayıcı bir programlama dilidir. Kullandığım versiyon 3.7.0'dır.

BÖLÜM 4. ÖNERİLEN MODEL

Modelimizde, haber makaleleri veri seti olarak verilerek, bunların tek cümleden oluşan başlıklar olarak özetlenmesi hedeflenmiştir. Kütüphane olarak Keras, IDE olarak PyCharm kullanılmıştır. Genel manada yöntemler dilden bağımsızdır. Modelde; derin öğrenme mimarisi olarak, derin oto-kodlayıcı yapısı kullanılmaktadır. Ara (gizli) katman diye tabir ettiğimiz katmanlarda ise LSTM yapısına yer verilmiştir. Böylece bu bütünleşik yapı ile daha verimli sonuçlara ulaşılması amaçlanmıştır. İlgili model açık kaynak bir projenin üzerine geliştirilmiştir. Modelin özeti Şekil 4.1.'de gösterilmiştir:

```
Model: "model_1"
```

Layer (type)	Output Shape	Param #	Connected to
encoder_inputs (InputLayer)	(None, None)	0	
encoder_embedding (Embedding)	(None, 500, 100)	500200	encoder_inputs[0][0]
decoder_inputs (InputLayer)	(None, None, 2001)	0	
encoder_lstm (LSTM)	[(None, 100), (None, 80400)		encoder_embedding[0][0]
decoder_lstm (LSTM)	[(None, None, 100), 840800		decoder_inputs[0][0] encoder_lstm[0][1] encoder_lstm[0][2]
decoder_dense (Dense)	(None, None, 2001)	202101	decoder_lstm[0][0]

```
Total params: 1,623,501  
Trainable params: 1,623,501  
Non-trainable params: 0
```

Şekil 4.1. Modelin Özeti.

İzleyen başlıklardaki anlatım sırası, şu şekildedir:

- Öncelikle model için ilgili çalışma mekanizmaları,
- Az da olsa encoding-decoding mekanizması,

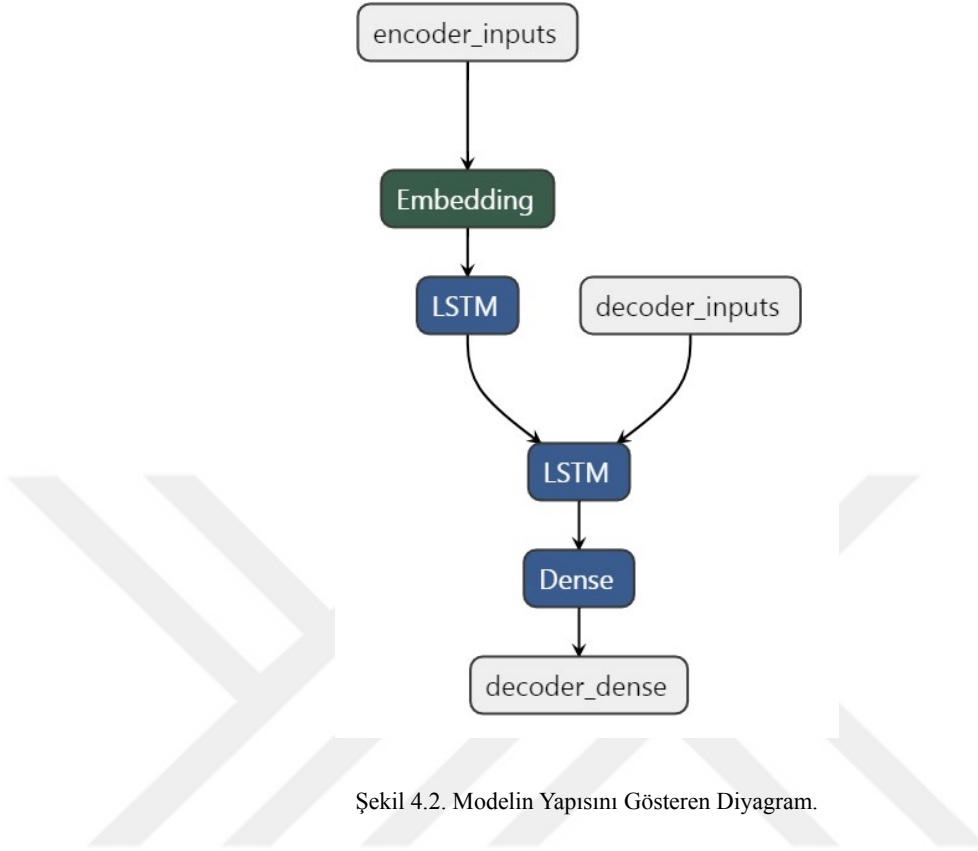
- Encode diye tabir ettiğimiz girişlere sahip asıl kod,
 - Encoding modelin çalıştırılıp; çıkışların kod çözme modeli üzerinden çalıştırılarak, kodların tamamen çözülmesi,
 - Asıl tahmin işleminin arka planındaki aşamalar,
 - Modelin eğitimi ve testi için ilgili kod anlatımı,
 - Modelin çalıştırılması ve örnek yapıları,
 - Modelin mimarisi,
- Şeklinde ilerlenecektir.

4.1. Modelin Oluşturulması

Encoding-decoding mekanizması iki alt modele ayrılmaktadır. Adından da anlaşılacağı üzere; bir encoding (kodlayıcı) ve bir decoding (kod çözücü). Yani encoder, kelime girdisi alıp ve onu bir grup sayıya dönüştürmektedir. Decoder ise tam tersi şekilde çıktıyı almakta ve sonra onu tekrar kelime girdisine çevirmektedir. Bu sistem bize hangi kelimelerin önemseneceğini, hangilerinin göz ardı edileceğini göstermektedir.

