T.C.

KIRIKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DOĞAL DİL İŞLEME TEKNİKLERİNİ VE DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK SOSYAL AĞLARDA SPAM TESPİTİ

REZAN BAKIR

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

DANIŞMAN

Prof. Dr. Hasan ERBAY

KIRIKKALE- 2022

T.C.

KIRIKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DOĞAL DİL İŞLEME TEKNİKLERİNİ VE DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK SOSYAL AĞLARDA SPAM TESPİTİ

REZAN BAKIR

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DOKTORA TEZİ

DANIŞMAN
Prof. Dr. Hasan ERBAY

KIRIKKALE- 2022

Rezan BAKIR tarafından hazırlanan “DOĞAL DİL İŞLEME TEKNİKLERİNİ VE DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK SOSYAL AĞLARDA SPAM TESPİTİ ” adlı tez çalışması, aşağıdaki jüri tarafından OY BİRLİĞİ ile Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar mühendisliği Anabilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Hasan ERBAY

İmza……

Bilgisayar mühendisliği Anabilim Dalı, Türk Hava Kurum Üniversitesi

Bu tezin, kapsam ve kalite olarak DOKTORA Tezi olduğunu onaylıyorum

Başkan : Dr. Öğr. Üyesi. Fahrettin HORASAN

İmza ..……..

Bilgisayar Mühendisliği Anabilim Dalı, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak DOKTORA Tezi olduğunu onaylıyorum

Üye : Doç. Dr. Bülent Gürsel EMİROĞLU İmza ..……..

Bilgisayar Mühendisliği Anabilim Dalı, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak DOKTORA Tezi olduğunu onaylıyorum

Üye : Dr. Öğr. Üyesi. Hakan KÖR İmza ..……..

Bilgisayar Mühendisliği Anabilim Dalı, Hitit Üniversitesi

Bu tezin, kapsam ve kalite olarak DOKTORA Tezi olduğunu onaylıyorum

Üye : Dr Öğr. Üyesi. Ahmet Haşim YURTTAKAL İmza ..……..

Bilgisayar Mühendisliği Anabilim Dalı, Afyon Kocatepe Üniversitesi

Bu tezin, kapsam ve kalite olarak DOKTORA Tezi olduğunu onaylıyorum

Tez Savunma Tarihi: 12/08/2022

Jüri tarafından kabul edilen bu tezin DOKTORA Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

…….…….

Prof. Dr. Recep ÇALIN

ETİK BEYANI

Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygunolarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

(İmza)

Rezan BAKIR

13/08/2022

ÖZET

DOĞAL DİL İŞLEME TEKNİKLERİNİ VE DERİN ÖĞRENME ALGORİTMALARINI KULLANARAK SOSYAL AĞLARDA SPAM ALGILAMASI

BAKIR, Rezan[1]
Kırıkkale Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı, Doktora tezi
Danışman: Prof. Dr. Hasan ERBAY
Ağustos 2022, 123 sayfa

Kısa metin sınıflandırma problemi olarak kabul edilen sosyal ağlarda spam tespiti, metnin seyrekliği ve belirsizliği nedeniyle doğal dil işlemede zorlu bir görevdir. Sorunu çözmek için en önemli görevlerden biri güçlü bir metin gösterimi bulmaktır. Geleneksel Kelime gömme (word embedding) modelleri, yoğun vektörlerle kelimeleri temsil ederek veri seyrekliği problemini çözmektedir, ancak bu modellerin bazı problemleri etkili bir şekilde ele almalarını engelleyen bazı sınırlamaları vardır. Geleneksel kelime gömme yöntemlerinin maruz kaldığı en yaygın sınırlamalarından birisi, "kelime dağarcığı (Out Of Vocabulary)" olarak adlandırılan ve modelin sözlüğünde olmayan sözcükleri için herhangi bir vektör temsili sağlayamamasından çıkan problemidir. Bu modellerin karşılaştığı bir diğer problemi ise, bu tip modellerin, kelimenin cümle içindeki konumundan bağımsız olarak her bir kelime için yalnızca bir vektör verdiği bağlamdan bağımsız olarak temsil etmektedir. Bu sorunların üstesinden gelebilmek için, derin öğrenme teknikleriyle birlikte bağlamsal doğal dil işleme modelleri benimsenmiştir. Doğal dil işlemenin ana hedeflerinden biri, farklı bağlamlarda kelime anlamları ve benzerlikleri yakalama yeteneğini güçlendiren kelimelerin anlamlı bir temsilini geliştirmektir. Sonuç olarak bu tez çalışması, spam mesajlarını etkili bir şekilde tespit etmek amacıyla sosyal ağlardaki kısa metinlerin seyrekliğini ve diğer kısıtlamalarını ele almak için farklı modelleri önerilmiştir. Önerilen modelleri, üç kıyaslama veri seti üzerinde test ederek elde edilen sonuçları,

[1]Razan Ghanem: Yazarın çift vatandaşlığından dolayı, onun adı iki farklı şekilde yazılabilir

bu modellerin yüksek sınıflandırma doğruluk elde ettiğini ve sosyal ağlarda spam masajları tespit etmek için mevcut son teknoloji yöntemlerden daha iyi performans gösterdiğini görülmüştür.

Anahtar kelimeleri: Spam tespiti, Kelime Gösterilimi, Transformatörlerden Çift Yönlü Enkoder Gösterimleri, Derin öğrenme, Doğal dil işleme, Transformer, tekrarlayan sinir ağı

ABSTRACT

USING NATURAL LANGUAGE PROCESSING TECHNIQUES AND DEEP LEARNING ALGORITHMS FOR DETECTING SPAM ON SOCIAL NETWORKS

BAKIR, Rezan[2]

Kırıkkale University

Graduate School of Natural and Applied Sciences

Department of Computer Engineering, Ph. D. Thesis

Supervisor: Prof. Dr. Hasan ERBAY

August 2022, 123 pages

Spam detection on social networks, considered a short text classification problem, is a challenging task in natural language processing due to the sparsity and the ambiguity of the text. One of the key tasks to address such a problem is powerful text representation. Traditional word embedding models solve the data sparsity problem by representing words with dense vectors, but these models have some limitations that make them unable to handle some problems effectively. The most common limitation that traditional word embedding methods suffer from is the "out of vocabulary" problem in which they fail to provide any vector representation for words that are not in the model's dictionary. Another problem these models face is the independence from the context, in which the models output just one vector for each word regardless of the position of the word in the sentence. To overcome these problems, we relied on contextualized natural language processing models in combination with deep learning techniques. One of the main goals of natural language processing is developing a meaningful representation of words, that improves the ability to capture word senses and similarity in different contexts. Consequently, in this thesis, we proposed different models to handle the sparsity and other limitations of short text on social networks in order to detect spam messages effectively. The results obtained on three benchmark datasets stated that our proposed methods achieve high accuracy and outperform the existing state-of-the-art methods to detect spam on social networks.

[2] Razan GHANEM: The author’s name can be written in two different ways due to his dual citizenship.

Keywords: Spam detection, Word embedding, Bidirectional encoder representations from transformers, Natural language processing, Deep learning, Embedding from language model, Recurrent neural network, Transformer.

TEŞEKKÜR

İlk olarak, bu tez ve araştırmayı tamamlamak için bana güç ve cesaret veren yüce Allah'ıma binlerce kez şükrediyorum.

Tezimin hazırlanması esnasında hiçbir yardımı esirgemeyen Sayın Prof. Dr. Hasan ERBAY'a, tez çalışmalarım esnasında, bilimsel konularda yardımlarını aldığım, Sayın Doç. Dr. Bülent Gürsel EMİROĞLU'na, Sayın Dr.Öğr. Üyesi. Hakan KÖR'e teşekkür ederim.

Bu doktora tez çalışmamı hayatımdaki her aşamada bana destek veren ve bu günlere gelmemde üzerimde çok emeği olan ve maddi manevi desteğini üzerimden hiç eksik etmeyen Sevgili annem Hamida ARSALİ'ye ve kıymetli babam Rizk GHANEM'e sonsuz teşekkür ediyorum.

Yıllar süren benzersiz destek ve rehberlik veren, akademik ve profesyonel özlemlerimi tanımlamama yardımcı olan, inancımı veya kararlılığımı asla kaybetmeme yardım eden eşim Halit BAKIR' a sonsuz teşekkürlerimi sunarım.

Hayatıma geldiğinden beri bana mutluluk veren, hayatımın en güzel varlığı olan, her zaman bana destek veren, çocuğuma Ahmet BAKIR'a teşekkürlerimi sunarım.

Çalışmalarım boyunca maddi manevi destekleriyle beni hiçbir zaman yalnız bırakmayan kardeşlerime de sonsuz teşekkürler ederim.

Bu günlere gelmemde üzerimde emeği olan bütün öğretmenlerime, hocalarıma ve bana destek veren arkadaşlarıma teşekkür ederim.

Rezan BAKIR

İÇİNDEKİLER DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

ŞEKİL **Sayfa**

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

KISALTMALAR DİZİNİ

NLP	Doğai dil işleme
BOW	Kelime çantası
GloVe	Global vector
BERT	Transformatörlerden Çift Yönlü Kodlayıcı Temsilleri
RoBERTA	Sağlam Bir Şekilde Optimize Edilmiş BERT Ön Eğitim Yaklaşımı
DISTIL BERT	BERT'nin damıtılmış versiyonu
AI	Yapay Zekâ
SVM	Destek Vektör Makineleri
RNN	Tekrarlayan sinir ağları
CNN	Evrişimsel Sinir Ağları
LSTM	Uzun Kısa Vadeli Hafıza Ağları
BLSTM	çift yönlü Uzun Kısa Vadeli Hafıza Ağları
RF	Rastgele Orman
LR	Doğrusal regresyon
OOV	kelime dağarcığı
ELMO	Dil Modellerinden Gömme
POS	Metin Parçası Etiketleme
CBOW	Sürekli Sözcük Torbası
Distil RoBERTa	RoBERTa'nın damıtılmış versiyonu
ReLU	Doğrultulmuş lineer birim
GRU	Geçitli tekrarlayan birim
FCNN	Tam Bağlantılı sınır ağları
GAN	Çekişmeli üretici ağları
RBFNs	Yarıçapsal Temelli Ağları
MLPs	Çok Katmanlı Algılayıcılar
SOM	Kendi Kendini Düzenleyen Haritalar
RBMs	Kısıtlı Boltzmann Makineleri
URL	Tekdüzen Kaynak Bulucu
NB	NaiveBayes sınıflandırıcı
TF-IDF	Terim Frekansı-Ters Belge Frekansı
BPE	Bayt düzeyinde bayt çifti kodlaması
MCC	Matthews korelasyon katsayısı
TP	Gerçek pozitif
TN	Gerçek negatif
FN	Yanlış negatif
FP	Yanlış pofitif
DA	Data Augmentation –Veri Büyütme

SİMGELER DİZİNİ

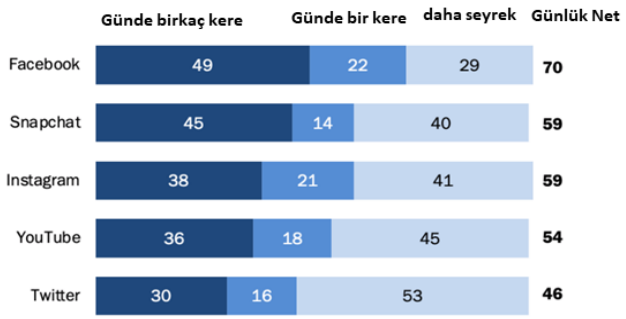
$\sum$	Toplam
$TF(t,d)$	d belgesinde t teriminin Terim Frekansı
$\in$	Kümede bir elemanın o kümeye ait olduğunu belirtmek için kullanılmaktadır
θ	Olabilirlik fonksiyonu
$\mathrm{H(P,Q)}$	P ve Q′ olayların Çapraz entropi

1. GİRİŞ

Son zamanlarda teknolojinin hızlı büyümesi nedeniyle, arkadaşlarımız ve akrabalarımızla iletişimimizi kolaylaştıran, farklı toplumlara açılmamıza katkı sağlayan sosyal paylaşım siteleri farklı ırk ve kültürlerle sosyal iletişimi kolay bir iş haline getirmiştir.

Sağladığı şaşırtıcı özellikler nedeniyle sosyal ağ sitelerinin popülerlikleri son on yılda önemli ölçüde artmıştır, Data Reportal'a göre Türkiye'de 2020 ile 2021 yılları arasındaki sosyal medya kullanıcı sayısı, %11' e varan artış oranı ulaşarak 60 milyona ulaşmıştır. Ayrıca, Ocak 2021'deki sosyal medya kullanıcıları, Türkiye'deki toplam nüfusun %70.8'ine ulaştığını aynı raporda söylenmektedir. Bahse geçen rapor, 2021'in başında dünya çapında 4.33 milyar sosyal medya kullanıcısı olduğunu ve bu da toplam küresel nüfusun yüzde 55'inden fazlasına eşit olduğunu göstermiştir (KEMP 2021).

Ayrıca, ABD'li yetişkinlerle yapılan ulusal çaplı bir anket kullanılarak 2021'de sosyal medya kullanımı ölçülmüştür. Şekil 1.1'de yer alan anket sonuçlarına göre her yüz Facebook kullanıcısından yetmişi Facebook sitesini her gün ziyaret etmiş, bunların %49'u ise siteyi günde birkaç kez ziyaret etmiştir. Ayrıca, anket katılımcılarının %54'ü YouTube web sitesini günlük olarak kullandığı ve %36'sı siteyi günde birkaç kez ziyaret ettiğini belirtilmektedir. Diğer sosyal medya ağlarıyla karşılaştırıldığında, Twitter'ın daha az kullanıldığı, ankete katılanlarının %46'sının siteyi her gün ziyaret ettiği söylenmiştir (Brooke Auxir 2021).



Şekil 1.1 ABD'li yetişkinlerine yönelik bir anketine göre sosyal medya kullanımı

Sosyal ağlar birçok avantajlarına rağmen kullanıcıyı, spam gibi birçok sorunla karşı karşıya bırakmaktadır. Spam, sosyal medya ağ kullanıcılarının yaşadığı en büyük sorunlardan biridir ve sosyal spam mesajları, kötü niyetli bağlantılar, reklamlar veya herhangi düşük kaliteli içerik gibi sosyal medya üzerinden gönderilen gereksiz veya istenmeyen mesajları olarak tanımlanmaktadır.

Sosyal ağlardaki spam mesajları, farklı formatlarda bulunabilmektedir, örneğin:

1) Aynı veya çok benzer metinlerle sürekli tekrarlanan veya bir dizi yorumdan oluşan toplu mesajlar,
2) Küfür veya hakaret içeren mesajlar,
3) Belirli bir kişiye veya kişilere karşı hafif veya şiddetli bir şekilde aşağılayıcı kelimeleri içeren mesajlar,
4) Bir kullanıcıya veya bilgisayara uygunsuz bir şekilde zarar verecek, yanlış yönlendirecek veya başka şekilde zarar verecek kötü niyetli bağlantılar içeren mesajlar.

Kısa metin yeterli kelime birlikteliği sağlamadığından ve farklı metinler arasında iyi bir benzerlik ölçümü için yeterli özellikleri sağlamadığından dolayı sosyal ağlardaki spam tespiti zorluklar içermektedir. Bunların üstesinden gelmek için, sınıflandırıcının mesajları etkili bir şekilde ayırt etme yeteneğini geliştirmek amacıyla güçlü bir metin temsili kullanılmalıdır.

Buna istinaden, bu çalışmada yukarıda bahse geçen sorunu çözmek için doğal dil işleme tabanlı modellerin kullanılması önerilmiştir. Bilindiği gibi farklı bağlamlarda kelime anlamlarını ve benzerlikleri yakalama yeteneğini geliştirerek, kelimelerin anlamlı bir temsilini oluşturulması doğal dil işlemenin (NLP) ana hedeflerinden biridir. Bu amaç ile, kelime çantası (Bag of Words) modeline dayalı geleneksel yöntemlerden başlayarak word2vec (Mikolov, Chen Vd. 2013), global vektör (GloVE) (Pennington, Socher Vd. 2014) ve fastText (Bojanowski, Grave Vd. 2017) gibi geleneksel kelime temsili (Word embedding) yöntemleri gibi birçok yaklaşım tanıtılmıştır.

Son zamanlarda BERT gibi Transformatör (transformer) tabanlı yöntemler geliştirilmiştir. Geleneksel kelime temsil yöntemleri, normal uzunluktaki metinlere başarıyla uygulanmıştır, ancak kısa metinlerle, özellikle de sosyal ağ platformlarından

çıkarılan metinlerle ilgilenirken mesele farklılık göstermektedir. Ayrıca, geleneksel kelime temsil modelleri bağlamdan bağımsız modellerdir, yani anlamı ve cümle içindeki sırası ne olursa olsun herhangi bir kelimeyi tek bir temsil vektörüne dönüştürmektedirler. Ayrıca, kelime temsil modelleri "kelime dağarcığı" (Out of vocabulary-OOV) problem ile karşı karşıya kalmaktadır, yani kelimelerin temsil edilebilmesi için bu kelimelerin, eğitim veri setinde bulunması gerekmektedir. Dolayısıyla, kısa metnin seyrekliği doğasıyla başa çıkmak için kelimelerin güçlü bir temsiline ihtiyaç vardır. Bağlamsal temsilde (Contextual embeddings) her bir kelime için oluşturulan desen cümledeki diğer kelimelere bağlı olduğundan, bu tip temsiller, kısa metinlerdeki kelimeleri temsil etmek için ideal bir seçimdir.

Bağlamsal kelime temsilleri (CoVe) (McCann, Bradbury Vd. 2017), bir kodlayıcının eğitmesine ve daha iyi bir kelime temsilinden yararlanılabilecek şekilde bu kodlayıcının başka bir göreve dönüştürmesine odaklanan kelime temsillerinin gelişmiş bir versiyonudur. Bu model aynı zamanda bilinmeyen kelimeleri temsil etmek için sıfır değerler atadığı OOV probleminden muzdariptir.

Öte yandan, Konu (Topic) modeli, bir belge koleksiyonunda ortaya çıkan soyut "konuları" keşfetmek için bir tür istatistiksel modeldir. Konu modellemenin en yaygın biçimi Latent Dirichlet Allocation (LDA) modelidir, bu model tweetler gibi kısa metinlerle başa çıkamama gibi bazı sorunlarla da karşı karşıya kalmaktadır.

Bu araştırmada önerilen modellerin eğitilmesi ve test edilmesi için Twitter, YouTube ve SMS veri kümeleri olmak üzere üç kıyaslama veri kümesi kullanılmıştır.

Twitter, akademisyenlerin yanı sıra öğrenciler, politikacılar ve genel halk arasında giderek daha popüler hale gelmektedir. Twitter, sosyal servisler içerisinde en çok kullanılan platformlardan bir tanesidir. Twitter, dokuz popüler sosyal medya ağı listesinde yedinci sırada yer almaktadır. 2020'nin dördüncü çeyreğinden elde ettiği son rakamlara göre, Twitter platformu, günlük 192 milyon aktif kullanıcıya sahiptir. Twitter, Temmuz 2006'da halka açılan gerçek zamanlı bir mikroblog platformudur. Kayıtlı kullanıcıların Tweet adı verilen kısa gönderiler göndermesine ve almasına izin vermektedir. Tweet, Twitter'a gönderilen metin, fotoğraf, ve/veya video içeren bir mesajdır. Başlangıçta, kullanıcılar 140 karakterlik bir Tweet gönderebilir iken 2017'de

280 karakter uzunluğa kadar (iki kata) çıkmıştır. Tweet birçok formatta olabilmektedir;

1) **Tweet**: Gönderenin profil sayfasında, gönderenin ana zaman çizelgesinde ve gönderenin takip edenlerin ana zaman çizelgesinde görünen **genel tweet'ler**.

2) **Bahsetmeler (mentions):** Başka bir Twitter hesaptaki kullanıcı adını içeren ve başında "@" sembolü bulunan Tweetlerdir. Bu tür Tweet'leri, gönderenin herkese açık profil sayfasında ve alıcının bildirimler sekmesinde görünmektedir. Ek olarak, göndereni takip ediyorlarsa, alıcının ana zaman çizelgesi görünümünde de gösterilmektedir.

3) **Yanıtlar:** Başka bir kişinin Tweetine yazılan yorumların yanıtlar denilmektedir**.** Gönderenin profil sayfasında ve alıcının Bildirimler sekmesinde görünmektedir. Yanıtlar, göndereni takip ediliyorsa alıcının ana sayfa zaman çizelgesinde de görünmektedir.

4) **Retweet:** Retweet, bir Tweetin yeniden gönderilmesidir. Bu Tweet'in tüm kullanıcı takipçileriyle hızla paylaşılmasına yardımcı olmaktadır. Bir kullanıcı, takip ettiği kişilerin Retweet'lerini ve alıntı Tweetlerini kendi zaman tünelinde görebilmektedir.

Net Daily Twitter'a göre, spam, bir platform manipülasyonu biçimi olarak tanımlanmaktadır. Platform manipülasyonu, insanların Twitter'daki deneyimlerini olumsuz etkilemeyi amaçlayan, istenmeyen veya tekrarlanan eylemleri içeren bir faaliyettir. Ayrıca, spam, kötü niyetli otomasyonu ve sahte hesaplar gibi diğer platform manipülasyon eylemlerini de içerebilmektedir.

Şekil 1.2 Twitter kullanıcı artışı

1.1. Sosyal Medya Analizi

Sosyal medya siteleri, uzun yıllardır aktif bir şekilde kullanılmaktadır. İlk sosyal medya kanalı, 1997'de kurulmuş olup Six Degrees olarak adlandırılmıştır. Bahse geçen sosyal medya kanalı, kullanıcıların mesaj ve bülten parçaları göndermesine izin vermekteydi. O zamandan beri çok şey değişmiş olup günümüzde Twitter, Instagram veya Facebook gibi birden çok sosyal medya platformu geliştirilmiş olup günlük hayatımızın vazgeçilmez bir parçası haline gelmiştir.

Sosyal medya analizi, Facebook, Instagram, LinkedIn ve Twitter gibi sosyal ağlardan veri toplama, analiz etme ve eyleme geçirilebilir sonuçlar çıkarma sürecidir. Sosyal medyayı analiz etme sürecin üç ana adımı vardır: veri tanımlama, veri analizi ve bilgi yorumlama.

Anlamlı bir mesaj taşıyan herhangi bir veri, bilgi olarak adlandırılmaktadır. İşlenmemiş veriler, bilgi mesajına dönüştürülürken Şekil 1.3'de yer alan ve aşağıdaki maddelerde gösterilen biçimleri almaktadır:

1) **Ham veri**: başlangıçta, işlenecek veri hem anlamlı bilgiyi hem de gürültüyü içeren ham veri biçiminde olmaktadır,
2) **Filtrelenmiş veri**: veriyi ön işledikten sonra yalnızca ilgili verileri içeren filtrelenmiş veriler hâle gelmektedir,
3) **Enformasyon (Mâlumât)**: filtrelenmiş verileri işlendiğinde enformasyon hâle gelmektedir,
4) **Bilgi**: kesin bir mesaj ileten verileri bilgi olarak bilinmektedir,
5) **Bilgelik**: kesin mesajı ve arkasındaki nedeni ileten verileri bilgelik olarak bilinmektedir.

İşlenmemiş verilerden bilgelik elde etmek için işleme tabi tutulmalıdır. Buna istinaden odaklanmak istediğimiz verileri dâhil ederek veri kümesini iyileştirip ve bilgileri tanımlamak için verilerin düzenlenmesi gerekmektedir.

Veri analizi, ham verileri, anlamlı formata dönüştürmeye yardımcı olan faaliyet dizisidir. Başka bir deyişle, veri analizi, filtrelenmiş verileri girdi olarak alan ve bunu analistler için değerli bilgiye dönüştüren işlemler bütünü olarak tanımlanmaktadır.

Sosyal medya verileri, gönderilerin, duyguların, duygu faktörlerinin, coğrafyanın, demografinin analizi gibi birçok farklı analiz türü gerçekleştirilebilmek amacıyla kullanılabilmektedir.

Şekil 1.3 Veri biçimler arasında dönüştürme adımları

1.2. Araştırmanın Önemi ve Katkıları

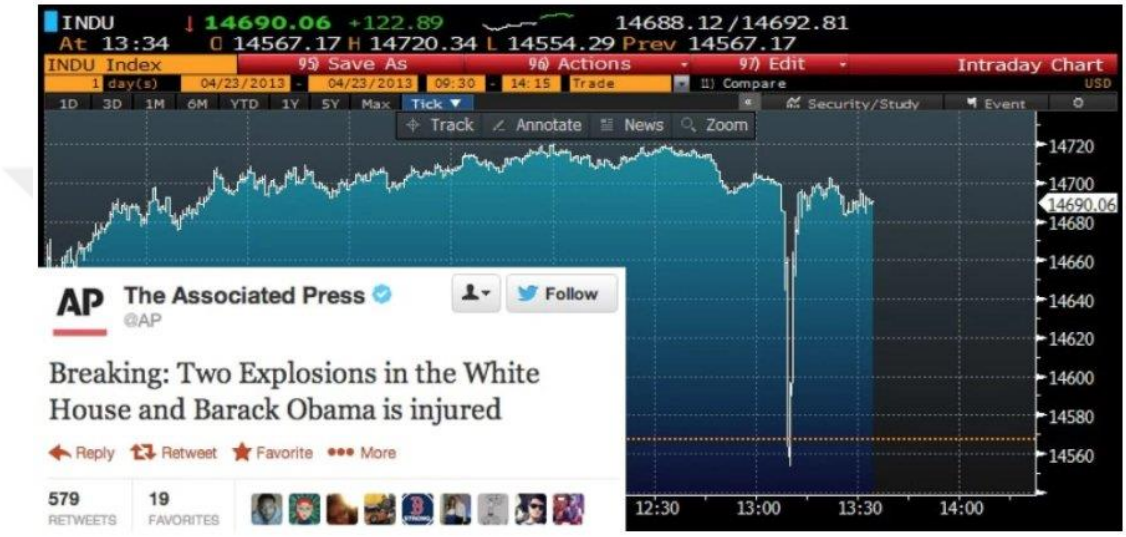
Daha önce de belirttiğimiz gibi, sosyal ağların artan popülaritesi, spam göndericiler için önemli bir hedef hale getirmiştir.

Sosyal medya hedef alan spam, sadece kullanıcıların deneyimi üzerindeki olumsuz etkisi veya cihazlarına zarar vermesi ile sınırlı olmayıp, aynı zamanda çok büyük ekonomik ve sosyal zararlara da yol açabilmektedir.

Örneğin, 2013 yılında Amerika Birleşik Devletleri'nde meydana gelen ve Associated Press'in Twitter hesabındaki sahte bir Tweet'in Beyaz Saray'da Başkan Obama'yı yaralayan patlamaları bildirdiği ünlü hikâyeden bahsedelim. Sahte Tweet'te "Kırılma: Beyaz Saray'da İki Patlama ve Barack Obama yaralandı" denilmiştir. Neredeyse anında, Şekil 1.4' de görüldüğü gibi ABD piyasaları bu Tweet tarafından olumsuz etkilenmiş ve keskin bir şekilde düşmüştür.

Daha sonra, mesajın sahte olduğunu netleştiğinde piyasalar aynı hızla toparlanmıştır. Ancak önlemler anlık kaosu önleyecek kadar hızlı alınamamıştır. Örneğin, Washington Post 'a göre, Dow Jones endüstriyel ortalaması 13:08 ve 13:10 saatleri arasında 100 puandan fazla düşmüştür.

Bu olaydan da anlaşılacağı gibi, sosyal medyadaki mesajların gerçek dünya üzerindeki çok etkili olup sosyal medya platformları hedef alan spam ataklardan kaçınmak için gerçek zamanlı çalışan etkili yöntemlerin önerip geliştirilmesi ciddi bir ihtiyaç haline geldiği aşikârdır.



Şekil 1.4 Sahte tweetin piyasalara etkisi

Bu çalışmanın ana katkıları şu şekilde özetlenebilmektedir:

- Çalışma kapsamında, sosyal ağlardaki spam mesaj sorununu çözmek için derin öğrenme ve NLP tekniklerine dayanan dört farklı çalışmayı gerçekleştirilmiştir. Çalışmalarda önerilen modellerin sonuçları, spam tespiti alanında daha önce önerilen son teknoloji yöntemlerdeki sonuçlarıyla karşılaştırılmıştır. Gerçekleştirilen çalışmalarda önerilen modelleri, üç kıyaslama veri seti ile test edilmiştir, bunlar: Twitter veri seti, YouTube yorum veri seti ve SMS spam veri setidir.
- Birinci çalışmada (Ghanem ve Erbay 2022), çift yönlü tekrarlayan sinir ağına dayalı yeni bir derin öğrenme mimarisi önerilmiştir. Önerilen mimaride, bağlamsal metin temsili sağlamak için gömme katmanında ELMO modeli kullanılmıştır. Bu çalışmada önerilen model, CBLSTM (Contextualized Bi-directional Long Short Term Memory neural network) olarak adlandırılmıştır. Önerilen modelde kullanılan bilişenlerin etkisi test edip her bilişenin en iyi

sonuçları verebilecek değeri bulmak için bir analiz çalışması gerçekleştirilmiştir. Ayrıca, önerilen CBLSTM modelin etkili olduğunu ispatlamak için geleneksel özellik ağırlıklandırma yöntemlerine ek olarak GloVe ve fastText gibi geleneksel kelime gömme yöntemleri, CBLSTM modelinde yer alan ELMO yerinde kullanılarak bir karşılaştırma yapılmıştır.

- İkinci çalışmada (Ghanem ve Erbay 2020), farklı kelime temsil yöntemleri benimsenerek derin öğrenme tabanlı birden fazla model uygulanmıştır. Ardından, Twitter'daki spam mesajlar algılamak için BERT modeline dayalı yeni bir mimari önerilmiştir. Ayrıca, geleneksel ağırlıklandırma yöntemlerini, geleneksel kelime gömme dayalı yöntemleri ve bazı son teknoloji yöntemleri, performans açıdan önerilen yöntem ile karşılaştırılmıştır.
- Üçüncü çalışmada ise, transformer tabanlı yöntemler ve bunların sosyal ağlarda spam tespiti alanındaki uygulamaları hakkında karşılaştırmalı bir çalışma yapılmıştır. Ardından sosyal ağlarda spam tespiti için derin bir yığın çift yönlü sinir ağı mimarisi önerilmiştir. Bundan sonra, Twitter'da spam tespit etmek için yeni bir melezleştirilmiş derin sinir ağ modeli önerilmiştir. Önerilen modelde daha fazla bağlam sağlamak için yığılmış çift yönlü uzun kısa vadeli sinir ağı mimarisinin yardımıyla RoBERTa önceden eğitilmiş modeli kullanılmıştır.
- Son gerçekleştirilen çalışmada ise, Sosyal ağlarda spam tespiti için ALBERT tabanlı bir derin öğrenme olan ve ALBERT4Spam olarak adlandırılan modeli tanıtılmaktadır. Çalışmada önerilen RNN tabanlı derin sinir ağ mimarisinin performansını iyileştirmek için, ALBERT iyi bilinen model, kelime temsillerini öğrenmek amacıyla kullanılmıştır. Önerilen ALBERT4Spam modelinde kullanılan nöron sayısı, katman sayısı, aktivasyon fonksiyonu, ağırlık başlatıcı, öğrenme oranı, optimize edici ve bırakma gibi birden çok hiper parametreler, modelin en iyi performansına ulaşmak için rastgele arama yöntemi kullanılarak ince ayar (Hyperparameter fine-tuning) yapılmıştır.

2. LİTERATÜR TARAMASI

Günümüzde, sosyal medyada vakit geçirmek en popüler çevrimiçi aktivitelerden biridir. 2020'deki Statista raporuna göre, dünya çapında yaklaşık 3.6 milyar insan sosyal medya kullanılmaktadır ve bu sayının 2025'te neredeyse 4.41 milyara çıkması beklenmektedir. Bu kadar yaygın olan bir sosyal ağın spam gönderenlerin hedefi haline gelmesi kaçınılmazdır. Spam gönderen (spammer), kullanıcı bilgilerini pazarlamak, kullanıcıyı yanıltmak veya tuzağa düşürmek amacıyla, kişisel bilgilerini çalmak için kötü niyetli bağlantılara tıklamaya zorlayarak veya cihazında kötü amaçlı içerik yaymak amacıyla spam adı verilen istenmeyen mesajları yaymaya çalışan bir kişi veya gruptur.

Sosyal ağlardaki spam sorunu son on yılda zorlu bir sorun hale gelmiştir. Dolayısıyla, bilimsel araştırmacılar, son yıllarda spam sorununu çözmek için büyük çaba sarf etmektedir. İlk olarak e-postalardaki spam sorununu çözmek için kullanılan kara listeye alma ve içerik tabanlı algılama gibi klasik yöntemleri kullanmayı denemişler, ancak kısa ve gürültülü mesajlarına sahip sosyal medya yapılardaki spam tespiti, e-posta spam sorununa göre daha zorlayıcı olduğunu ortaya çıkmıştır. Araştırmacıların çoğu, Yang Vd.(Yang, Harkreader Vd. 2013) gibi spam'ı kötü niyetli bağlantılar içeren mesajlar olarak ele almıştır. Diğerleri ise, Chen vd. gibi (Chen, Yeo Vd. 2017) reklamları, otomatik olarak oluşturulan içeriği ve düşük kaliteli içerikleri dâhil ederek spam tanımını genelleştirmiştir.