Sistemde, katıştırma katmanı adlı bir katman işlemektedir. Bu katman, kelimeleri bir grup sayıya kodlamaya yaramaktadır. Gömme mekanizmasından sonra kodlanmış bir dizi hücreyi temsil eden bir hücre vardır. Bu hücre kodlamaya gömülmek üzere beslenmektedir. Sonrasında bu hücreler aracılığı ile üç adet çıkış elde edilmektedir. Modeli nasıl çözeceğimize dair gerçek çıktılar elde edilmiştir. Yani bir diziye aktarma işlemi yapılmaktadır.

Temel model, üç alt modelden oluşmaktadır. İkinci aşamada kodlayıcı alt modeli işlemektedir. Son olarak da kod çözücü alt model gelmektedir. Modelin yapısını gösteren diyagram Şekil 4.2.'de gösterilmiştir.



Şekil 4.2. Modelin Yapısını Gösteren Diyagram.

4.2. Modellerin Eğitilmesi ve Özetleme Mekanizması

Önceki bölümde, modelin nasıl inşaa edildiği incelenmiştir. Kodun büyük bir kısmı başlatıcıda bulunmaktadır. Burada uyum fonksiyonu adında bir fonksiyon mevcuttur. Bu kısımda ilgili model eğitilmektedir. Önceki kısımdaki birkaç kod satırı, bir grup veriyi tahmin işlemi yapmaktadır. Örneğin; kontrol noktası dosyaları ve ağırlık dosyaları vb alınmakta ve yapımız kaydedilmektedir. Bundan sonra eğitim ve test verileri vardır. Asıl eğitim model uyum jeneratörü çağırdığımız satırda gerçekleşmektedir. Buradaki durum oldukça farklı işlemektedir. Aslında eğitimi kendimiz yapsak da, modelin verimliliğine paralel çalışmaktadır.

Eğitimden sonra, istediğimiz parti büyüklüğü ve epoch'lar için çalıştırmak istediğimiz adımların sayısı belirtilmektedir. Burdaki epoch'lar yani dönem sayıları sadece günlük kaydı içindir. Bunlar asıl verilerimiz değildir. İlgili fonksiyon sonunda tüm ağırlıklar, ağırlık dosya yoluna kaydedilmektedir. Aslında herşey standarttır.

Sadece ağırlık konfigürasyonu gibi ihtiyacımız olan tüm parçalar tahmin edilmektedir. Burada güzel olan kısım, ağırlık kaydedildikten sonra, her seferinde onların yeniden eğitilmesi yerine, tahmini veya diziden diziye tahmin dizisini tekrar tekrar çağırabilmesidir.

Özetleme fonksiyonunda ise, tahmin değeri ve bazı girdiler ile tahmin yapılmaktadır. Hedef metin üzerinde çıktı alınmaktadır. Sona ulaştığımız zaman ya da hedef metin toplam diziden büyük olduğunda olay tamamlanmaktadır.

4.3. Kodlama Yapısı

Esasen ilgilendiğimiz konu başlık ve metindir. Başlık etikettir. Metin modelimizin girdisidir. Ağırlıklar, ağırlık dosyasından yükleneceği zaman, eğer biz oraya giderken mevcutlarsa, şu an eğitimdedir. Ama daha önce eğitilmişse, o zaman mevcut ağırlığa olabildiğince yaklaşabilmemiz için bu ağırlıklar yüklenmek istenmektedir. Sonra verilerimiz eğitim ve teste ayrılmaktadır.

En son özetleyici fonksiyon çağırılmaktadır. Özetleyici fonksiyon, model kullanılarak çalıştırılmakta, çıktı alınmakta, ilgili çıktı biçimlendirilmekte ve böylece metnin özünü veren, tek cümlelik manşet çıktısı elde edilmektedir. İlgili makaleye bir göz atmak, sistemin başlığı nasıl elde ettiği konusunda size fikir verecektir.

4.4. Modelin Mimarisi

Modelin mimari yapısını anlatan .json dosyası şu şekildedir:

```
{
  "class_name": "Model",
  "config": {
    "name": "model_1",
    "layers": [
      {
        "name": "encoder_inputs",
        "class_name": "InputLayer",
        "config": {
          "batch_input_shape": [
            null,
```

```

        null
    ],
    "dtype": "float32",
    "sparse": false,
    "name": "encoder_inputs"
},
"inbound_nodes": []
},
{
    "name": "encoder_embedding",
    "class_name": "Embedding",
    "config": {
        "name": "encoder_embedding",
        "trainable": true,
        "batch_input_shape": [
            null,
            500
        ],
        "dtype": "float32",
        "input_dim": 5002,
        "output_dim": 100,
        "embeddings_initializer": {
            "class_name": "RandomUniform",
            "config": {
                "minval": -0.05,
                "maxval": 0.05,
                "seed": null
            }
        },
        "embeddings_regularizer": null,
        "activity_regularizer": null,
        "embeddings_constraint": null,
        "mask_zero": false,
        "input_length": 500
    },
    "inbound_nodes": [
        [
            [
                "encoder_inputs",
                0,
                0,
                {}
            ]
        ]
    ]
},
{
    "name": "decoder_inputs",

```

```

"class_name": "InputLayer",
"config": {
    "batch_input_shape": [
        null,
        null,
        2001
    ],
    "dtype": "float32",
    "sparse": false,
    "name": "decoder_inputs"
},
"inbound_nodes": []
},
{
    "name": "encoder_lstm",
    "class_name": "LSTM",
    "config": {
        "name": "encoder_lstm",
        "trainable": true,
        "dtype": "float32",
        "return_sequences": false,
        "return_state": true,
        "go_backwards": false,
        "stateful": false,
        "unroll": false,
        "units": 100,
        "activation": "tanh",
        "recurrent_activation": "sigmoid",
        "use_bias": true,
        "kernel_initializer": {
            "class_name": "VarianceScaling",
            "config": {
                "scale": 1,
                "mode": "fan_avg",
                "distribution": "uniform",
                "seed": null
            }
        },
        "recurrent_initializer": {
            "class_name": "Orthogonal",
            "config": {
                "gain": 1,
                "seed": null
            }
        },
        "bias_initializer": {
            "class_name": "Zeros",
            "config": {}
        }
    }
}