Araştırmacılar, Twitter'da hem spam gönderenleri hem de spam mesajlarını tespit etmek için çeşitli yöntemler geliştirmiştir. Örneğin, Mateen vd. (Mateen, Iqbal Vd. 2017), Inuwa-Dutse vd.(Inuwa-Dutse), Liptrott Vd.(Liptrott Vd. 2018), Concone vd.(Concone, Re Vd. 2019), Karakaşlı vd.(Karakaşlı, Aydin Vd. 2019), Adewole vd.(Adewole, Han Vd. 2020), Alom vd.(Alom, Carminati Vd. 2020), Raj vd.(Raj, Srinivasulu Vd. 2020) tarafından yapılan çalışmalarda, spam gönderenleri tespit etmek için çalışılırken, Chen vd. (Chen, Wang Vd. 2016), Sedhai ve Sun (Sedhai ve Sun 2017), Wu vd.(Wu, Liu Vd. 2017), Madisetty ve Desarkar (Madisetty ve Desarkar 2018), Alsaffar vd.(Alsaffar, Alfahhad Vd. 2019), Imam vd.(Imam, Issac Vd. 2019), Jain vd.(Jain, Sharma Vd. 2019), Kabakus ve Kara(Kabakus ve Kara 2019), Kumar vd.(Kumar, Singh Vd. 2019), Tajalizadeh ve Boostani (Tajalizadeh ve Boostani 2019),

Wang (Wang, Kang Vd. 2019), Sun vd.(Sun, Lin Vd. 2020) tarafından yapılan çalışmalarda spam mesajlarını tespit etmek için çalışılmıştır.

Literatürde spam tespiti için kullanılan yöntemler üç ana yaklaşıma dayanmaktadır: kara liste (blacklist), sözdizimi analizi (syntax analysis) ve özellik analizi (feature analysis).

Kara liste en basit tespit yöntemidir. Bu yöntem kısaca, kara liste ile karşılaştırarak Tweet'lere gömülü kötü niyetli URL'leri tespit edilmektedir.

Sözdizimi analiz yöntemlerinde, özellikler temel olarak Tweetlere gömülü anahtar parçalardan çıkarılmaktadır. Öte yandan, özellik analizi yöntemlerinde, özellikler genellikle tweet, Twitter hesaplarının istatistikleri veya sosyal grafiklerin topolojik bilgilerinden toplanmaktadır (Wu, Wen Vd. 2018).

Bu çalışmada, Twitter'daki spam tweet'leri tespit etmek için tweet'lerin içeriğine odaklanılmıştır. Literatürdeki incelenen son çalışmalar, tweet metninden özellikler çıkarmak için çeşitli geleneksel ve modern temsil yöntemlerine dayanmaktadır, ardından sınıflandırma görevi makine öğrenme algoritmaları veya sinir ağları kullanarak gerçekleştirilmiştir.

Araştırmacılar, içerik tabanlı özellikler, hesap tabanlı özellikler veya bunların bir kombinasyonu dâhil olmak üzere birçok özellik üzerinden spam mesajı ayırt etmeye çalışmışlardır.

Yapay zekâ ve makine öğrenimi tekniklerinin gelişmesiyle birlikte, son yıllarda, Zheng Vd.(Zheng, Zeng Vd. 2015), Watcharenwong ve Saikaew (Watcharenwong ve Saikaew 2017), Kandasamy vd. (Kandasamy ve Koroth 2014), Mateen vd.(Mateen, Iqbal vd. 2017), Uysal (Uysal 2018), Tur vd. (Tur ve Homsi 2017), Gupta vd. (Gupta, Jamal Vd. 2018), Samsudin vd. (Samsudin, Mohd Foozy Vd. 2019) gibi birçok bilimsel araştırmacı daha iyi sonuçlar elde etmek için önerilen yöntemlerine makine öğrenmesi algoritmaları entegre etmeye çalışmışlardır. Detaylı bir örnek verecek olursak, Zheng vd. (Zheng, Zeng vd. 2015) çalışmalarında, Sina Weibo platformundaki spam hesaplarını tespit etmek için destek vektör makinesi (SVM) sınıflandırıcı olarak kullanılmıştır. Benzer şekilde Kandasamy vd. (Kandasamy ve

Koroth 2014) çalışmalarında, Twitter'daki spam tespiti için SVM sınıflandırıcısıyla birlikte Naïve Bayes sınıflandırıcı kullanmışlardır.
Ayrıca, Watcharenwong ve Saikaew. (Watcharenwong ve Saikaew 2017), kapalı Facebook gruplarında spam tespitine yönelik rastgele orman sınıflandırıcı kullanmıştır. Öte yandan, Uysal (Uysal 2018) ve Samsudin vd. (Samsudin, Mohd Foozy vd. 2019) çalışmalarında YouTube platformunda spam yorumlarını filtrelemek için Naïve Bayes, karar ağacı ve lojistik regresyon kullanılmıştır.

Daha sonra bilgisayarlı görü (computer vision), konuşma tanıma, makine çevirisi ve doğal dil işleme gibi birçok alanda derin öğrenme algoritmalarının elde ettiği dikkat çekici sonuçlar nedeniyle sosyal ağlarda çeşitli spam tespit modellerinde kullanılmaya başlamıştır. Madisetty ve Desarkar (Madisetty ve Desarkar 2018), Wu vd. (Wu, Wen Vd. 2017), Ameen ve Kaya (Ameen ve Kaya 2018), Jain vd.(Jain, Sharma Vd. 2019), Barushka ve Hajek (Barushka ve Hajek 2020) tarafından yapılan çalışmalar örnek olarak verilebilir.

Bu çalışmada, ağırlıklı olarak Twitter ve YouTube sosyal platformlarına odaklanılmıştır. Twitter ‘da spam'ı tespit etmek için iki ana eğilim izlenmiştir: (1) Spam gönderen hesapların tespiti amaçlayan çalışmalar, Yang ve arkadaşları (Yang, Harkreader vd. 2013), Kandasamy ve Koroth (Kandasamy ve Koroth 2014), Lee ve Kim (Lee ve Kim 2014), Miller vd.(Miller, Dickinson Vd. 2014), El-Mawass ve Alaboodi (El-Mawass ve Alaboodi 2015), Zheng vd.(Zheng, Zeng vd. 2015) tarafından gerçekleştirilen çalışmalar örnek olarak verilebilir, (2) Spam mesajlarının tespiti hedef alan çalışmaları, Chen vd. (Chen, Yeo Vd. 2015), Gupta vd.(Gupta, Jamal vd. 2018), Jain vd.(Jain, Sharma vd. 2019), Wu vd. (Wu, Wen vd. 2017), Ameen ve Kaya (Ameen ve Kaya 2018) tarafından gerçekleştirilen çalışmalar örnek olarak verilebilir.
Egele vd. (Egele, Stringhini vd. 2015) gibi bazı araştırmacılar, takipçilerin güvenini istismar ederek spam yaymak için kullanılabilecek, güvenliği ihlal edilmiş hesapları tespit etmekle de ilgilenmiştir.
YouTube'da spam tespiti alanında gerçekleştirilen çalışmalar, çalışmanın amacı açısından yorum spam mesajlarının tespiti veya video yanıt spam mesajının tespiti olmak üzere iki ana başlık altında ayrılabilmektedir. Alberto vd. (Alberto, Lochter Vd. 2015), Aiyar ve Shetty (Aiyar ve Shetty 2018), Hidalgo ve Zurutuza (Hidalgo ve

Zurutuza 2018), Uysal (Uysal 2018), Samsudin vd. (Samsudin, Mohd Foozy vd. 2019) tarafından yapılan çalışmalar yorum spam mesajlarının tespiti hedef alan çalışmalar için örnek olarak verilebilirken, Chaudhary ve Sureka (Chaudhary ve Sureka 2013), Chowdury vd. (Chowdury, Adnan Vd. 2013), Kanodia vd. (Kanodia, Sasheendran Vd. 2018) tarafından yapılan çalışmalar video yanıt spam mesajının tespiti hedef alan çalışmalar için örnek olarak verilebilir.

Geleneksel spam tespit yöntemleri, ister metin ister profil bilgisinden olsun, temel olarak farklı özelliklerin çıkarılmasına dayanmaktadır. Örneğin, Chen vd. (Chen, Yeo vd. 2015) tarafından yapılan çalışmada, Rastgele Orman sınıflandırıcısını kullanarak spam içerik tespit etmek için taranan JSON tweetinden çıkarılan doğrudan özellikleri kullanılmıştır. Öte yandan, Gupta vd. (Gupta, Jamal vd. 2018) metin tabanlı ile birlikte tweet tabanlı ve kullanıcı tabanlı özelliklere dayalı bir platformu önermişlerdir. Detaylı olarak anlatacak olursak, Tweet metnindeki spamı tespit etmek için özellikler çıkarılmıştır, ardından sinir ağı modeli de dâhil olmak üzere bazı makine öğrenme sınıflandırıcı modelleri eğitilip test edilmiştir.

Spam tespiti problemi gibi metin sınıflandırma problemlerinde yeni benimsenen yaklaşım, metinlerin temsilinde word embedding yöntemlerini kullanmak ve elde edilen öznitelik vektörlerini makine öğrenimi tabanlı veya derin öğrenme tabanlı bir modele girdi olarak kullanmaktır. Madisetty ve Desarkar (Madisetty ve Desarkar 2018), Jain vd. (Jain, Sharma vd. 2019), Wu vd. (Wu, Wen vd. 2017) tarafından yapılan çalışmalar örnek olarak verilebilir. Mesela, Wu vd. (Wu, Wen vd. 2017) tarafından yapılan çalışmada, Twitter platformunda spam tespit etmek için word2vec modeli kullanılmıştır. Her tweet'in sözdizimi Word2Vector modeli aracılığıyla öğrenilmiştir, ardından önceki temsil veri kümesine dayalı olarak ikili sınıflandırıcı oluşturulmuştur.
Bahse geçen önceki yöntemlerin, doğal dil işleme alanında, özellikle metin sınıflandırma problemlerinde büyük başarı elde ettiğini belirtmekte fayda var. Ancak bu yöntemlerin de bazı sınırlamaları vardır.

Geleneksel word embedding modellerinin en yaygın sınırlamaları, bu modellerin bağımsız modeller olması, yani bir cümledeki kelimenin sırası ve anlamından bağımsız olarak her kelime için yalnızca bir tane sabit vektör üretmektedir. Bu problem üstesinden gelmeye yönelik Jain vd. (Jain, Sharma vd. 2019), ve Wang vd.

(Wang, Wang Vd. 2017) gibi bazı araştırmacılar, daha fazla anlamsal bilgi eklemek için ek bilgi tabanlarını derin öğrenme mimarisiyle birleştirerek sınıflandırma görevlerinin performansını iyileştirmek için çalışmışlardır. Örneğin, Wang vd. (Wang, Wang vd. 2017) çalışmasında, kısa metin sınıflandırma sorununu ele almak için ilgili kavramları almak ve bunları kısa metinlerle ilişkilendirmek için Probase bilgi tabanı kullanmışlardır. Aynı fikir, Jain vd.(Jain, Sharma vd. 2019) çalışmasında sosyal ağlardaki spam tespiti alanında da uygulanmıştır. Çalışmada önerilen model, WordNet ve ConceptNet gibi bilgi tabanları yardımıyla kelimelerin temsilinde semantik bilgilerin tanıtılmasıyla desteklenmiştir.

Geleneksel word embedding modelleriyle ilgili bir diğer sınırlama ise, özellikle kısa metin sınıflandırma problemleri üzerinde çalışırken karşılaşılan OOV problemidir.

Önceki problemlerin üstesinden gelmek için, bu tez çalışmalarında farklı bağlamsal temsil tabanlı yaklaşımlar önerilmiştir. Örneğin, tezdeki ikinci çalışmada önerilen modeli önceki çalışmalarda (Jain, Sharma vd. 2019) bulunan modellerden ayıran en önemli fark, kelime gösterimi için ELMO modeli kullanıldığında metne anlam katmak için zaman alan ve algoritmanın karmaşıklığını artıran ek bilgi tabanlarına ihtiyaç duyulmamasıdır. Aynı zamanda modelin karakter tabanlı olduğu ve eğitim aşamasında daha önce görmemiş herhangi bir metni işleyebileceği için OOV problemi de ele alabilmektedir, bu da diğer geleneksel kelime gömme tabanlı yöntemlerden ayıran en önemli özelliklerden bir tanesidir. Ve son olarak, önerilen modelin, kelime temsili için kullanılan ELMO özelliklerinden dolayı herhangi bir metni kabul ettiği belirtmekte fayda vardır.

3. MATERYALAR VE YÖNTEMLER

Bu bölümde, bu araştırmada kullanılan metodolojilerden bahsedilmektedir. Bölüm iki ana kısımdan oluşmaktadır. Öncelikle bu çalışmada ihtiyaç duyulan doğal dil işleme teknikleri açıklanmaktadır. Ardından simülasyon aşamasında kullanılan makine öğrenmesi, derin öğrenme algoritmaları ve önerilen modellerinden bahsedilmektedir.

3.1. Doğal Dil İşleme

Doğal Dil İşleme (NLP), doğal dil kullanarak bilgisayarlar ve insanlar arasındaki etkileşimle ilgilenen bir yapay zekâ dalıdır. Makinelerin insan dilini işlemesine ve anlamasına yardımcı olmaktadır, böylece tekrarlayan görevleri otomatik olarak gerçekleştirebilmektedir. Doğal dil işlemenin önemi, büyük miktarda metin verisini insanlardan daha hızlı ve daha doğru bir şekilde işleme yeteneğinden kaynaklanmaktadır. Ancak, öncelikle makinelerin doğal dili anlayabilmesi için yorumlanabilir başka bir forma dönüştürülmesi gerekmektedir. Bu amaçla, bilgisayarların sözlü veya yazılı metni insanlarla aynı şekilde okuyup anlamasına yardımcı olmak için metinler parçalara ayrılmalıdır, böylece cümlelerin dilbilgisi yapısı ve kelimelerin anlamı bağlam içinde analiz edilebilip anlaşılabilmektedir. Bahsedilen bu işleme ön işleme denmektedir. Burada, doğal dillerin işlenmesinde kullanılan en popüler ön işleme tekniklerinden bahsedilmektedir.

3.1.1. Doğal Dil İşlemede Ön İşlemler

- **Belirteçleştirme (Tokenization):**

Belirteçleştirme, Metni daha küçük anlamsal birimlere veya tek tümcelere ayırmak işlemidir.

- **Küçük harfe çevirme (lower casing):**

Boyut azaltma amacıyla bir kelimeyi küçük harfe dönüştürmek lazımdır. Bu işlem önemlidir, çünkü aynı kelime küçük ve büyük harflerle göründüğünde, vektör uzay modelinde iki farklı kelimeye dönüştürülerek daha büyük boyut elde edilmektedir**.**

- **Gereksiz sözcüklerinin (stop words) kaldırılması**:

Belgelerde çok az benzersiz bilgi ekleyen veya hiç benzersiz bilgi içermeyen çok yaygın olarak kullanılan sözcükleri filtrelenmektedir, örneğin İngilizcede a, an, the, vb. gibi edatlar. Bu kelimeler, iki belgeyi ayırt etmeye yardımcı olmadığı için gerçekten herhangi bir önem ifade etmemektedir.

- **Köklendirme (Stemming):**

Bir sözcüğü, son ek ve öneklerin eklendiği sözcük köküne veya stem olarak bilinen sözcüklerin köklerine indirgeme işlemidir.

- **Kök çözümleme (Lemmatization):**

Köklendirmeden farklı olarak, kök çözümleme kelimeleri dilde var olan bir kelimeye indirger. Hem köklendirme hem de kök çözümleme, kelimeleri kök biçimlerine indirgeyerek standartlaştırmaktadır.

Kök kelimeye köklendirme işleminde **stem**, kök çözümleme işlemine ise **lemma** adı verilmektedir. Köklendirme ve kök çözümleme arasındaki temel fark, iki işlemde kullanılan algoritmalarda yatmaktadır; burada köklendirme, kelimenin ait olduğu dil anlamı hakkında herhangi bir bilgi olmadan kelimenin köküne ulaşmak için kelimenin bir kısmının kuyruk ucundan kesilmesidir. Öte yandan, kök çözümleme, algoritmalar bu bilgiye sahiptir ve kelimenin anlamını köküne indirgemeden önce anlamak için bir sözlük kullanabilmektedir.

- **Konuşmanın bir Kısmını/Parçasını Etiketleme (Part-of-Speech-Tagging**):

Komuşmanın bir parçasını etiketleme (PoS, POST), isimler, fiiller, sıfatlar, zarflar, zamirler gibi hem tanımına hem de bağlamına dayalı olarak, bir metindeki bir kelimeyi konuşmanın belirli bir bölümüne karşılık gelecek şekilde işaretleme işlemidir.
Ön işleme adımlarından sonra, kelimelerin temsili, modellerin metni daha iyi anlamasına ve kategorize etmesine yardımcı olmak için önemli bir süreç haline gelmektedir. Bir dili anlamak için kelimelerin anlamlarını anlamak çok önemlidir. Anlamsal ve sözdizimsel özelliklerin etiketlenmemiş metin verilerinden yakalandığı kelime temsili, Doğal dil işleme'de temel bir prosedürdür. Aşağıdaki alt bölüm, doğal dil işlemede ana kelime temsili yaklaşımlarını sunmaktadır.

3.1.2. Doğal Dillerin İşlenmesinde Kelime Temsili:

Kelimelerin temsili, Derin öğrenme modellerinin performansı üzerinde önemli bir etkiye sahiptir. Doğal dil işlemede kelimeleri temsil etmek için farklı yöntem ve yaklaşımlar geliştirilmiştir. En ünlü yaklaşımlar bu bölümde açıklanacaktır.

3.1.2.1. Sözlük Arama

Sözlüğe bakmak, kelimeleri temsil etmenin en basit yoludur. Bu yöntemde, kelimelerin, cümlelerin veya metinlerin bir koleksiyonu olan tümce alınmaktadır ve amaçlanan bir formatta ön işleme tabi tutulmaktadır, daha sonra önceden işlenmiş kelimelerin kelime hazinesi bir dosyaya kaydedilmektedir. Bundan sonra, kelimeler ve kimlikler arasında bir eşleme oluşturularak arama sözlüğü oluşturulmaktadır, yani sözlükteki her benzersiz kelimeye bir kimlik atanır. Sonuç olarak, Çizelge 3.1'de gösterildiği gibi, kelime kimliklerinin aranabileceği basit bir arama sözlüğü oluşturulmaktadır. Ardından, verilen her bir kelime için sözlükte aranarak karşılık gelen tamsayı gösterimi döndürülür. Sözcük,

Sözlükte mevcut değilse, kelime dağarcığı (Out of Vocabulary) belirtecine karşılık gelen tam sayı döndürülmelidir.

Çizelge 3.1 Sözlük Arama örneği

ID	Kelime
0	Car
1	Plane
2	airport

3.1.2.2. Tek Sıcak Kodlama (One Hot Encoding)

Tek Sıcak kodlama, kategorik değişkenlerin ikili vektörler olarak bir temsilidir. Bu yaklaşımda, biri hariç dolu sıfırları olan bir kelime boyutu vektörü oluşturulmaktadır. Tek bir kelime için sadece ilgili sütun bir değeri ile doldurulmaktadır ve geri kalanı sıfır değeri ile doldurulmaktadır. Kodlanmış jetonlar, 1 × (N+1) boyutuna sahip bir vektörden oluşacaktır, burada N, sözlüğün boyutudur ve ekstra bir bileşen OOV jetonu için N'ye eklenmektedir. Çizelge 3.2'de görüldüğü gibi, yalnızca ilgili sütun kelimeye her kelime için etkinleştirilmektedir.

Çizelge 3.2 Tek Sıcak kodlama örneği

	car	plane	airport	Unknown
Car	1	0	0	0
Airport	0	0	1	0
star	0	0	0	1

Bu yöntemin ana dezavantajları şu şekilde özetlenebilmektedir:

- **Boyutluluk Laneti (The curse of dimensionality)**

Boyutluluk laneti, yüksek boyutlu verilerle çalışırken hesaplamalarda ortaya çıkan ve büyük bellek gerektiren bir dizi sorunu ifade etmektedir. Bu nedenle, tek sıcak kodlama temsili, hesaplama için büyük bellek gerektirmektedir.

- **Bağlamdan Bağımsızlık**

Ortaya çıkan vektör kümesinin birbirleri hakkında fazla bir şey söylemediği bağlamdan bağımsız olması, bu yöntemde hiçbir bağlamsal veya anlamsal bilginin gömülü olmadığı anlamına gelmektedir.

Yukarıda bahsedilen iki konuyu ele almak için, boyutsallığı azaltmaya ve bağlamsal benzerlik hakkındaki bilgileri korumaya yardımcı olmak için farklı yollar tanıtılmıştır.

3.1.2.3. Kelime Çantası Modeli (Bag of Words/BoW)

Bir kelime Çantası, bir belge içindeki kelimelerin oluşumunu açıklayan metnin basitleştirici bir temsilidir. Bu modelde, bir metin, dilbilgisi ve hatta kelime düzeni göz ardı edilerek, ancak çokluğu korunarak, kelimelerinin torbası olarak temsil edilmektedir.

Çanta kelimelerin nasıl çalıştığını anlamak için aşağıdaki örneği ele alalım:

1) Ahmad likes to read books. Suzi likes to read books too.
2) Ahmad likes to watch movies too.

Bu iki cümle, bir kelime koleksiyonuyla da temsil edilebilmektedir.

1) {‘Ahmad’, ‘likes’, ‘to’, ‘read’ ,’books.’, ‘Suzi’, ‘likes’ ,’to’ ,’read’, ‘books’, ‘too.’}.
2) {‘Ahmad’,‘likes’,‘to’,‘watch’,‘movies’,‘too.’}

Daha sonra her cümle için, kelimenin birden çok tekrarı çıkarılmakta ve kelimeleri temsil etmek için kelime sayısı kullanılmaktadır.

1) {“Ahmad”:1, “likes”:2, “to”:2, “read”:2 ,” books.”:2’, “Suzi”:1}
2) {“Ahmad”:1, “likes”:1, “to”:1, “watch”:1, “movies”:1, “too.”:1}

BOW modelinin ana sınırlaması, büyük bir belge ile uğraşırken, vektör boyutunun çok fazla hesaplama ve zaman ile sonuçlanan çok büyük olabilmesidir. Ayrıca, yaklaşım kelimenin anlamını dikkate almamakta ve cümledeki kelimenin bağlamını görmezden gelmektedir.

1. Terim Frekansı-Ters Doküman Frekansı (Term Frequency-Inverse Document Frequency /TF-IDF)

a) Terim Frekansı (TF) : d belgesinde t teriminin ne sıklıkta göründüğünün bir ölçüsüdür

$$TF\left(t,d\right)=\frac{n\left(t,d\right)}{belgedeki\ terim\ sayısı} \tag{3.1}$$

Burada n, "t" teriminin "d" dökümanında görünme sayısıdır.

b) Ters Döküman Frekansı (IDF), bir terimin ne kadar önemli olduğunun bir ölçüsüdür. IDF değerine ihtiyacımız var çünkü sadece TF'yi hesaplamak kelimelerin önemini anlamak için yeterli değildir. IDF değeri aşağıdakı denklemden hesaplanmaktadır:

$$IDF\left(t\right)=log\frac{döküman\ sayısı}{t\ terimli\ döküman\ sayısı} \tag{3.2}$$

D döküman kümesinden d belgesindeki t kelimesi için TF-IDF skoru şu şekilde hesaplanmaktadır:

$$TF\text{-}IDF\left(t,d,D\right)=TF\left(t,d\right)\times IDF\left(t,D\right) \tag{3.3}$$

3.1.2.4. Dağıtılmış Kelime Temsilleri

Dağıtılmış kelime temsilinin ana fikri, kelimeleri bir özellik vektörü olarak temsil etmektir. Dağıtılmış kelime temsili, anlamsal veya sözdizimsel bağımlılıkları ortaya çıkarabilmektedir. Benzer anlama sahip kelimelerin benzer bir temsile sahip olmasına izin veren kelimelerin kullanımına dayalı olarak öğrenilmektedir.

a. Word2vec

Word2vec, 2013 yılında Google'da Tomas Mikolov ve diğerleri (Mikolov, Chen vd. 2013) tarafından bir metin yığınını gömme bağımsız bir kelimeyi verimli bir şekilde öğrenmek için geliştirilen istatistiksel bir yöntemdir.

Word2vec'te iki farklı öğrenme modeli tanıtılmıştır, bunlar:

- Continuous Bag-of-Words (CBOW) modeli.
- Skip-Gram Modeli.

- **Continuous Bag-of-Words modeli**

CBOW modeli, mevcut kelimeyi bağlamına göre tahmin ederek yerleştirmeyi öğrenmektedir. CBOW modelindeki araçlar, ortadaki kelimeyi tahmin etmek için bağlamın (veya çevreleyen kelimelerin) dağıtılmış temsilleri birleştirmektedir. Model, cümleyi (bağlam sözcüğü, hedef sözcük) biçimindeki sözcük çiftlerine dönüştürmektedir. Kullanıcının pencere boyutunu ayarlaması gerekmektedir. Örneğin, “Ahmad likes to read books” cümlesinde, bağlam kelimesi için pencere boyutu 2 ise, kelime çiftleri şöyle görünmektedir: ([Ahmad, to], likes), ([likes, read], to) ,([to, books], read). Bu kelime çiftleriyle model, bağlam kelimeleri olarak kabul edilen hedef kelimeyi tahmin etmeye çalışmaktadır.

- **Skip-Gram Modeli**

Skip-gram modeli, geçerli bir kelime verilen çevreleyen kelimeleri tahmin ederek gömmeyi öğrenmektedir. Skip-gram modelinde, bağlamı tahmin etmek için girdi kelimesinin dağıtılmış temsili kullanılmaktadır. Model, tipik olarak bağlamları ve hedefleri tersine çevirmektedir ve her bağlam kelimesini hedef kelimesinden tahmin etmeye çalışmaktadır. Başka bir deyişle model, hedef sözcüğe dayalı olarak bağlam penceresi sözcüklerini tahmin etmeye çalışmaktadır. Örneğin, "Ahmad likes to read books” cümlesinde, önceki örnekte aynı pencere boyutuyla görev, verilen hedef kelime “likes” veya [likes, read] verilen hedef kelime olan “to” bağlamını [Ahmad,to] tahmin etmek olmaktadır.

b. GloVe

Kelime Temsili için Global Vektörler veya GloVe, kelime temsillerinin denetimsiz öğrenilmesi için yeni bir global log-bilinear regresyon modelidir. Stanford üniversitesinde Pennington ve diğerleri tarafından geliştirilen, kelime vektörlerini verimli bir şekilde öğrenmek için word2vec yönteminin bir uzantısıdır (Pennington, Socher vd. 2014) .

Word2vec ve GloVe, kelimeleri vektörler (gömmeler) biçiminde temsil etmemize izin verse de word2vec yerleştirmeleri sığ bir ileri sinir ağı eğitimine dayanırken, GloVe yerleştirmeleri matris çarpanlarına ayırma tekniklerine dayanmaktadır. Ayrıca, GloVe modeli, tüm derlemden yararlanan global kelimeden kelimeye birlikte bulunma sayımlarından yararlanmaya dayanmaktadır. Öte yandan Word2vec, yerel bağlamda (komşu kelimeler) birlikte bulunmayı güçlenmektedir.

Word2vec ve GloVe'nin ana sınırlaması bağlamdan bağımsız olmasıdır, başka bir deyişle, çokanlamlılık (polysemy) dikkate alınmamaktadır, ayrıca nadir kelimeler için vektör temsilleri yeterince iyi öğrenilmemektedir.

Diğer kısıtlılık ise her kelime için gömmenin yaratıldığı kelime dağarcığı "OOV" problemidir, bu nedenle eğitimi sırasında karşılaşmadığı hiçbir kelimeyi işleyememektedir.

Skip-Gram modelinin amacı şu şekilde tanımlanmaktadır:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log p\left(\frac{w_{t+j}}{w_t}\right) \tag{3.4}$$

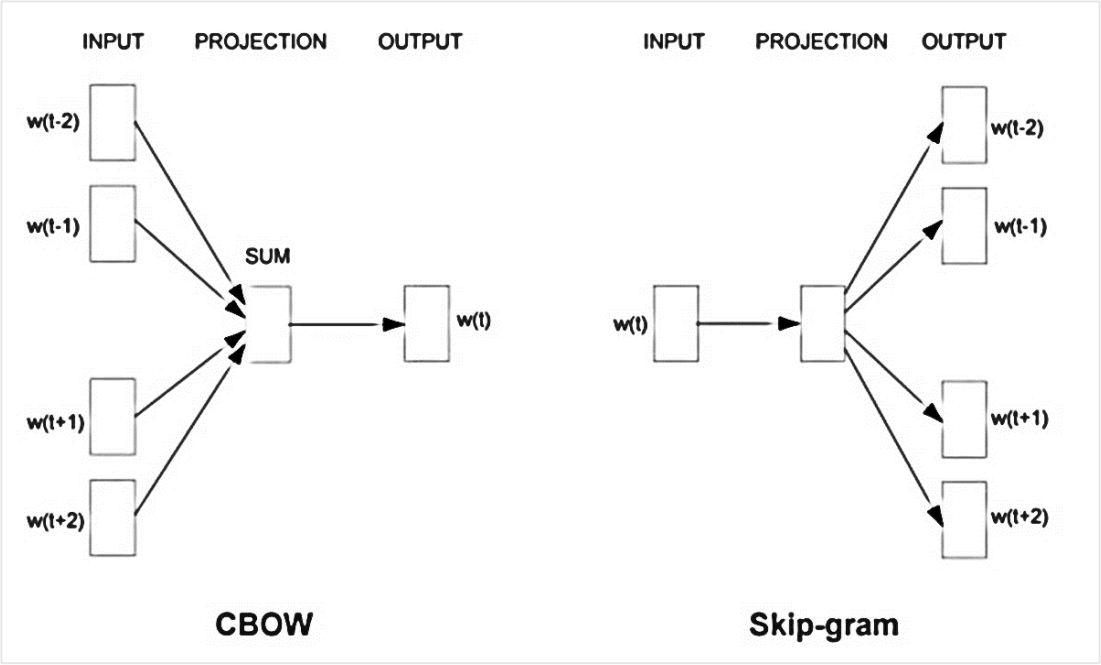
Burada w eğitim kelimesi ve c bağlamın boyutudur.

c. FastText

FastText (veya Alt Kelime modeli), önceki zorlukları çözmek için Bojanowski ve diğerleri (Bojanowski, Grave vd. 2017) tarafından önerilen atlama gram yönteminin bir uzantısıdır. FastText, her bir kelimeyi alt kelimelerinin toplamı olarak ele alarak kelimelerin morfolojik yapısını ele almaya çalışmaktadır. Alt kelimeler, kelimenin n-gram karakteri olarak alınmaktadır. Bir kelimenin vektörü, basitçe, bileşen char-n-gramlarının tüm vektörlerinin toplamı olarak alınmaktadır. Model, karakter-

gramlardan en az birinin eğitim verilerinde mevcut olması koşuluyla, bileşen karakter-gramları için vektörleri toplayarak kelime dağarcığı dışındaki sözcükler için bile vektörler elde edebilmektedir.

Modelin karakter tabanlı yapısı OOV sorununu çözse de diğer yandan yüksek bellek gereksinimi fastText modelinin en büyük dezavantajlarından biridir.



Şekil 3.1 Word2vec eğitim modelleri (Mikolov, Chen vd. 2013)

3.1.2.5. Bağlamsal Kelime Temsilleri

a. ELMO

ELMO, hem kelime kullanımının karmaşık özelliklerini (örneğin, sözdizimi ve anlambilim) hem de bu kullanımların dilsel bağlamlar arasında nasıl değiştiğini (yani çokanlamlılığı modellemek için) modelleyen derin bağlamsal bir kelime temsilidir.
ELMO, kelimeleri, eğitim verilerinde görülmeyen kelime dağarcığı belirtecini işlemeye izin veren saf karakter tabanlı bir temsilde temsil edebilmektedir.

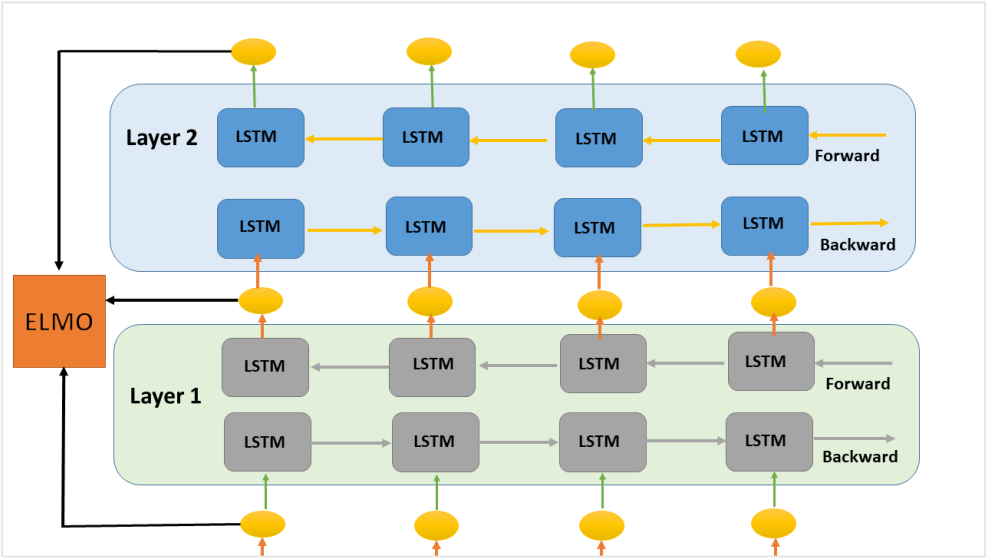
ELMO önceden eğitilmiş modellerdendir, Moses Tokenizer ile belirtilmiş olan Google 1-Billion Words veri kümesinde eğitilmiştir. ELMO modeli, cümledeki sonraki ve önceki kelimeleri anlayabilmesi için büyük bir metin bütünü üzerinde eğitilmiş çift yönlü dil modelini (biLM) kullanmaktadır. Bu biLM modelinde birlikte yığılmış iki

katman vardır. Her katmanın 2 geçişi vardır: ileri geçiş ve geri geçiş. Her simge, karakter yerleştirme kullanılarak uygun bir temsile dönüştürülür. Daha sonra karakter yerleştirme gösterimi, LSTM katmanına bir girdi sağlamadan önce bir evrişim katmanına, ardından bir maksimum havuzlama katmanına, ardından 2 katmanlı rota ağına beslenmektedir. biLm katmanının ileri geçişi, belirli bir kelime ve o kelimeden önceki bağlam hakkında bilgi içerirken, geri geçiş, kelime ve ondan sonraki bağlam hakkında bilgi içermektedir. Hem ileri hem de geri geçiş, biLM'in ikinci katmanına beslenen bir ara kelime vektörü oluşturmaktadır. Son olarak, nihai temsil, ham kelime vektörlerinin ve iki ara kelime vektörünün ağırlıklı toplamıdır. Şekil 3.2, orijinal ELMO makalesinden yeniden üretilen ELMO mimarisinin blok şemasını göstermektedir.