```

```

    },
    "unit_forget_bias": true,
    "kernel_regularizer": null,
    "recurrent_regularizer": null,
    "bias_regularizer": null,
    "activity_regularizer": null,
    "kernel_constraint": null,
    "recurrent_constraint": null,
    "bias_constraint": null,
    "dropout": 0,
    "recurrent_dropout": 0,
    "implementation": 2
  },
  "inbound_nodes": [
    [
      [
        "encoder_embedding",
        0,
        0,
        {}
      ]
    ]
  ]
},
{
  "name": "decoder_lstm",
  "class_name": "LSTM",
  "config": {
    "name": "decoder_lstm",
    "trainable": true,
    "dtype": "float32",
    "return_sequences": true,
    "return_state": true,
    "go_backwards": false,
    "stateful": false,
    "unroll": false,
    "units": 100,
    "activation": "tanh",
    "recurrent_activation": "sigmoid",
    "use_bias": true,
    "kernel_initializer": {
      "class_name": "VarianceScaling",
      "config": {
        "scale": 1,
        "mode": "fan_avg",
        "distribution": "uniform",
        "seed": null
      }
    }
  }
}

```

```

    },
    "recurrent_initializer": {
        "class_name": "Orthogonal",
        "config": {
            "gain": 1,
            "seed": null
        }
    },
    "bias_initializer": {
        "class_name": "Zeros",
        "config": {}
    },
    "unit_forget_bias": true,
    "kernel_regularizer": null,
    "recurrent_regularizer": null,
    "bias_regularizer": null,
    "activity_regularizer": null,
    "kernel_constraint": null,
    "recurrent_constraint": null,
    "bias_constraint": null,
    "dropout": 0,
    "recurrent_dropout": 0,
    "implementation": 2
},
"inbound_nodes": [
    [
        "decoder_inputs",
        0,
        0,
        {}
    ],
    [
        "encoder_lstm",
        0,
        1,
        {}
    ],
    [
        "encoder_lstm",
        0,
        2,
        {}
    ]
]
}
{

```

```

    "name": "decoder_dense",
    "class_name": "Dense",
    "config": {
        "name": "decoder_dense",
        "trainable": true,
        "dtype": "float32",
        "units": 2001,
        "activation": "softmax",
        "use_bias": true,
        "kernel_initializer": {
            "class_name": "VarianceScaling",
            "config": {
                "scale": 1,
                "mode": "fan_avg",
                "distribution": "uniform",
                "seed": null
            }
        },
        "bias_initializer": {
            "class_name": "Zeros",
            "config": {}
        },
        "kernel_regularizer": null,
        "bias_regularizer": null,
        "activity_regularizer": null,
        "kernel_constraint": null,
        "bias_constraint": null
    },
    "inbound_nodes": [
        [
            [
                "decoder_lstm",
                0,
                0,
                {}
            ]
        ]
    ]
},
"input_layers": [
    [
        "encoder_inputs",
        0,
        0
    ],
    [
        "decoder_inputs",

```

```
        0,  
        0  
    ],  
    ],  
    "output_layers": [  
        [  
            "decoder_dense",  
            0,  
            0  
        ]  
    ]  
},  
"keras_version": "2.3.1",  
"backend": "tensorflow"  
}
```



BÖLÜM 5. DENEYSEL SONUÇLAR

5.1. Sonuçlar

Sonuçlara geçmeden önce, aşağıda sonuçlarda kullanacağımız bazı kavramlardan bahsedilmiştir.

İlgili modelin aşırı uyum gösterme (overfitting) durumunda olup olmadığını öğrenmek adına, veri bilimcileri çapraz doğrulama adı verilen bir teknik kullanmaktadırlar. Bu teknikte verileri iki bölüme ayırmaktadırlar. Bunlar eğitim seti ve doğrulama setidir. Eğitim seti modeli eğitmek için kullanılırken, doğrulama seti yalnızca modelin performansını değerlendirmek için kullanılmaktadır.

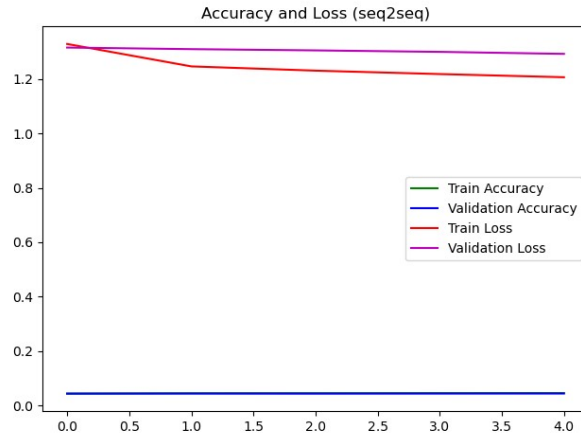
Kayıp fonksiyonu (train loss), veri setinin eğitimindeki hata oranıdır.

İsabet fonksiyonu (train accuracy), eğitilmiş ağ üzerinde veri doğrulama seti çalıştırıldıktan sonraki hata oranıdır.

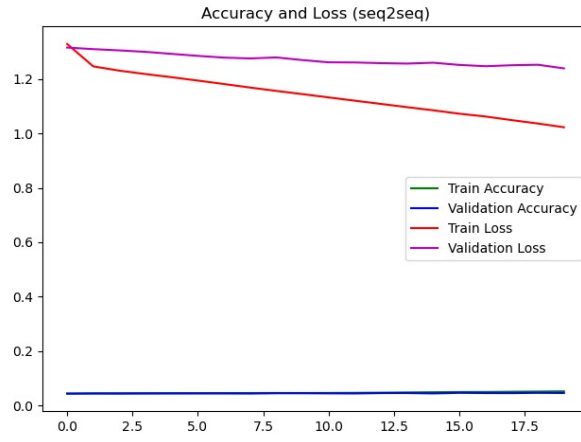
Doğrulama kayıp fonksiyonu (validation loss), çapraz doğrulama verileri için maliyet fonksiyonu değeridir.