ELMO işlevi, girdinin k'ıncı kelimesi üzerinde aşağıdaki işlemi gerçekleştirir:

$$\boldsymbol{ELMO}_{k}^{task} = \gamma_{k}.\left(\boldsymbol{s}_{0}^{task}.\boldsymbol{xk} + \boldsymbol{s}_{1}^{task}.\boldsymbol{h}_{1,k} + \boldsymbol{s}_{2}^{task}.\boldsymbol{h}_{1,k}\right) \tag{3.5}$$

Burada si, dil modelindeki gizli temsiller üzerindeki softmax normalleştirilmiş ağırlıkları temsil eder ve γ, göreve özgü bir ölçeklendirme faktörünü temsil etmektedir. ELMO modeli, her LSTM katmanında sabit yerleştirmeler, 3 katmanın öğrenilebilir bir birleşimi ve girdinin sabit bir ortalama havuzlu vektör temsili çıktısı vermektedir.

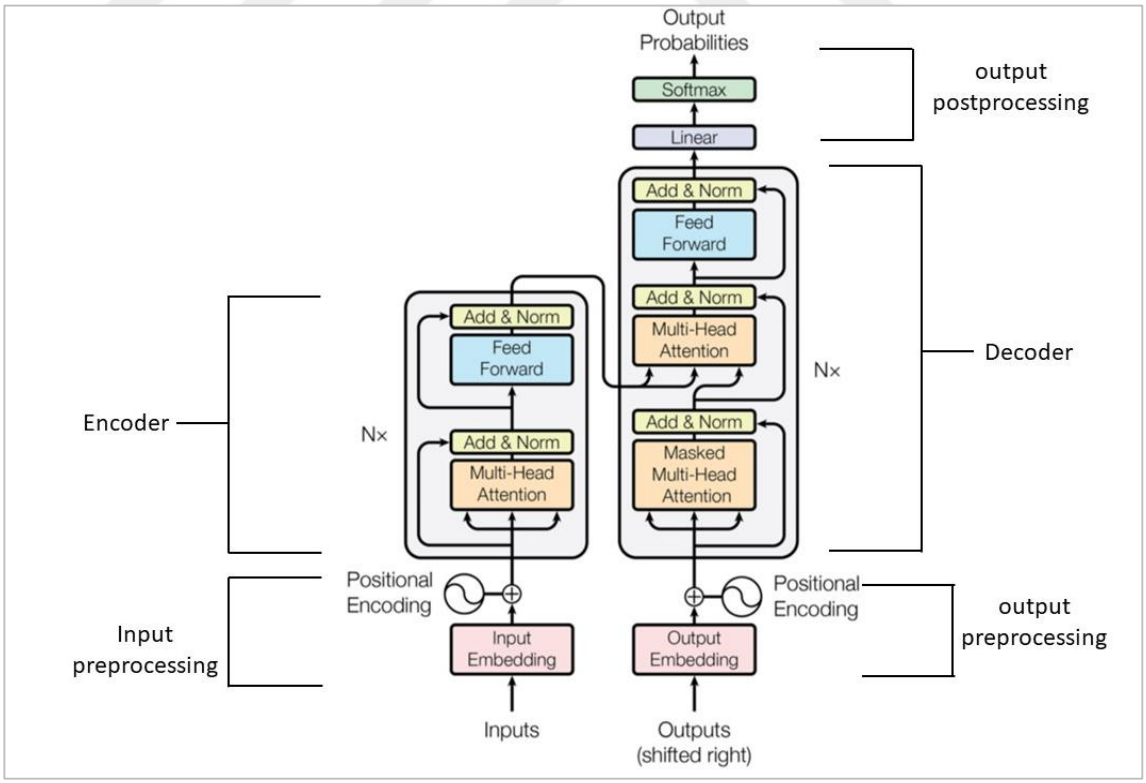


Şekil 3.2 ELMO mimarisi

b. Transformer

Transformer, uzun menzilli bağımlılıkları kolaylıkla ele alırken diziden diziye görevleri çözmeyi amaçlayan yeni bir mimaridir. Sıra hizalı RNN'ler veya evrişim kullanmadan giriş ve çıkışının temsillerini hesaplamak için tamamen self-attention üzerine kuruludur. Self-attention, dizinin bir temsilini hesaplamak için tek bir dizinin farklı pozisyonlarını ilişkilendiren bir Attention mekanizmasıdır. Transformatör, kodlayıcı ve kod çözücü olmak üzere iki ana bloktan oluşmaktadır.

Kodlayıcı bloğu, attention mekanizmasını kullanarak girdinin farklı bölümlerine seçici olarak katılarak girdi verilerini kodlar ve kodlamaları kodu çözülecek kod çözücüye iletmektedir. Kodlayıcıda önce kelimeler bir dizi sayıya dönüştürülmektedir. Bu sayı dizisi bir Self-attention katmanını beslemektedir. Sonuçlar, ileri beslemeli bir sinir ağına geçirilmekte ve bunun sonucu, daha sonra kod çözücüye iletilebilen bağlama duyarlı kodlanmış bilgi olarak kullanılmaktadır. Kod çözücü bloğu, kod çözücünün giriş dizisinin önemli bölümlerine odaklanmasına yardımcı olan ek bir kodlayıcı-kod çözücü dikkat katmanına sahip kodlayıcı blok bölmesinden farklıdır.



Şekil 3.3 Transformer mimarisi (Vaswani, Shazeer Vd. 2017)

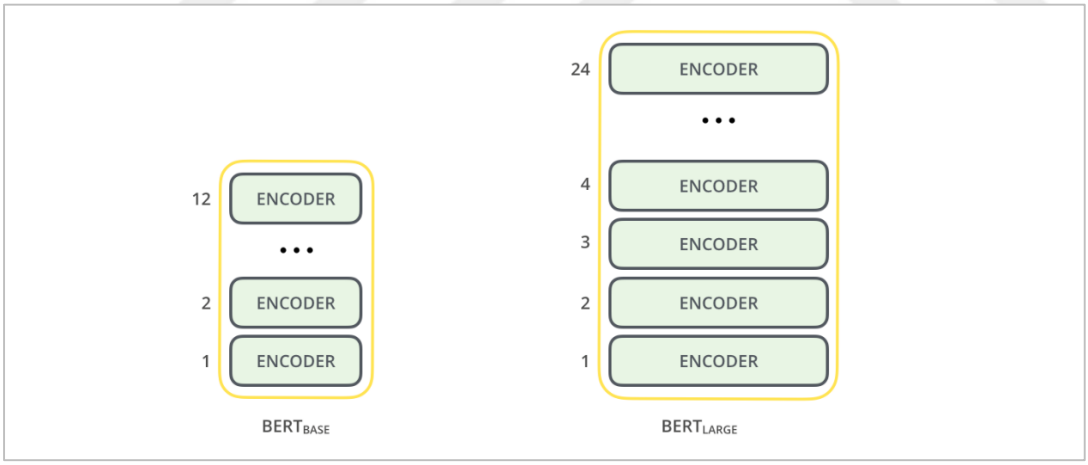
Transformer modelinin geleneksel RNN ağları üzerindeki olağanüstü performansı nedeniyle, son birkaç yılda NLP araştırmalarının çoğu Transformer NLP modellerine odaklandı ve BERT, GPT, XLNET gibi yeni bir transformer tabanlı modeller geliştirilmiştir.

Bir sonraki alt bölümde en yaygın Transformer tabanlı modelleri incelenecektir.

3.1.2.6. Transformer tabanlı NLP modelleri:

a. BERT

Transformer'den Çift Yönlü Kodlayıcı Temsilleri anlamına gelen BERT, en popüler Transformer tabanlı modeldir. Bir kod çözücüye sahip olmadan transformer modelinin orijinal uygulamasına dayanmaktadır. BERT, tüm katmanlarda hem sol hem de sağ bağlamda ortak koşullandırma yoluyla etiketlenmemiş metinden derin çift yönlü temsilleri önceden eğitmek için tasarlanmıştır (Devlin, Chang Vd. 2018). İki yönlü eğitim, temsilin sol ve sağ bağlamı birleştirmesini sağlayan maskeli dil modeli (MLM) konsepti kullanılarak yapılmaktadır. Model iki boyutta mevcuttur: 12 katman, 768 gizli, 12 baştan oluşan $BERT_{Base}$ ve 24 katman, 1024 gizli ve 16 baştan oluşan $BERT_{Large}$.



Şekil 3.4 $BERT_{Base}$ ve $BERT_{Large}$

BERT, çeşitli zorlu görevlerde son teknoloji NLP'den daha iyi performans göstermesine rağmen, hesaplama açısından pahalıdır ve bellek yoğundur. Bu sorunu çözmek için, Parametre budama (Parameter pruning), niceleme (quantization) ve bilgi damıtma (knowledge distillation) gibi çeşitli sıkıştırma yöntemleri tanıtılmıştır. Önerilen sıkıştırma yöntemlerine bağlı olarak, yeni BERT benzeri modeller önerilmiştir.

b. DistilBERT

Bilgi damıtması, derin ağları oldukça yüksek performansı korurken daha sığ ağlara sıkıştırmak için kullanılan sıkıştırma yöntemlerinden biridir. BERT'nin damıtılmış bir versiyonu olan DistilBERT, Sanh ve diğerleri (Sanh, Debut Vd. 2019) tarafından 2019'da tanıtılan BERT modeline dayanan küçük, hızlı, ucuz ve hafif bir transformatör modelidir.

Model, BERT modelindeki jeton tipi gömmelerin kaldırılmasına ek olarak, BERT temel modelinde altı olan Transformer bloklarının sayısı damıtılmış versiyonda üçe düşürülerek elde edilmiştir.
Model, BERT modeliyle karşılaştırılabilir bir performans elde etmiştir.

c. ALBERT

BERT base modeli 110 milyon parametreden oluşmaktadır ve bu da onu hesaplama açısından yoğun hale getirmektedir. Sonuçta, BERT'de kullanılan parametre sayısını azaltmak için BERT'nin hafif versiyonu geliştirilmiştir. Lan ve diğerleri tarafından (Lan, Chen Vd. 2019) bellek tüketimini azaltmak ve BERT modelinin eğitim hızını artırmak için ALBERT (A Lite BERT) önerilmiştir.
ALBERT'in ana fikri, performansı yüksek tutarken parametre sayısını azaltmaktır. ALBERT modeli, 768 gizli katman ve 128 gömme katmanı ile 12 milyon parametreye sahiptir. Google araştırma ekibi, BERT modelinde kurulan bellek sınırlamaları, uzun eğitim süreleri ve beklenmeyen model bozulması sorunlarını iki farklı parametre azaltma tekniği önererek ele almıştır. İlk teknik, faktörleştirilmiş embedding parametreleştirme (factorized embedding parameterization)dir.. Bu yöntemde, gömme katmanını 768'de tutmak yerine gömme katmanı, çarpanlara ayırma yoluyla 128 katmana düşürülmüştür. İkinci teknik, katmanlar arası parametre paylaşımıdır (cross-layer parameter sharing). Bu yöntemde sadece ilk kodlayıcının parametresi öğrenilmektedir ve aynısı tüm kodlayıcılarda kullanılmaktadır. Yani, embedding matrisini iki küçük matrise bölmek ve gruplar arasında bölünmüş yinelenen katmanları kullanmaktır. Önerilen yaklaşım, cümleler arası tutarlılığı geliştirmek için cümle sırası tahmini için kendi kendini denetleyen bir kayıp içermiştir.
ALBERT'in hafif olmasının yanı sıra, ALBERT'i BERT'den ayıran diğer şey Cümle Sırası Tahminidir (Sentence Order Prediction - SOP). SOP, hedefin verilen iki cümlenin doğru sırada olup olmadığının tespitinin olduğu bir sınıflandırma modelidir.

Öte yandan, BERT sonraki cümle tahmini (next sentence prediction - NSP) üzerinde çalışmaktadır. NSP'de, eğitim sırasında, model giriş cümle çiftlerini alır ve ikinci cümlenin orijinal metindeki bir sonraki cümle olup olmadığını tahmin etmeyi öğrenmektedir.

d. ELECTRA

ELECTRA, jeneratör ve ayırıcı olmak üzere iki transformer modelini eğiten yeni bir ön eğitim yaklaşımıdır. ELECTRA modeli, BERT modelinde olduğu gibi girdiyi maskelemek yerine, değiştirilmiş belirteç algılama (Replaced Token Detection - RTD) adı verilen yeni bir ön eğitim görevi kullanmıştır ve bu görevde, bazı belirteçleri küçük bir jeneratör ağından örneklenen makul alternatiflerle değiştirmiştir. Daha sonra, bozuk belirteçlerin orijinal kimliklerini tahmin eden bir model eğitmek yerine, bozuk girdideki her bir belirtecin bir üreteç örneğiyle değiştirilip değiştirilmediğini tahmin etmek için ayırt edici bir model eğitilmiştir (Clark, Luong Vd. 2020).

RoBERTa Facebook AI'a göre, BERT modeli önemli ölçüde yetersizdir, bu amaçla, Facebook araştırma ekibi, performansını artırmak için eğitim prosedüründe bir sonraki dizi tahmini hedefini eğitim prosedüründen çıkarmak, eğitmek için daha fazla veri kullanmak yineleme sayısını artırmak gibi birkaç değişiklik önermiştir.

BERT'de %1000 daha fazla veri ve hesaplama gücü ile eğitim metodolojisini geliştirmek için Facebook araştırma ekibi tarafından Robustly optimized BERT (RoBERTa) yaklaşımı tanıtılmıştır. RoBERTa (Liu, Ott Vd. 2019), BERT'nin ön eğitiminde NSP görevi yerine dinamik maskelemeyi tanıtmıştır. Model ayrıca çok daha büyük mini partiler (mini batches) ve öğrenme oranları ile eğitilmiştir. RoBERTa, farklı bir ön eğitim şemasına sahip bir belirteç olarak bayt düzeyinde bayt çifti kodlaması (byte-level byte-pair- encoding) veya BPE kullanmaktadır.

e. DistilRoBERTa

DistilRoBERTa, RoBERTa modelinin DistilBERT modeliyle aynı eğitim prosedürüne sahip ve RoBERTa modelinden iki kat daha hızlı ve %35 daha küçük olan damıtılmış bir versiyonudur. DistilRoBERTa, DistilBERT'ten daha moderndir ve orijinal RoBERTa'nın BERT modeline getirdiği mimari iyileştirmeler nedeniyle tavsiye edilmektedir.

3.2.Makine Öğrenmesi

Makine öğrenimi, sistemlere açıkça programlanmadan otomatik olarak öğrenme ve deneyimden iyileştirme yeteneği sağlayan bir yapay zekâ uygulamasıdır.

Makine öğreniminin 1950'lerde İngiliz matematikçi Alan Turing'in yapay zekâlı "öğrenme makinesini" önermesiyle gerçekleştiği söylenilmiştir. Arthur Samuel ilk bilgisayar öğrenme programını yazmıştır. Onun programı, bir IBM bilgisayarının dama oyununda daha uzun süre oynamasını sağlamıştır. Takip eden yıllarda, çeşitli makine öğrenimi teknikleri ortaya çıkmıştır.

Yapay sinir ağları, ağırlıklandırmaların yanlış göründüğü "yerel minimum (local minima)" probleminden müzdarip olduklarından, makine öğrenimi araştırmacıları tarafından çoğunlukla göz ardı edilmiştir.2001 yılında, bir görüntüdeki yüzleri gerçek zamanlı olarak algılamak için Adaboost adlı bir makine öğrenme algoritması geliştirilmiştir. Güçlü grafik işleme birimleri nihayet pazara girdiğinde, sinir ağları birkaç yıl daha lehine dönmemiştir. Donanım desteği ile araştırmacılar, görüntüleri çalıştırmak ve işlemek için süper bilgisayarlar yerine masaüstü bilgisayarları kullanabilmiştir.

Giderek artan muazzam hacimler ve veri çeşitliliği, bilgi işlem gücüne erişim ve satın alınabilirlik ve yüksek hızlı İnternet'in kullanılabilirliği nedeniyle son yıllarda makine öğreniminin önemi artmaktadır.

Genel olarak, büyük miktarda veri ve çok sayıda değişken içeren, ancak mevcut bir formül veya denklem içermeyen karmaşık bir görevimiz veya problemimiz olduğunda makine öğrenimini kullanılabilmektedir. Örneğin:

- Yüz tanıma ve konuşma tanımada olduğu gibi elle yazılmış kurallar ve denklemler çok karmaşık olduğunda,
- İşlem kayıtlarından dolandırıcılık tespitinde olduğu gibi, bir görevin kuralları sürekli değiştiğinde,
- Verilerin doğası değişmeye devam ettiğinde ve programın otomatikleştirilmiş ticaret, enerji talebi tahmini ve alışveriş trendlerini tahmin etmede olduğu gibi uyum sağlaması gerektiğinde.

Makine öğrenimi algoritmaları genellikle denetimli öğrenme, denetimsiz öğrenme, yarı denetimli öğrenme ve pekiştirmeli öğrenme olarak kategorize edilmektedir.

Denetimli makine öğrenimi algoritmaları, gelecekteki olayları tahmin etmek için etiketli örnekler kullanarak geçmişte öğrenilenleri yeni verilere uygulayabilmektedir. Buna karşılık, denetimsiz makine öğrenimi algoritmaları, eğitmek için kullanılan bilgiler sınıflandırılmadığında veya etiketlenmediğinde kullanılmaktadır. Denetimsiz öğrenme, sistemlerin etiketlenmemiş verilerden gizli bir yapıyı tanımlamak için bir işlevi nasıl çıkardığını incelemektedir.

Yarı denetimli makine öğrenimi algoritmaları, eğitim için hem etiketli hem de etiketsiz verileri kullanmaktadır.
Pekiştirmeli (ya da Güçlendirme) makine öğrenimi algoritmaları, eylemler üreterek ve hataları veya ödülleri keşfederek çevresiyle etkileşime giren bir öğrenme yöntemidir. Deneme yanılma araştırması ve gecikmeli ödül, pekiştirmeli öğrenmenin en belirgin özellikleridir.

Denetimli öğrenme, makine öğrenimi modelleri geliştirmek için sınıflandırma ve regresyon tekniklerini kullanmaktadır. Sınıflandırma teknikleri, örneğin bir e-postanın gerçek mi yoksa spam mı olduğu gibi ayrı tepkileri tahmin etmektedir. Tersine, Regresyon teknikleri, örneğin sıcaklıktaki değişiklikler veya güç talebindeki dalgalanmalar gibi sürekli tepkileri tahmin etmektedir.

Sınıflandırma gerçekleştirmek için yaygın algoritmalar arasında destek vektör makinesi (SVM), artırılmış ve torbalanmış karar ağaçları, k-en yakın komşu, Naive Bayes, diskriminant analizi, lojistik regresyon ve sinir ağları bulunmaktadır.

Yaygın kullanılan regresyon algoritmaları, lineer model, lineer olmayan model, düzenlileştirme, adım adım regresyon, artırılmış ve torbalanmış karar ağaçları, sinir ağları ve uyarlanabilir nöro-bulanık öğrenmeyi içermektedir.

Öte yandan, kümeleme en yaygın denetimsiz öğrenme tekniğidir. Verilerdeki gizli kalıpları veya gruplamaları bulmak için keşifsel veri analizi için kullanılmaktadır. Kümeleme gerçekleştirmek için yaygın algoritmalar arasında k-means ve k-medoids, hiyerarşik kümeleme, Gauss karışım modelleri, gizli Markov modelleri, kendi kendini

organize eden haritalar, bulanık fuzzy c-means kümeleme ve çıkarıcı (subtractive) kümeleme bulunmaktadır.

Bu çalışmada Naive Bayes ve lineer regresyon sınıflandırıcıları kullanıldığından, bu modellerin ana teorisini ve formülasyonunu açıklamak için bunlardan kısaca bahsedilecektir.

3.2.1. Çalışmada Kullanılan Makine Öğrenme Sınıflandırıcıları

a. Naive Bayes Sınıflandırıcı

Naive Bayes sınıflandırıcı, sınıflandırma için üretken bir modeldir. Naive Bayes sınıflandırıcıları, Bayes teoremine dayanmaktadır.

Bayes teoremi, istatistiksel niceliklerin koşullu olasılıkları arasındaki ilişkiyi tanımlayan bir denklemdir. Ön bilgimiz göz önüne alındığında, belirli bir sınıfa ait bir veri parçasının olasılığını hesaplayabilmemiz için bir yöntem sunmaktadır.

Bayes Teoremi şu şekilde ifade edilmektedir:

$$\boldsymbol{P(sınıf|veri)} = (\boldsymbol{P(veri|sınıf)} * \boldsymbol{P(sınıf)}) \,/\, \boldsymbol{P(veri)} \quad (3.6)$$

Burada $\boldsymbol{P(sınıf|veri)}$, sağlanan veri verilen sınıfın olasılığıdır.

Bayes sınıflandırmasında, P(L | özellikler) olarak yazabileceğimiz, gözlenen bazı özellikler verilen bir etiketin olasılığını bulmakla ilgilenilmektedir. Bayes teoremi bize bunu daha doğrudan hesaplayabileceğimiz miktarlar cinsinden nasıl ifade edeceğimizi söylemektedir:

$$\boldsymbol{P(L|\,özellikler)} = \boldsymbol{P(özellikler|\,L)P\left(L\right)P\left(özellikler\right)} \quad (3.7)$$

İki etiket L1 ve L2 için iki etiket arasında karar vermek için her bir etiket için sonsal olasılıkların oranı aşağıdaki gibi hesaplanmalıdır:

$$P(L1 \mid özellikler)P(L2 \mid özellikler)=P(özellikler \mid L1)P(özellikler \mid L2)P(L1)P(L2) \quad (3.8)$$

- **Naive Bayes Sınıflandırıcı Türleri:**

- **Çok terimli Naive Bayes:**

Çok terimli Naive Bayes, çoğunlukla belge sınıflandırma problemi için kullanılmaktadır, yani bir belgenin spor, siyaset, teknoloji vb. kategorisine ait olup

olmadığı. Sınıflandırıcı tarafından kullanılan özellikler/tahminler, belgede bulunan kelimelerin sıklığıdır.

- **Bernoulli Naive Bayes:**

Bernoulli Naive Bayes, çok terimli Naive Bayes'e benzer, ancak tahmin ediciler boole değişkenleridir. Sınıf değişkenini tahmin etmek için kullandığımız parametreler örneğin, metinde bir kelime olup olmadığı gibi yalnızca evet veya hayır değerlerini almaktadır.

- **Gauss Naive Bayes**

Tahminciler sürekli bir değer aldığında ve ayrık olmadığında, bu değerlerin bir Gauss dağılımından örneklendiğini varsayılmaktadır.

- **Naive Bayes Eğitimi**

Naive Bayes'i eğitmek kolaydır. Eğitim süreci iki nicelik çıkarmaktadır.

- Tüm sınıflar $C_m \in [C_1, C_2, \ldots, C_M]$ için $P(C_m)$.
- Tüm özellikler $x_n \in [x_1, x_2, \ldots, x_N]$ için , tüm sınıflar $C_m \in [C_1, C_2, \ldots, C_M]$ için, $P(x_n|C_m)$.

$$\mathbf{P(Cm)} = \frac{\textbf{Cm'ye ait eğitim örneklerinin sayısı}}{\textbf{Toplam eğitim örneği sayısı}} \quad (3.9)$$

$P(x_n|C_m)$'nin hesaplanması, eldeki sınıflandırma görevi için kullanılan belirli tek değişkenli modele bağlıdır. Burada farklı durumlarla karşılaşabilmektedir:

1- Sınıflandırma ikili özellikler içermekteyse (0 veya 1'e eşit karşılık gelen özellik)

$$\mathbf{P(x_n=1 \mid C_m)} = \frac{\textbf{Cm sınıfında xn=1 olan eğitim örneklerinin sayısı}}{\textbf{Cm sınıfının toplam eğitim örneği sayısı}} \quad (3.10)$$

$$\mathbf{P(x_n = 0 | C_m)} = \frac{\textbf{Cm sınıfında xn=0 olan eğitim örneği sayısı}}{\textbf{Cm sınıfının toplam eğitim örneğisayısı}} \quad (3.11)$$

Değişken, birbirini dışlayan iki değerden yalnızca birini alabileceğinden, $P(x_n = 1|C_m) = 1 - P(x_n = 0|C_m)$ bu nedenle, bu genellikle yalnızca başarı olasılığını öğrenerek tek değişkenli bir Bernoulli dağılımı olarak uygulanabilmektedir.

2- Sınıflandırma, Kategorik özellikleri içermekteyse, yukarıdaki Bernolli dağılımı, çok terimli bir dağılımla değiştirilmektedir. Bu durumda, her olası $X_n = v$ değeri için olasılıklar şu şekilde hesaplanmaktadır:

$$\mathbf{P(x_n = v|C_m)} = \frac{\textbf{Cm sınıfında xn=v ile eğitim örneklerinin sayısı}}{\textbf{Cm sınıfının toplam eğitim örneği sayısı}} \quad (3.12)$$

3- Sınıflandırma Sürekli özellikleri içermekteyse, Gauss dağılımı veya düzgün dağılım gibi $P(x_n|C_m)$ olasılığını modellemek için uygun bir sürekli dağılım kullanılmaktadır.

- **Naive Bayes sınıflandırıcısı neden kullanılmıştır?**

Naive Bayes sınıflandırıcısının başlıca avantajları aşağıdaki gibi özetlenebilmektedir:

1- Naive Bayes, bir veri kümesi sınıfını tahmin etmek için hızlı ve kolay makine öğrenme algoritmalarından biridir,
2- İkili ve Çok Sınıflı Sınıflandırmalar için kullanılabilmektedir,
3- Diğer Algoritmalara kıyasla Çok sınıflı tahminlerde iyi performans göstermektedir,
4- Metin sınıflandırma problemleri için en popüler seçimdir.

Buna karşılık, Naive Bayes, tüm özelliklerin bağımsız veya ilişkisiz olduğunu varsaydığından, özellikler arasındaki ilişkiyi öğrenememektedir.

b. Doğrusal Regresyon

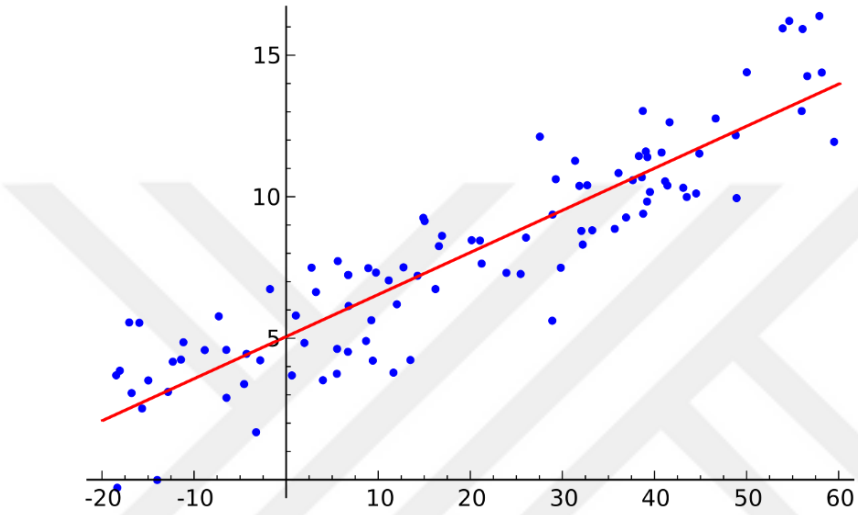
- **Regresyon**

Regresyon, bağımsız tahmin edicilere dayalı bir hedef değeri modelleme yöntemidir. Bu yöntem daha çok değişkenler arasındaki neden-sonuç ilişkisini tahmin etmek ve bulmak için kullanılmaktadır. Regresyon teknikleri çoğunlukla bağımsız değişkenlerin sayısına ve bağımsız ve bağımlı değişkenler arasındaki ilişkinin türüne göre farklılık göstermektedir.

Doğrusallık: Y bağımlı değişkeninin bağımsız değişkenlerle doğrusal olarak ilişkili olması gerektiğini belirtmektedir.

Bu nedenle, **Doğrusal Regresyon**, modelin bağımsız ve bağımlı değişken arasında en uygun doğrusal çizgiyi bulduğu, yani bağımlı ve bağımsız değişken arasındaki doğrusal ilişkiyi bulduğu denetimli Makine Öğrenimi modelidir. İki tip Basit Doğrusal Regresyon ve Çoklu Doğrusal Regresyon içermektedir.

Basit Doğrusal Regresyon, yalnızca bir bağımsız değişkenin bulunduğu ve modelin bağımlı değişkenle doğrusal ilişkisini bulması gerektiği yerdir. Oysa **Çoklu Doğrusal Regresyon**da modelin ilişkiyi bulması için birden fazla bağımsız değişken vardır.



Şekil 3.5 Doğrusal Regresyon

Basit Doğrusal Regresyon denklemi:

$$y = b_0 + b_1 x \tag{3.13}$$

b_0 kesişim, b_1 katsayı veya eğim, x bağımsız değişken ve Y bağımlı değişkendir.

Çoklu Doğrusal Regresyon denklemi ise:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + \cdots + b_n x_n \tag{3.14}$$

b_0 kesme noktası olduğunda, $b_1, b_2, b_3, b_4, \ldots, b_n$ bağımsız $x_1, x_2, x_3, x_4 \ldots, x_n$ değişkenlerinin katsayıları veya eğimleridir ve Y bağımlı değişkendir.

Doğrusal regresyon algoritmasının amacı, b_0 ve b_1 için en iyi değerleri bulmaktır. Bunu elde etmek için Maliyet Fonksiyonuna ihtiyacımız vardır. Maliyet fonksiyonu, veri noktaları için en uygun çizgiyi sağlayacak olan b_0 ve b_1 için mümkün olan en iyi değerleri bulmamıza yardımcı olmaktadır. Bunu başarmak için tahmin edilen değer ile gerçek değer arasındaki hatayı en aza indirilmelidir.

$$J = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} (\mathrm{pred}\, i - y_i)^2 \quad (3.15)$$

Önceki fonksiyonda, hata farkının karesini alınmaktadır ve tüm veri noktalarının toplamını alınmaktadır ve bu değeri toplam veri noktası sayısına bölünmektedir. Bu, tüm veri noktaları üzerinde ortalama kare hatası sağlamaktadır. Bu nedenle, bu maliyet fonksiyonu, Ortalama Kare Hata (Mean Squared Error /MSE) işlevi olarak da bilinmektedir.

3.3.Derin Öğrenme

Derin sinirsel öğrenme veya derin sinir ağı olarak da bilinen derin öğrenme, yapay sinir ağları adı verilen beynin yapısı ve işlevinden ilham alan algoritmalarla ilgili makine öğreniminin bir alt alanıdır.
Derin öğrenmedeki "derin" kelimesi, ağda ikiden çok katmanın kullanımını ifade etmektedir. Derin öğrenmedeki her seviye, girdi verilerini biraz daha soyut ve bileşik bir temsile dönüştürmeyi öğrenmektedir.

Bir ağ üç tür katmana sahip olabilmektedir: etki alanından ham girdi alan girdi katmanları, başka bir katmandan girdi alıp çıktıyı başka bir katmana ileten gizli katmanlar ve bir tahminde bulunan çıktı katmanlarıdır.

3.3.1. Neden Derin Öğrenme?

Son birkaç yılda yapay sinir ağları alanındaki büyük atılımlar, şirketleri yapay zekâ stratejilerinin bir parçası olarak derin öğrenme çözümlerini uygulamaya ittiğinden, derin öğrenme bugün çok popüler hale gelmiştir.

Derin öğrenmenin büyük bir avantajı ve neden popüler hale geldiğini anlamanın önemli bir parçası, çok büyük miktarda veri tarafından desteklenmesidir.

Ayrıca, Geleneksel Makine öğrenimi tekniklerinde, uygulanan özelliklerin çoğunun bir alan uzmanı tarafından tanımlanması gerekmektedir. Öte yandan, Derin Öğrenme algoritmaları, verilerden artımlı bir şekilde üst düzey özellikleri öğrenmeye çalışmaktadır. Bu, alan uzmanlığı ve temel özellik çıkarma ihtiyacını ortadan kaldırmaktadır.

Derin Öğrenme, geleneksel Makine Öğrenimi algoritmalarının aksine üst düzey makineler gerektirmektedir.

Derin Öğrenme ve Makine Öğrenimi tekniği arasındaki bir diğer önemli fark, problem çözme yaklaşımıdır. Derin Öğrenme teknikleri, sorunu uçtan uca çözme eğilimindedir, ancak Makine öğrenimi teknikleri, problem ifadelerinin önce çözülmesi için farklı parçalara ayrılmasına ve ardından sonuçlarının son aşamada birleştirilmesine ihtiyaç duymaktadır.

3.3.2. Derin Öğrenme ne Zaman Kullanılır veya Kullanılmaz?

- Derin Öğrenme, veri boyutunun büyük ve çeşitliliğin fazla olduğu durumlarda etkindir. Ancak küçük veri boyutuyla geleneksel Makine Öğrenimi algoritmaları tercih edilmektedir,
- Derin Öğrenme tekniklerinin makul sürede eğitilmesi için üst düzey işlem gücü ve yazılım altyapısına sahip olması gerekmektedir,
- Özellik iç gözlemi için alan anlama eksikliği olduğunda kullanılmaktadır,
- Görüntü sınıflandırma, doğal dil işleme ve konuşma tanıma gibi karmaşık sorunları ele alırken tercih edilmektedir.