Doğrulama isabet fonksiyonu (validation accuracy), çapraz doğrulama verileri için isabet fonksiyonu değeridir. Bu değer ne ise, model yeni verilerde o doğrulukta çalışmaktadır.

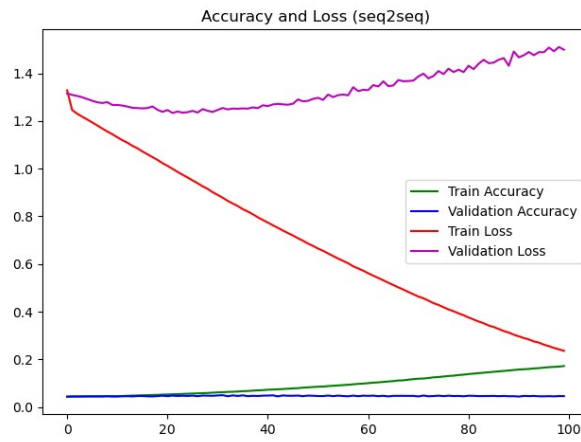
Çıktılarımız, bir metnin en özet hali olacak şekilde başlıklar türetme üzerinedir. Sistem, öncelikle anadilimiz Türkçe için özetleme esas alınarak; 5, 20, 100 ve son olarak 250 epoch değerleri ve 5000'e yakın örnekle ile çalıştırılmıştır. İlgili sonuçlar elde edilmiştir: (Bknz. Şekil 5.1., Şekil 5.2., Şekil 5.3. ve Şekil 5.4.)



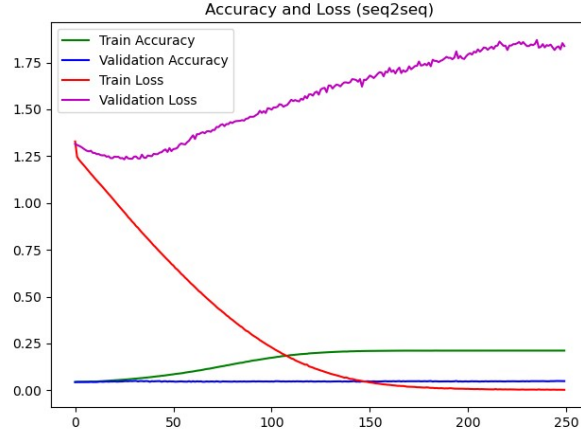
Şekil 5.1. Beş (5) Epoch İçin Grafik.



Şekil 5.2. Yirmi (20) Epoch İçin Grafik.



Şekil 5.3. Yüz (100) Epoch İçin Grafik.



Şekil 5.4. İkiyüz elli (250) Epoch İçin Grafik.

Kayıp fonksiyonu (train loss), 5 ve 20 epoch değerleri için aşırı değişmemişken, 100 epoch sonrasında 0.23 ve 250 epoch sonrasında ise 0 değerine düşmüştür. Yani sıfırlanmıştır.

İsabet fonksiyonu (train accuracy), 5 ve 20 epoch değerleri için hemen hemen hiç değişmemiştir. Ufak değişimler olsa da grafikte belirli olacak düzeyde değildir. 100 epoch için ise 0.17 ye, 250 epoch için 0.21 değerine ulaşmıştır.

Doğrulama değerleri için ise doğrulama isabet fonksiyonu (validation accuracy) ufak değerlerde değişimler gösterirken, doğrulama kayıp fonksiyonu (validation loss) genel anlamda artmıştır. Sonuç olarak sistem, kayıp fonksiyonunun sıfırlanması ve isabet fonksiyonunun % 21 gibi oranlara ulaşması ile başarılı bir gidişat göstermiştir.

İlgili başlıklar, önceleri 5 epoch için “ve”, “son” vb. kelime tekrarları şeklinde tahminler yaparken, gitgide doğru cümle yapıları artış göstermiş ve en son 250 epoch’da yüksek oranda benzeşen veya tamamen aynı olan başlıklar bile gözlenmiştir. Bunlardan biri “Marmara’da sıcaklık artıyor” cümlesidir.

Beş (5) epoch için özet örnekleri şu şekildedir:

Üretilen Başlık: son son ve ve

Orjinal Başlık: Asansör kabininde feci kaza - Son Dakika Haberler

Üretilen Başlık: son son ve

Orjinal Başlık: Hepimiz #ŞuleÇetİçinADALET istiyoruz

Üretilen Başlık: bakan ve ve ve

Orjinal Başlık: Patates ithalatında 20 Nisan'a kadar gümrük vergisi sıfırlandı

Üretilen Başlık: bakan ve ve ve

Orjinal Başlık: Mars'a tek yönlü yolculuk projesi sunan şirket iflas etti

Üretilen Başlık: son son ve ve

Orjinal Başlık: 2 aydır nöbet tutuyordu! Geldiklerini görünce dehşet saçtı

Üretilen Başlık: bakan ve ve ve

Orjinal Başlık: Soğukla gelen güzellik yürekleri ısıtıyor

Üretilen Başlık: son son ve ve

Orjinal Başlık: Huawei, Mobil Dünya Kongresi'ne 5G ile damga vurdu

Üretilen Başlık: son son ve

Orjinal Başlık: TÜİK'ten rakamlarla kadınlar

Üretilen Başlık: son son ve

Orjinal Başlık: Bavul ticaretini kayıt altına alıyoruz

Üretilen Başlık: bakan ve ve ve

Orjinal Başlık: Marmara'da sıcaklık artıyor

Yirmi (20) Epoch İçin Özet Örnekleri şu şekildedir:

Üretilen Başlık: son dakika - - haberleri

Orjinal Başlık: Asansör kabininde feci kaza - Son Dakika Haberler

Üretilen Başlık: son dakika: yeni yeni önemli - | ekonomi

Orjinal Başlık: Hepimiz #ŞuleÇetİçinADALET istiyoruz

Üretilen Başlık: son dakika... ve ve bin bin lira bin lira

Orjinal Başlık: Patates ithalatında 20 Nisan'a kadar gümrük vergisi sıfırlandı

Üretilen Başlık: cumhurbaşkanı ve bakanı bakanı bakanı ve ve bin ile ile için için

Orjinal Başlık: Mars'a tek yönlü yolculuk projesi sunan şirket iflas etti

Üretilen Başlık: israil ve fetö ve 2 yaralı

Orjinal Başlık: 2 aydır nöbet tutuyordu! Geldiklerini görünce dehşet saçtı

Üretilen Başlık: son dakika... yeni yeni bir bir | ekonomi

Orjinal Başlık: Soğukla gelen güzellik yürekleri ısıtıyor

Üretilen Başlık: bakan dakika... bakanı bakanı ve ve ile ile ile ile bir bir bir

Orjinal Başlık: Huawei, Mobil Dünya Kongresi'ne 5G ile damga vurdu

Üretilen Başlık: son - - son dakika dakika haberler

Orjinal Başlık: TÜİK'ten rakamlarla kadınlar

Üretilen Başlık: son dakika... ve yeni için için bir bir

Orjinal Başlık: Bavul ticaretini kayıt altına alıyoruz

Üretilen Başlık: israil ve kar kar

Orjinal Başlık: Marmara'da sıcaklık artıyor

Yüz (100) epoch için özet örnekleri şu şekildedir:

Üretilen Başlık: beşiktaş'ta feci feci feci - son dakika haberler

Orjinal Başlık: Asansör kabininde feci kaza - Son Dakika Haberler

Üretilen Başlık: bir ilk ilk kez bir nasıl yapılır?

Orjinal Başlık: Hepimiz #ŞuleÇetİçinADALET istiyoruz

Üretilen Başlık: bakan açıkladı: yüzde 1,5 diğer gümrük kadar saat

Orjinal Başlık: Patates ithalatında 20 Nisan'a kadar gümrük vergisi sıfırlandı

Üretilen Başlık: çin'de ile suudi arabistan'a bakanı şirket şirket tartışması etti

Orjinal Başlık: Mars'a tek yönlü yolculuk projesi sunan şirket iflas etti

Üretilen Başlık: israil'den gazze'ye hava saldırısı

Orjinal Başlık: 2 aydır nöbet tutuyordu! Geldiklerini görünce dehşet saçtı

Üretilen Başlık: kadınlar ay depresyon için araç

Orjinal Başlık: Soğukla gelen güzellik yürekleri ısıtıyor

Üretilen Başlık: son dakika... bakan kurum 5g diğer önemli girdi

Orjinal Başlık: Huawei, Mobil Dünya Kongresi'ne 5G ile damga vurdu

Üretilen Başlık: bakan albayrak'tan kadınlar

Orjinal Başlık: TÜİK'ten rakamlarla kadınlar

Üretilen Başlık: sağlık seçim seçim nasıl

Orjinal Başlık: Bavul ticaretini kayıt altına alıyoruz

Üretilen Başlık: marmara'da sıcaklık artıyor

Orjinal Başlık: Marmara'da sıcaklık artıyor

İkiyüzelli (250) epoch için özet örnekleri şu şekildedir:

Üretilen Başlık: asansör - feci kaza - son dakika haberler

Orjinal Başlık: Asansör kabininde feci kaza - Son Dakika Haberler

Üretilen Başlık: bakan kurum: tek bir kez daha çok 11 şekilde

Orjinal Başlık: Hepimiz #ŞuleÇetİçinADALET istiyoruz

Üretilen Başlık: patates karşı 20 yıllık kadar gümrük vergisi

Orjinal Başlık: Patates ithalatında 20 Nisan'a kadar gümrük vergisi sıfırlandı

Üretilen Başlık: abd tek yolculuk yolculuk projesi etti

Orjinal Başlık: Mars'a tek yönlü yolculuk projesi sunan şirket iflas etti

Üretilen Başlık: manisa'da terör operasyonu: 3 ton bulundu

Orjinal Başlık: 2 aydır nöbet tutuyordu! Geldiklerini görünce dehşet saçtı

Üretilen Başlık: milli başakşehir için 40 yıldır

Orjinal Başlık: Soğukla gelen güzellik yürekleri ısıtıyor

Üretilen Başlık: mobil dünya dünya kadınlar 5g ile davası

Orjinal Başlık: Huawei, Mobil Dünya Kongresi'ne 5G ile damga vurdu

Üretilen Başlık: tanzim satış kadınlar

Orjinal Başlık: TÜİK'ten rakamlarla kadınlar

Üretilen Başlık: yemen'de duvarlara altına altına

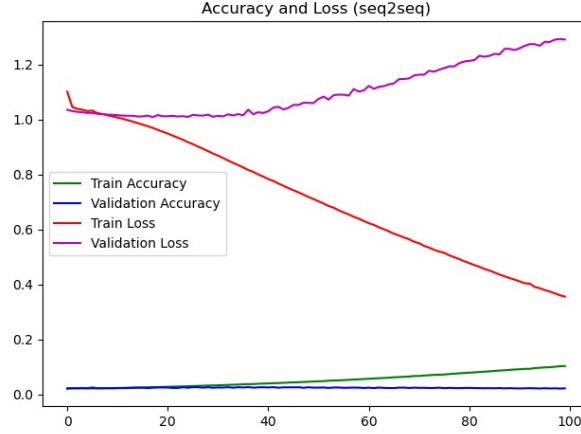
Orjinal Başlık: Bavul ticaretini kayıt altına alıyoruz

Üretilen Başlık: marmara'da sıcaklık artıyor

Orjinal Başlık: Marmara'da sıcaklık artıyor

Genel anlamda epoch sayısı artışı ile sonuçlar düzelmiş ve en son 250 epoch'da daha düzgün hale ulaşmıştır. Daha yüksek epoch sayısında daha iyi sonuçlar vereceği kanısına ulaşmıştır.