3.3.3. Derin Öğrenmede Temel Kavramlar

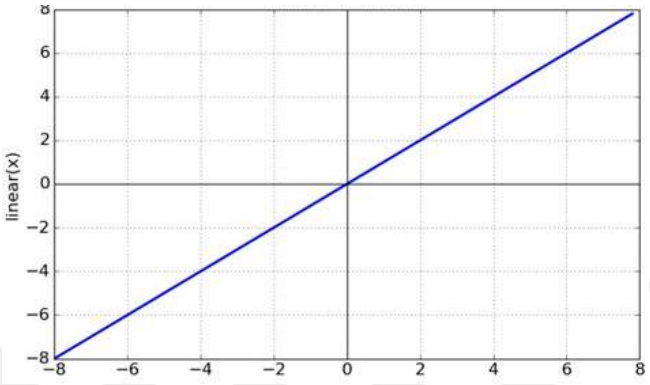
3.3.3.1. Aktivasyon Fonksiyonları

Aktivasyon fonksiyonu, derin öğrenme modelinin en önemli parametrelerinden biridir. Ağın değişkenleri arasındaki her türlü sürekli ve karmaşık ilişkiyi öğrenmek ve tahmin etmek için kullanılmaktadır. Aktivasyon fonksiyonu, çıktıyı belirlemek için bir sinir ağının çıktı ucuna eklenmektedir. Aktivasyon fonksiyonunun seçimi, sinir ağının kapasitesi ve performansı üzerinde büyük bir etkiye sahiptir. Örneğin, gizli katmandaki aktivasyon fonksiyonunun seçimi, ağ modelinin eğitim veri setini ne kadar iyi öğrendiğini kontrol ederken, çıktı katmanındaki aktivasyon fonksiyonunun seçimi, modelin yapabileceği tahminlerin türünü tanımlayacaktır. Logistic (Sigmoid), Hyperbolic Tangent (Tanh),ve Rectified Linear Activation (ReLU) gizli katmanda kullanılan en popüler aktivasyon fonksiyonlarıdır. Öte yandan çıkış katmanında kullanılan Lineer, Logistic (Sigmoid) ve Softmax olmak fonksiyonları en sık tercin edilenlerdir.

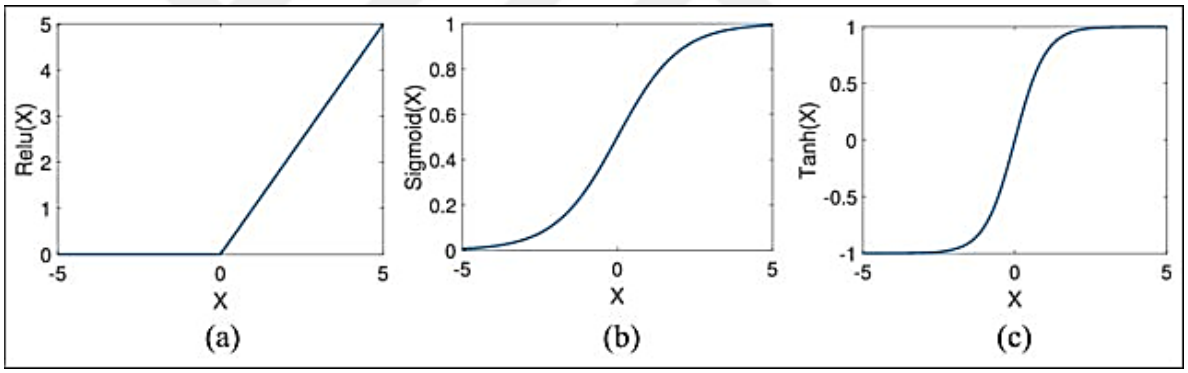
a. Doğrusal Fonksiyon (Linear function)

Doğrusal fonksiyon girdilerini sabit bir çarpanla çarpan tek bir satırı temsil etmektedir. Doğrusal bir fonksiyon aşağıdaki forma sahiptir:

$$\boldsymbol{y = f(x) = a + bx} \tag{3.16}$$



Şekil 3.6 doğrusal fonksiyon



Şekil 3.7 Doğrusal olmayan fonksiyonları(Carson, Chakshu Vd. 2020)

a) ReLu aktivasyon fonksiyonu; **(b)** lojistik (sigmoid) aktivasyon fonksiyonu ve **(c)** tanh aktivasyon fonksiyonu

b. Doğrusal Olmayan Fonksiyonları

- **Sigmoid Fonksiyonu:**

Sigmoidal eğri veya lojistik fonksiyon olarak da adlandırılan sigmoid fonksiyonu, 0 ile 1 aralığında S şeklinde bir eğriye sahip fonksiyondur, ve aşağıdaki denklemi ile tanımlanmıştır

$$\boldsymbol{f(x) = \frac{1}{1+e^{-x}}} \tag{3.17}$$

- **Hiperbolik Tanjant (tanh)**

Hiperbolik tanjant -1 ila 1 aralığına sahip S-şekilli eğriyi ifade etmektedir. Matematiksel tanımı aşağıdaki denklemi ile verilmektedir.

$$\tanh(x) = \frac{\sin(x)}{\cos(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.18}$$

- **Rectified Linear Unit (ReLU)**

Giriş değeri set değerinden küçük olduğunda 0 veren ve girişin set değerinden büyük olması durumunda doğrusal kat veren tek noktalı bir fonksiyondur: Matematiksel ifadesi

$$\boldsymbol{f(x) = max\,(0, x)} \tag{3.19}$$

benklemi ile verilir. Burada x bir nöronun girdisidir.

- **Softmax Fonksiyonu**

Softmax fonksiyonu aşağıdaki gibi hesaplanmaktadır:

$$\mathbf{y}(\boldsymbol{x_i}) = \frac{\boldsymbol{e^{x_i}}}{\sum \boldsymbol{e^{x_i}}} \tag{3.20}$$

Burada x, çıktıların bir vektörüdür.

3.3.3.2. Kayıp Fonksiyonları

Genellikle, sinir ağları ile hatayı en aza indirmeye çalışılmaktadır. Bu nedenle, amaç fonksiyonu genellikle bir maliyet fonksiyonu veya bir kayıp fonksiyonu olarak adlandırılmaktadır ve kayıp fonksiyonu tarafından hesaplanan değer "kayıp" olarak adlandırılmaktadır.

Bir sinir ağındaki bir dizi ağırlıkların hatasını tahmin etmek için kullanılabilecek birçok fonksiyon vardır.

- **Kayıp fonksiyon türleri**

Günümüzde en sık kullanılan kayıp fonksiyonları arasında Ortalama kare hatası, Olabilirlik kaykı ve Log kaybı bulunmaktadır. Bunları kısaca detalandıralım.

a. Ortalama kare hatası (Mean squared error /MSE)

Ortalama kare hatası (MSE), temel kayıp fonksiyonlarındandır. Anlaşılması ve uygulanması kolaydır ve genellikle oldukça iyi çalışmaktadır. Ortalama kare hatasını hesaplamak için, tahminler ve gerçek değerler arasındaki farkın karesinin ortalaması alınmaktadır.

b. Olabilirlik kaybı (Likelihood loss)

Olabilirlik kayıp fonksiyonu da nispeten basittir ve sınıflandırma problemlerinde yaygın olarak kullanılmaktadır. Fonksiyon, her girdi örneği için tahmin edilen olasılığı almaktadır ve bunları çarpmaktadır. Çıktı tam olarak insan tarafından yorumlanabilir olmasa da modelleri karşılaştırmak için kullanışlıdır.

$X_1, X_2, \ldots, X_n$'nin bir ortak yoğunluk fonksiyonu $f(X_1, X_2, \ldots, X_n|\theta)$ olsun.

Verilen $X_1 = x_1$, $X_2 = x_2$, .., $X_n = x_n$ gözlenmektedir,

θ fonksiyonu olabilirlik fonksiyonudur olarak şu şekilde tanımlanmaktadır:

$$\mathbf{L(\theta) = L(\theta|x_1, x_2, \ldots, x_n) = f(x_1, x_2, \ldots, x_n|\theta)} \quad (3.21)$$

Maksimum Olabilirlik Tahmini, yoğunluk tahmini problemini çözmek için olasılıksal bir çerçevedir. Gözlenen verileri en iyi açıklayan olasılık dağılımını ve parametreleri bulmak için bir olabilirlik fonksiyonunu maksimize etmeyi içermektedir. Model parametreleri bulmanın bir optimizasyon problemi olarak çerçevelenebileceği, makine öğreniminde tahmine dayalı modelleme için bir çerçeve sağlamaktadır.

Maksimum olabilirlik çerçevesi altında, iki olasılık dağılımı arasındaki hata, çapraz entropi kullanılarak ölçülmektedir.

c. Log Kaybı (Cross Entropy Loss)

Log kaybı ya da çapraz entropi kaybı, sınıflandırma problemlerinde de sıklıkla kullanılan bir kayıp fonksiyonudur.

Çapraz entropi, P ve Q'dan gelen olayların olasılıkları kullanılarak aşağıdaki gibi hesaplanabilmektedir:

$$H(P,Q) = \sum_{x \in X} P(x) \log(Q(x)) \tag{3.22}$$

Bu, logaritmalarla olabilirlik fonksiyonunun basit bir modifikasyonudur.

$$-\mathrm{y}\log(\mathrm{p}) + (1-\mathrm{y})\log(1-\mathrm{p})) \tag{3.23}$$

3.3.3.3. Aşırı Uyum Sorunu

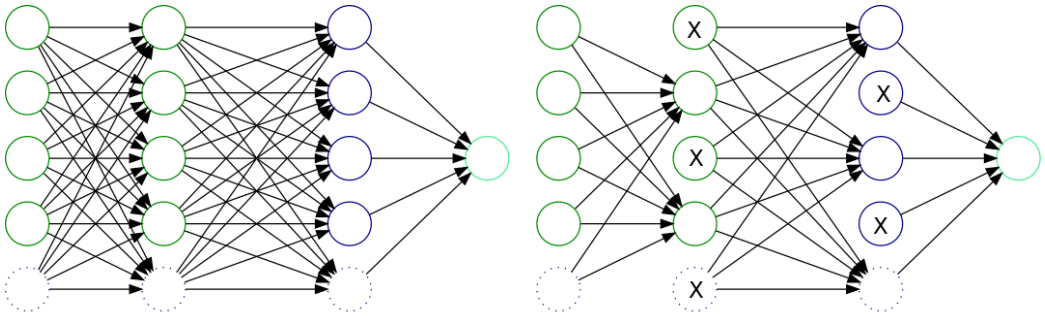
Bir modelin eğitim verilerine tamamen uymaya çalıştığı ve sonunda veri kalıplarını, gürültü ve rastgele dalgalanmaları ezberlediği derin öğrenme algoritmalarında yaygın bir sorundur.

Model yüksek bir varyansa sahip olduğunda, yani model eğitim verilerinde iyi performans gösterdiğinde ancak değerlendirme setinde kötü performans gösterdiğinde aşırı uyum oluşmaktadır.

Bu sorunun üstesinden gelmek için, kayıp fonksiyonuna bir penaltı ekleyerek aşırı uymayı azaltmak için düzenlileştirme kullanılmaktadır. Bu penaltı eklenerek model, birbirine bağlı özellik ağırlıkları kümesini öğrenmeyecek şekilde eğitilmektedir.

3.3.3.4. Bırakma (Dropout)

"Bırakma /dropout" **S**inir ağlarında nöronlar arasında birbirine bağlı öğrenmeyi azaltmaya yardımcı olan bir düzenleme yaklaşımıdır."Bırakma terimi, bir sinir ağındaki nöronların çıkarılması anlamına gelmektedir, burada bir bırakma katmanı, eğitim süreci sırasında sinir ağından rastgele bir şekilde düşerek modelin boyutunun küçülmesine de neden olmaktadır. Ayrıca, aşırı uyum probleminin etkisinin azalmasına da yardımcı olur.



3.8.a.Bırakma işlemi olmadan standart sınır ağı

3.8.b.Bırakma işleminden sonra

Şekil 3.8 Bırakam işlemi(Ha, Tran Vd. 2019)

3.3.3.5.Veri Büyütme

Aşırı uyum sorununun yaygın olarak meydana gelmesini önlemek için iki yöntem uygulanabilmektedir, düzenleme (regularization) veya daha fazla veri ile eğitim.

Veri kümesinin boyutu küçük olduğunda ve daha fazla örnek bulunmadığında, bu durumda Veri Büyütme (Data augmentation/DA) kullanışlı hale gelmektedir.

Wikipedia'ya göre veri büyütme "hâlihazırda var olan verilerin biraz değiştirilmiş kopyaları veya mevcut verilerden yeni oluşturulmuş sentetik veriler ekleyerek veri miktarını artırmak için kullanılan teknikler" biçimindedir."

Veri büyütme, veri kümelerini eğitmek için yeni ve farklı örnekler oluşturarak makine öğrenimi modellerinin performansını ve sonuçlarını iyileştirmek için yararlıdır.

- **Veri Büyütme teknikleri**

Görüntü tanıma ve doğal dil işleme için veri büyütmede klasik ve gelişmiş teknikler vardır. Veri büyütme için klasik görüntü işleme faaliyetleri şunlardır:

Dolgu (padding), rastgele döndürme, yeniden ölçeklendirme, dikey ve yatay çevirme, kırpma, yakınlaştırma, koyulaştırma ve parlaklaştırma/renk değiştirme, gri tonlama, kontrastı değiştirme, gürültü ekleme ve rastgele silme.

NLP'de veri büyütme için yaygın yöntemler şunlardır:

- Kolay Veri Geliştirme (Easy Data Augmentation/EDA) işlemleri: eşanlamlı değiştirme, sözcük ekleme, sözcük değiştirme ve sözcük silme gibi işlemler.
- Geri çeviri.
- Bağlamsallaştırılmış sözcük yerleştirmeleri (Contextualized word embeddings).

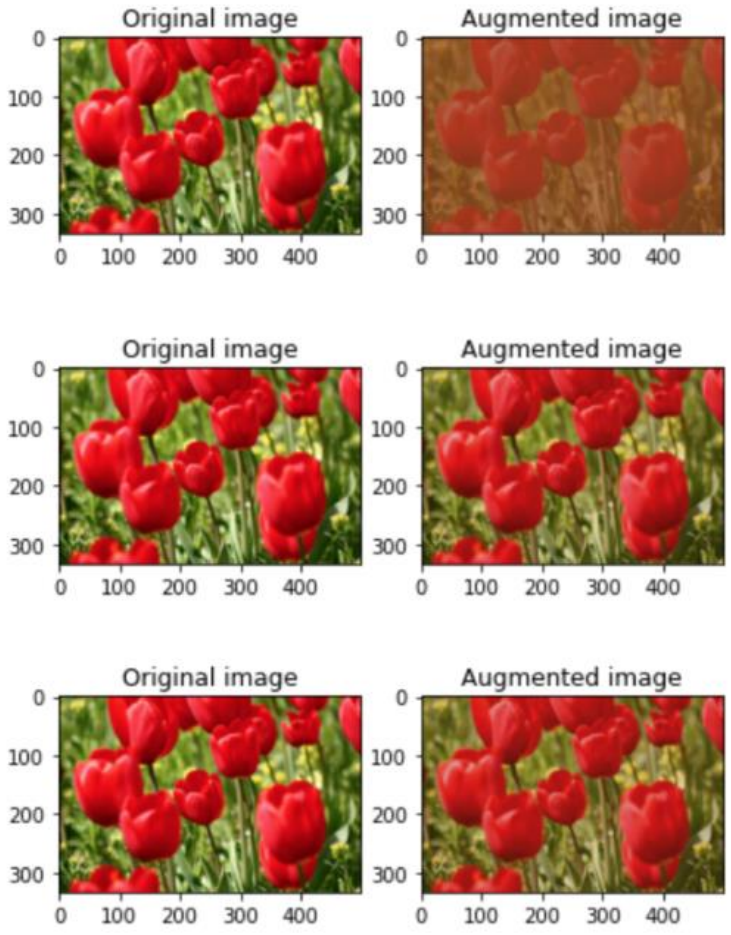
3.3.3.6. Öğrenme oranı düşüşü (Learning rate decay):

Öğrenme oranı, modelin probleme ne kadar hızlı uyarlandığını kontrol eden bir hiper parametredir. Başka bir deyişle, model ağırlıkları her değiştirildiğinde tahmin edilen hataya yanıt olarak modelin ne kadar değişiklik yaşadığını kontrol etmektedir.

Daha küçük öğrenme oranları, her güncellemenin ağırlıklarında yapılan daha küçük değişiklikler göz önüne alındığında daha fazla eğitim dönemi gerektirirken, daha büyük öğrenme oranları hızlı değişikliklerle sonuçlanır ve daha az eğitim dönemi gerektirmektedir.