Sistemin dil performansı açısından karşılaştırılması adına, İngilizce için de sistem 5, 20 ve 100 epoch için çalıştırılmıştır (veri sayısı sabit tutularak). Genel anlamda grafikler Türkçe için olanlarla benzer eğilimler göstermiştir ve son olarak 100 epoch için ilgili grafik elde edilmiştir (Şekil 5.5.).



Şekil 5.5. Yüz (100) Epoch İçin Grafik (İngilizce Dili İçin).

Kayıp fonksiyonu ve isabet fonksiyonu Türkçe için sırasıyla 0.23 ve 0.17 iken, İngilizce için kayıp fonksiyonu 0.35, isabet fonksiyonu 0.1 dir.

Doğrulama değerleri için ise doğrulama isabet fonksiyonu (validation accuracy) ufak değerlerde değişimler gösterirken, doğrulama kayıp fonksiyonu (validation loss) genel anlamda artmıştır ve davranış olarak Türkçe ile benzer yapıda seyretmiştir. Sonuç olarak sistem, kayıp fonksiyonunun azaltılması ve isabet fonksiyonunun % 10 gibi oranlara ulaşması ile kısmen başarılı bir gidişat göstermiştir.

İlgili başlıklar, önceleri 5 epoch için “the”, “to” vb. kelime tekrarları şeklinde tahminler yaparken, gitgide doğru cümle yapıları artış göstermiş ve en son 100 epoch’da benzeşen başlıklar gözlenmiştir. İngilizce için benzeşen birkaç cümle örneği şu şekildedir:

Üretilen Başlık: police arrest responsibility for attacks in north

Orjinal Başlık: Taliban claim responsibility for Kabul attack

Üretilen Başlık: the battle of the clinton – a challenge of a results voter campaign

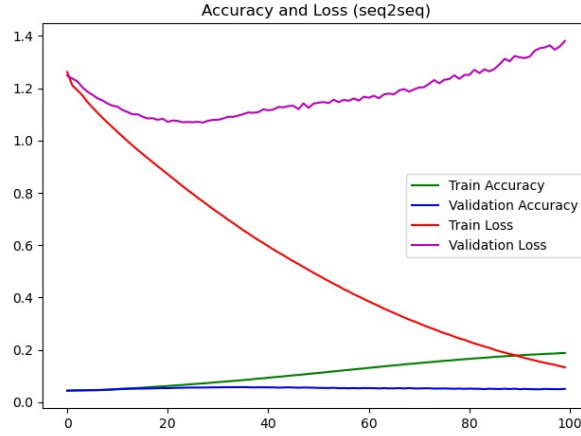
Orjinal Başlık: Paul Manafort resigns from Trump campaign

Üretilen Başlık: culture

Orjinal Başlık: Strategic Culture

İlgili cümleler anahatları ile benzeşenler düşünülerek tarafımdan seçilmiştir. Türkçe için 100 epoch sayısında tamamen aynı ya da oldukça benzeşen sonuçlar bulunmasına rağmen, İngilizce için bu durum daha düşüktür. Zaten bu durum ilgili sonuçların rakamsal değerleri ile de görülmektedir. Sonuç olarak İngilizce dili için başarımlar, Türkçe'ye göre oldukça düşük çıkmıştır.

Son olarak sistem Türkçe için daha çok veri ile (10000 veri) test edilmiştir. 5 ve 20 epoch için, az verideki ile benzer seyir göstermiş ancak doğal bir sonuç olarak kayıp fonksiyonu daha hızlı azalmıştır. Çok veri ile 100 epoch için grafik şu şekildedir (Şekil 5.6.):



Şekil 5.6. Yüz (100) Epoch İçin Grafik (Türkçe Dili Çok Veri İçin).

Türkçe için kayıp fonksiyonu 0.23, isabet fonksiyonu 0.17 dir. Türkçe çok veri için sırasıyla 0.13 ve 0.18 iken, isabet oranındaki artış % 1'dir.

Doğrulama değerleri için ise doğrulama isabet fonksiyonu (validation accuracy) ufak değerlerde değişimler gösterirken, doğrulama kayıp fonksiyonu (validation loss) genel anlamda artmıştır ve 1.38 değerinde son bulmuştur (daha az veride 1.49). Sonuç olarak sistem, kayıp fonksiyonunun oldukça azaltılması ve isabet fonksiyonunun % 18 gibi oranlara ulaşması ile başarılı bir gidişat göstermiştir.

İlgili başlıklar, önceleri 5 epoch için “ve”, “son” vb. kelime tekrarları şeklinde tahminler yaparken, gitgide doğru cümle yapıları artış göstermiş ve en son 100 epoch'da benzeşen başlıklar gözlenmiştir.

Yüz (100) epoch'da, çok veri için özet örnekleri şu şekildedir:

Üretilen Başlık: başkan erdoğan canlı yayında diyen chp diyen görüştü

Orjinal Başlık: Başkan Recep Tayyip Erdoğan hapse girmesine neden olan Ziya Gökalp'ın Asker Duası şiirini aynı meydanda tekrar okudu

Üretilen Başlık: thy'den iş iş 15 başvuru

Orjinal Başlık: THY'den gençlere iş fırsatı! İşte başvuru şartları...

Üretilen Başlık: fetö'nün tarihi kritik 13 gözaltı

Orjinal Başlık: 'Erdoğan'a hakaret' davasında hakime sürgün cezası

Üretilen Başlık: beşiktaş fenerbahçe maçı için yola yola çıktı

Orjinal Başlık: Sadece Ndiaye

Üretilen Başlık: beşiktaş fenerbahçe devam ediyor

Orjinal Başlık: Metrobüs bekleyen astronot Beşiktaş - Bursaspor maçında!

Üretilen Başlık: fetö'nün gizli yapılanmasına operasyon: 7 gözaltı

Orjinal Başlık: FETÖ'nün kripto yapılanmasına operasyon

Üretilen Başlık: fenerbahçe'ye arslan'dan iyi haber

Orjinal Başlık: Fenerbahçe'ye Tolgay Arslan'dan iyi haber

Üretilen Başlık: tbmm başkanı bu taziye mesajı

Orjinal Başlık: TBMM Başkanı Şentop'tan Etiyopya için taziye mesajı

Üretilen Başlık: müge anlı bu canlı yayında abdullah gazi dolu bir son dakika yeni askerlik açıkladı

Orjinal Başlık: Müge Anlı'nın programında şok itiraf! Abdullah Gazi A.'yı büyük kızı öldürmüştü...

Üretilen Başlık: mansur albayrak'tan flaş açıklama: dünya dünya günü günü

Orjinal Başlık: Binali Yıldırım 'PUBG' oynadı: "Torunlar oynuyor ama öğretmiyor" dedi

Türkçe için 100 epoch sayısında, % 1'lik fark olmasından ötürü çok büyük değişimler olmamıştır. Ancak doğru tahminlerde kısmen artış gözlenmiştir. Sonuç

olarak, Türkçe için veri sayısının artması aşırı bir değişime yol açmamıştır. Ancak 250 epoch ile yapılan testte, normal veri setinden elde edilen sonuçlar, önceki kısımlarda anlatıldığı üzere oldukça başarılı sonuçlar vermiştir.

5.2. Öneriler

Bu tez çalışmasında derin öğrenme ile çıkarımsal metin özetleme modeli geliştirilmiştir. Mevcut modeller incelenmiş ve var olan bir açık kaynak çalışma temel alınmıştır. Sistemde, algoritmik yapılar genel manada, Keras'ın temel dinamikleri üzerinde çalışmaktadır.

Model, özellikle anadilimiz olan Türkçe için başarılı sonuçlar vermiştir. Bu yönde yapılacak yeni çalışmalara ufuk açması amaçlanmıştır. Ayrıca daha düzgün hazırlanmış, daha çok sayıda verisiyle ve daha çok epoch sayısı, başarımları yüksek sistemler kurulabileceği düşüncesinde öncü olmuştur.

KAYNAKLAR

- [1] Güran A., Arslan S.N., Kılıç E., Diri B., Metin Özetleme İçin Cümle Seçim Metotları, 2014 IEEE 22nd Signal Processing and Communications Applications Conference (SIU 2014).
- [2] Uzundere E., Dedja E., Diri B., Amasyalı M.F., Türkçe Haber Metinleri İçin Otomatik Özetleme, ASYU 2008 Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu.
- [3] Wang P., Goertzel B., Franklin S., Artificial General Intelligence 2008, Proceedings of the First AGI Conference.
- [4] Rende F.Ş., Bütün G., Karahan Ş., Derin Öğrenme Algoritmalarında Model Testleri: Derin Testler, Bilişim Teknolojileri Enstitüsü, TÜBİTAK BİLGEM, Gebze, Kocaeli, 2017.
- [5] Kaya A., Derin Öğrenme ile Rakam Öğretme, İstanbul, 2017.
- [6] Chan G., Extractive Text Summarization with Deep Learning Senior Project Final Report, 2018.
- [7] Altan Z., Orhan Z., Anlam Belirsizliği İçeren Türkçe Sözcüklerin Hesaplamalı Dilbilim Uygulamalarıyla Belirginleştirmesi, 2015.
- [8] Chomsky N., Knowledge of Language: Its Nature, Origin, and Use. New York, 1986.
- [9] Young T., Hazarika D., Poria S., Cambria E., Recent Trends in Deep Learning Based Natural Language Processing. IEEE Computational Intelligence Magazine, 55 – 75, 2018.
- [10] Cambria E., White B., Jumping NLP curves: A review of natural language processing research. IEEE Computational Intelligence Magazine 9, 48–57, 2014.

- [11] Gu X., Zhang H., Zhang D., Kim S., Deep API learning. In Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering (pp. 631-642). ACM, November, 2016.
- [12] Deng L., Yu D., Deep Learning: Methods and Applications. Foundations and Trends® in Signal Processing, 7(3-4), 197-387, 2014.
- [13] [medium.com/türkiye/yapay-zekanın-tarihçesi-ve-gelişim-süreci-cb4c73deb01d.](https://medium.com/türkiye/yapay-zekanın-tarihçesi-ve-gelişim-süreci-cb4c73deb01d), Erişim Tarihi: 30.05.2020.
- [14] McCulloch W.S., Pitts W., A Logical Calculus of the Idea Immanent in Nervous Activity. Bulletin of Mathematical Biophysics, 1943.
- [15] Socher R., Bengio Y., Manning C.D., Deep learning for NLP (without magic). Tutorial Abstracts of ACL 2012, 5.
- [16] Gupta V., Lehal G.S., A Survey of Text Summarization Extractive Techniques, Journal of Emerging Technologies in Web Intelligence, Vol. 2, No. 3, August 2010.
- [17] Gündoğdu Ö.E., Duru N., Türkçe Metin Özetlemede Kullanılan Yöntemler, 2016.
- [18] Uğur A., Kınacı A.C., Yapay Zeka Teknikleri ve Yapay Sinir Ağları Kullanılarak Web Sayfalarının Sınıflandırılması, inet-tr'06 – XI. Türkiye'de İnternet Konferansı Bildirileri, 21-23 Aralık 2006, TOBB Ekonomi ve Teknoloji Üniversitesi, Ankara.
- [19] Ataseven B., Yapay Sinir Ağları ile Öngörü Modellemesi, İstanbul Kültür Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, İşletme Bölümü, 2013.
- [20] Elmas Ç., Yapay Sinir Ağları (Kuram, Mimari, Eğitim, Uygulama), Seçkin Yayınları, Ankara, 2003.
- [21] Güneri N., Apaydın A., Öğrenci Başarılarının Sınıflandırılmasında Lojistik Regresyon Analizi ve Sinir Ağları Yaklaşımı, Ticaret ve Turizm Eğitim Fakültesi Dergisi, Yıl: 2004, Sayı: 1, ss. 170 – 188, 2004.
- [22] [thedacs.com/techs/neural/neural.title.php.](https://thedacs.com/techs/neural/neural.title.php), Erişim Tarihi: 15.03.2020.
- [23] Alpaydın E., Introduction to Machine Learning Second Edition, The MIT Press, Cambridge, Massachusetts, London, England, 2009.
- [24] Çetin E., Yapay Zeka Uygulamaları, 4. baskı, 2011.

- [25] Tiken C., Ensari T., Derin Öğrenme Uygulamaları, Haziran 2015, İstanbul, TÜRKİYE.
- [26] Glaeser G., Strouhal E., Kempelen's Chess Playing Pseudo-Automaton and Racknitz Explanation of its Controls, University of Applied Arts Vienna, Vienna, Austria, 2000.
- [27] Turing A., Computing Machinery and Intelligence, Mind, 442, 1950.
- [28] medium.com/technical-library/derin-ogrenme-makine-ogrenmesi-ve-yapay-zeka-arasindaki-fark-nedir-25c1971d7433., Erişim Tarihi: 04.03.2020.
- [29] www.ahmetcevahircinar.com.tr/2017/08/09/matlab-ile-derin-ogrenmeye-giris-introducing-deep-learning-with-matlab., Erişim Tarihi: 04.03.2020.
- [30] Hochreiter S., Schmidhuber J., Long Short-Term Memory, Neural Computation, 9(8), 1735-1780, 1997.
- [31] Süzen A.A., Yıldız Z., Yılmaz T., LSTM Tabanlı Derin Sinir Ağı ile Ayak Taban Basınç Verilerinden VKİ Durumlarının Sınıflandırılması, BEÜ Fen Bilimleri Dergisi, 8(4), 1392-1398, 2019.
- [32] Samui S., Chakrabarti I., Ghosh S.K., Tensor-Train Long Short-Term Memory for Monaural Speech Enhancement, arXiv preprint arXiv:1812.10095, 2018.
- [33] Greff K. et al., LSTM: A Search Space Odyssey, Transactions on Neural Networks and Learning Systems, 2017.
- [34] Sezer Ö.B., Zaman Serisi Verilerinden Derin Yapay Sinir Ağları ile Analizi ve Eniyilemesi: Finansal Tahmin Algoritmaları, TOBB Ekonomi ve Teknoloji Üniversitesi, Fen Bilimleri Enstitüsü, 2018.
- [35] veribilimcisi.com/2018/06/04/otomatik-kodlayici-auto-encoder., Erişim Tarihi: 10.03.2020.
- [36] Hinton G.E., Salakhutdinov R.R., Reducing the dimensionality of data with neural networks, Science, 313(5786), 504– 507, 2006.
- [37] Hinton G.E., Zemel, R.S., Autoencoders, minimum description length and Helmholtz free energy, In Advances in neural information processing systems, 3-10, 1994.

- [38] Baldi P., Autoencoders, unsupervised learning, and deep architectures. In Proceedings of ICML Workshop on Unsupervised and Transfer Learning, 37-49, 2012.
- [39] Wani M.A., Bhat F.A., Afzal S., Khan A.I., Studies in Big Data, Singapore: Springer, Vol. 57, 2020.
- [40] Ravi D., Wong C., Deligianni F., Berthelot M., Andreu-Perez J., Lo B., Yang G.Z., Deep learning for health informatics, in IEEE Journal of Biomedical and Health Informatics, 21(1), 4-21., 2017.
- [41] Yücer Ş., Üç Boyutlu İskelet Verilerinden Metrik Öğrenme Tabanlı Hareket Tanıma, Gebze Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi, 2018.
- [42] Kaynar O., Aydın Z., Görmez Y., Sentiment Analizinde Öznitelik Düşürme Yöntemlerinin Oto Kodlayıcı Derin Öğrenme Makinaları ile Karşılaştırılması, Bilişim Teknolojileri Dergisi, 10(3), 319-326, 2017.
- [43] Bozşahin C., Zeyrek D., Pembeci I., Computer-aided Learning of Turkish Morphology, Debrecen, Hungary, Proc. Joint Conf. ACH ALLC, 1998.
- [44] teknobur.com/google-lens-nedir-nasil-kullanilir, Erişim Tarihi: 28.11.2019.
- [45] keras.io, Erişim Tarihi: 15.03.2020.
- [46] Akbaş E., Derin Yapay Sinir Ağları ve Derin Öğrenme'ye Kısa Bir Giriş, Ortadoğu Teknik Üniversitesi, Bilgisayar Mühendisliği, BMO Semineri, 24 Kasım 2018
- [47] MacCormick J., Geleceği Değiştiren Dokuz Algoritma, Tübitak Popüler Bilim Kitapları.
- [48] Mihalcea R., Tarau P., TextRank: Bringing Order into Text, University of North Texas, Department of Computer Science.

ÖZGEÇMİŞ

Burak ERHANDI, 04.08.1988'de Eskişehir'de doğdu. İlk, orta ve lise eğitimini Eskişehir'de tamamladı. 2006 yılında Fatih Anadolu Lisesi'nden mezun oldu. 2007 yılında başladığı Sakarya Üniversitesi Metalurji ve Malzeme Mühendisliği Bölümü'nü bir sene sonra bırakarak, yeniden 2009 senesinde ESOGÜ Endüstri Mühendisliğine başladı ve 2013 yılında bitirdi. 2014 yılında girdiği ESOGÜ İstatistik yüksek lisansını tez aşamasında bırakarak, Sakarya Üniversitesi Bilgisayar ve Bilişim Mühendisliği yüksek lisansına 2016 senesinde başladı. Küçüklükten hayali olan yazılımcılık mesleğini kendine meslek olarak seçti. Kariyerini yazılım üzerine devam ettirmek yönünde ilerledi. Okul ve bireysel çalışmalarında bu alana odaklandı. 2015-2019 arasında bir devlet kurumu ve bir özel sektör tecrübesi oldu ve ilgili yüksek lisanslar ile beraber iş hayatını devam ettirdi. Şu an tam zamanlı çalışmamakla beraber, Ekim 2018'den beridir serbest zamanlı (freelance) yazılım, web tasarım, 3d modelleme vb. işlere devam etmektedir.