Bir öğrenme oranı belirlerken, yakınsama ve aşma oranı arasında bir denge vardır. Öğrenme oranı çok büyükse, modelin çok hızlı bir şekilde optimal olmayan bir çözüme yakınsamasına neden olabilirken, öğrenme oranı çok küçükse sürecin tıkanmasına neden olabilmektedir.

Öğrenme oranı azalma yöntemi (öğrenme oranı tavlama veya uyarlamalı öğrenme oranları olarak da adlandırılmaktadır), performansı artırmak ve eğitim süresini azaltmak için öğrenme oranını uyarlama sürecidir.



Şekil 3.9 Veri büyütme örneği - görüntünün kontrastını değiştirme

(Kaynak: https://www.tensorflow.org/tutorials/images/data_augmentation)

3.3.3.7. Transfer Öğrenimi

Transfer öğrenimi, önceden eğitilmiş bir modelden özellik temsillerinden yararlanmakla ilgilidir, bu nedenle sıfırdan yeni bir model eğitmeye gerek yoktur.
Önceden eğitilmiş modeller genellikle belirli bir alanda standart bir kıyaslama olan geniş veri kümeleri üzerinde eğitilmektedir. Daha sonra modellerden elde edilen ağırlıklar başka görevlerde tekrar kullanılabilmektedir.

3.3.3.8. Sıfırdan eğitim (Training from scratch):

Bu yöntem, bir geliştiricinin büyük bir etiketli veri seti toplamasını ve özellikleri ve modeli öğrenebilen bir ağ mimarisi yapılandırmasını gerektirmektedir.
Bu teknik özellikle yeni uygulamaların yanı sıra çok sayıda çıktı kategorisine sahip uygulamalar için kullanışlıdır. Bununla birlikte, bu yaklaşım, eğitimin günler veya haftalar sürmesine neden olan büyük miktarda veri gerektirdiğinden daha az yaygın bir yaklaşımdır.

3.3.3.9. Stokastik Gradyan inişi (Stochastic gradient descent)

Stokastik gardyan inişi, uygun düzgünlük özelliklerine sahip bir amaç fonksiyonunu optimize etmek için yinelemeli bir yöntemdir.

Gradyan, bir yüzeyin eğimi anlamına gelmektedir. Yani gardyan inişi, kelimenin tam anlamıyla, o yüzeydeki en alçak noktaya ulaşmak için bir yokuştan aşağı inmek anlamına gelmektedir. Bu nedenle, gradyan inişi, bir fonksiyon üzerindeki rastgele bir noktadan başlayan ve o fonksiyonun en alt noktasına ulaşana kadar kademeli olarak eğiminden aşağı inen yinelemeli bir algoritma olarak tanımlanabilmektedir. Genel fikir, rastgele bir nokta ile başlamak ve bu noktayı her yinelemede güncellemenin bir yolunu bulmaktır, öyle ki eğimi indirilmektedir. Algoritma aşağıdaki adımları içermektedir:

1. Her parametreye/özelliğe göre amaç fonksiyonunun eğimini bulun. Başka bir deyişle, fonksiyonun gradyanını hesaplayın.
2. Parametreler için rastgele bir başlangıç değeri seçin.
3. Parametre değerlerini takarak gradyan işlevini güncelleyin.
4. Her bir özellik için adım boyutlarını şu şekilde hesaplayın: adım boyutu = gradyan * öğrenme oranı.

5. Yeni parametreleri şu şekilde hesaplayın: yeni params = eski params - adım boyutu
6. Gradyan neredeyse 0 olana kadar 3 ila 5 arasındaki adımları tekrarlayın.

Genel olarak, üç tür Gradyan İnişi vardır:
- Toplu Gradyan İnişi
- Stokastik Gradyan İnişi
- Mini toplu Gradyan İnişi

Gradyan iniş algoritmasının, algoritmanın her yinelemesi için yaptığımız büyük miktarda hesaplama gibi bazı dezavantajları vardır. Büyük veri kümesini işlerken gerçekleştirilmesi hesaplama açısından çok pahalı hale gelmektedir. Stokastik gradyan iniş (SGD) algoritması, her yineleme için tüm veri kümesi yerine rastgele birkaç örnek seçerek sorunu çözmektedir. Örnek rastgele karıştırılmaktadır ve yinelemeyi gerçekleştirmek için seçilmektedir. Dolayısıyla, SGD'de, tüm örneklerin maliyet fonksiyonunun gradyanının toplamı yerine, her yinelemede tek bir örneğin maliyet fonksiyonunun gradyanını bulunmaktadır.

Öte yandan, her yineleme için veri kümesinden yalnızca bir örnek rastgele seçildiğinden, minimuma ulaşmak için algoritmanın izlediği yol genellikle tipik Gradient Descent algoritmasından daha gürültülüdür. Ve minimuma ulaşmak için genellikle daha fazla yineleme gerekmektedir.

Modeller, doğrudan yeni görevler üzerinde tahminlerde bulunmak için kullanılabilmekte veya yeni bir modelin eğitim sürecine entegre edilebilmektedir.
Bu yöntem diğerlerinden çok daha az veri gerektirme avantajına sahiptir, bunun yanında önceden eğitilmiş modelleri yeni bir modele dâhil etmek daha düşük eğitim süresi ve daha düşük genelleme hatası sağlamaktadır.

3.3.3.10. Adam optimizasyon algoritması

Adam optimizasyon algoritması, bilgisayar vizyonu ve doğal dil işlemede derin öğrenme uygulamaları için son zamanlarda daha geniş bir şekilde benimsendiğini gören stokastik gradyan inişinin bir uzantısıdır. OpenAI'den Diederik Kingma ve Toronto Üniversitesi'nden Jimmy Ba tarafından 2015 ICLR "Adam: Stokastik Optimizasyon İçin Bir Yöntem" makalelerinde sunulmuştur (Kingma ve Ba 2014).

Adam algoritmasının başlıca avantajları aşağıdaki gibi özetlenebilmektedir:

- Uygulaması kolaydır.

- Hesaplama açısından verimlidir.

- Küçük bellek gereksinimi.

- Degradelerin değişmezden çapraza yeniden ölçeklenmesi.

- Veri ve/veya parametreler açısından büyük problemler için çok uygundur.

- Durağan olmayan amaçlar için uygundur.

- Çok gürültülü/veya seyrek eğimli problemler için uygundur.

- Hiper-parametrelerin sezgisel yorumu vardır ve genellikle çok az ayarlama gerektirir.

Stokastik gradyan inişi, tüm ağırlık güncellemeleri için tek bir öğrenme oranını korumaktadır ve eğitim sırasında öğrenme oranı değişmemektedir. Öte yandan, Adam algoritmasında her ağ ağırlığı için bir öğrenme oranı korunmaktadır ve öğrenme geliştikçe ayrı ayrı uyarlanmaktadır.

• Adam Yapılandırma Parametreleri

• **alfa**. Ayrıca öğrenme oranı veya adım boyutu olarak da adlandırılmaktadır. Ağırlıkların güncellendiği oran. Daha büyük değerler, oran güncellenmeden önce daha hızlı ilk öğrenmeyle sonuçlanmaktadır. Daha küçük değerler, eğitim sırasında öğrenmeyi yavaşlatmaktadır.

• **beta1**. İlk an tahminleri için üstel bozulma oranıdır.

• **beta2**. İkinci an tahminleri için üstel bozulma oranı. NLP ve bilgisayarla görme problemleri gibi Seyrek gradyanlı problemlerde bu değer 1.0'a yakın olarak ayarlanmalıdır.

• **epsilon**. Uygulamada sıfıra bölünmeyi önlemek için çok küçük bir sayıdır (örneğin 10E-8).

Orijinal makalede Adam, Şekil 3'te görülebileceği gibi yakınsamanın teorik analizin beklentilerini karşıladığını göstermek için ampirik olarak gösterilmiştir. Adam,

MNIST rakam tanıma ve IMDB duygu analizi veri kümeleri, MNIST veri kümesinde bir Çok Katmanlı Perceptron algoritması ve CIFAR-10 görüntü tanıma veri kümesinde Evrişimli Sinir Ağları uygulanmıştır.

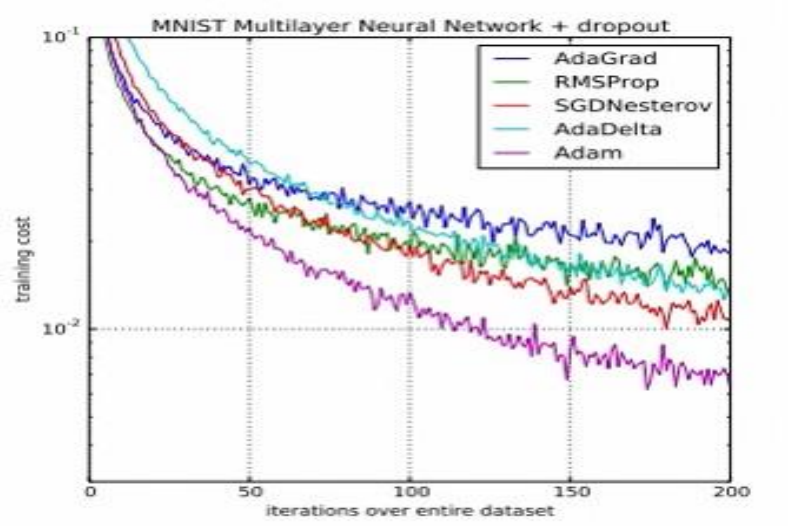
Yazarlar şu sonuca varmışlardır: "Büyük modeller ve veri kümeleri kullanarak Adam'ın pratik derin öğrenme problemlerini verimli bir şekilde çözebileceğini gösteriyoruz" (Kingma ve Ba 2014).

3.3.3.11. Batch boyutu ve Epoch

Batch boyutu ve Epoch sayısı bir tam sayıdır ve öğrenme algoritması için en önemli hiper parametreleri temsil etmektedir.

Batch boyutu, model güncellenmeden önce işlenen numune sayısıdır. Epoch sayısı, öğrenme algoritmasının tüm eğitim veri kümesi boyunca çalışacağı sayıyı tanımlayan bir hiper parametredir.

Bir batchın boyutu, eğitim veri setindeki numunelerin sayısından veya bire eşit veya daha büyük olmalıdır. Dönemlerin sayısı, bir ile sonsuz arasında bir tamsayı değerine ayarlanabilmektedir.



Şekil 3.10 Adam'ın Çok Katmanlı Algılayıcı Eğitimini Diğer Optimizasyon Algoritmalarıyla Karşılaştırılması (Kingma ve Ba 2014)

3.3.3.12. Hiper parametreler ayarlama (hyper parameters tuning)

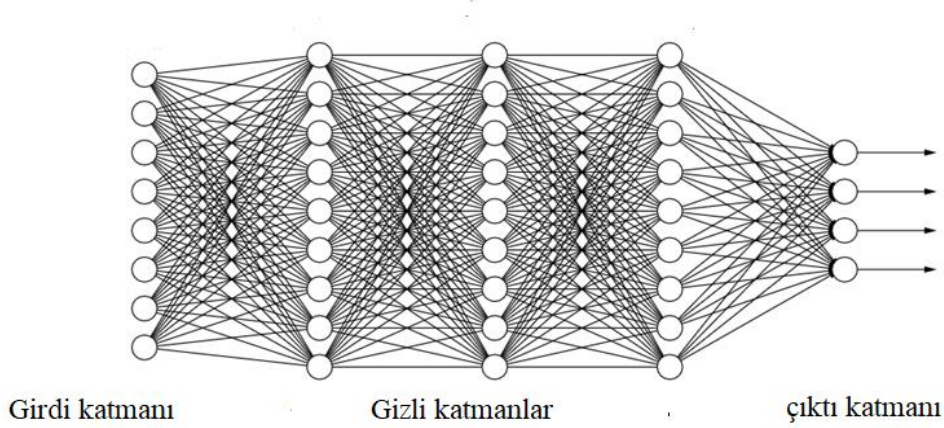
Hiper parametrelerin, derin öğrenme modellerinin performansı üzerinde doğrudan etkisi vardır. Bu nedenle, üstün performans elde etmek için bu parametreleri optimize etmek önemlidir. Hiper parametreleri optimize etmek için en yaygın stratejiler (Grid Search), Rastgele Arama (Random Search), Bayes Optimizasyonu (Bayesian Optimization), Gradyan Tabanlı Optimizasyon (Gradient-Based Optimization), ve Evrimsel Optimizasyon'dur (Evolutionary Optimization).

3.3.4. Derin Öğrenme Algoritması Türleri

Derin öğrenmede farklı modeller vardır. Aşağıda en önemlileri incelenecektir:

3.3.4.1. Klasik Sinir Ağları

Tam Bağlantılı Sinir Ağları (Fully connected neural networks/FCNN'ler) olarak da bilinmektedir, genellikle nöronların sürekli katmana bağlı olduğu çok katmanlı algılayıcıları tarafından tanımlanmaktadır. Bu, bir katmandaki tüm düğümlerin veya nöronların bir sonraki katmandaki nöronlara bağlı olduğu anlamına gelmektedir. Tam bağlantılı ağların en büyük avantajı, "yapıdan bağımsız" olmalarıdır. Yani, girdi hakkında özel bir varsayımda bulunulmasına gerek yoktur. Bu onu çok geniş bir şekilde uygulanabilir kılmaktadır, ancak bu ağların performansı, bir problem alanının yapısına ayarlanmış özel amaçlı ağların performansından daha zayıftır. Model üç fonksiyon içermektedir: doğrusal fonksiyon, sigmoid fonksiyon ve hiperbolik tanjant fonksiyonudur.



Şekil 3.11 Tam bağlantılı sinir ağı

3.3.4.2. Convolutional Neural Networks (Evrişimli Sinir Ağları)

Bir evrişimsel sinir ağı (CNN veya ConvNet), katmanlarından en az birinde genel matris çarpımı yerine evrişim adı verilen matematiksel bir işlemi kullanan bir derin sinir ağı sınıfıdır.

Bir ConvNet'te gereken ön işleme, minimum ön işleme gerektirecek şekilde tasarlanmış çok katmanlı algılayıcıların bir varyasyonunu kullandığından, diğer sınıflandırma algoritmalarına kıyasla çok daha düşüktür.

Görüntü ve görüntü olmayan verilerde uzmanlaşmak için en verimli esnek modellerden biri olarak kabul edilebilmektedir.
ConvNet mimarilerini oluşturmak için üç ana katman türü kullanılmaktadır: Convolutional (Evrişim) katmanı, Pooling katmanı ve tam bağlantılı katmandır.

a. Evrişim Katmanı

Girdiden çeşitli özellikleri çıkarmak için kullanılan ilk katmandır.
Evrişimin matematiksel işlemi, girdi ile belirli bir MxM boyutundaki bir filtre arasında gerçekleştirilmektedir.Filtreyi girişin (resim veya metin) üzerine kaydırarak, filtrenin boyutuna (MxM) göre filtre ile giriş görüntüsünün bölümleri arasında nokta çarpımı alınmaktadır.Bir Evrişim Katmanı, bir pooling Katmanı tarafından takip edilmektedir.

b. Pooling Katmanı

Bu katmanın birincil amacı, hesaplama maliyetlerini azaltmak için kıvrımlı özellik haritasının boyutunu küçültmektir.

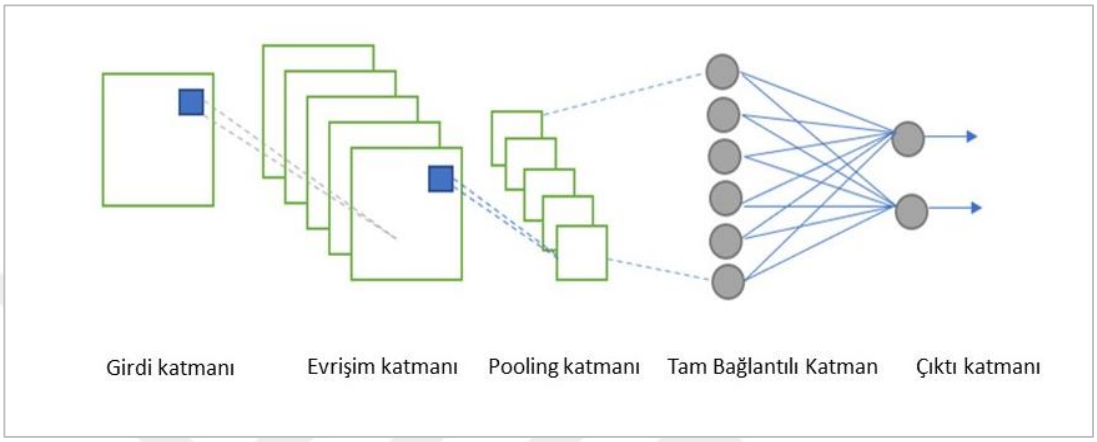
c. Tam Bağlantılı Katman

Tam bağlantılı katmanı genellikle bir model için kayıp işlevini derleyen gizli bir katman olarak tanımlanmaktadır. Nöronlarla birlikte ağırlıklar ve önyargılardan oluşmak ve nöronları iki farklı katman arasında bağlamak için kullanılmaktadır. Bu katmanlar genellikle çıktı katmanından önce yerleştirilmiştir ve bir CNN Mimarisinin son birkaç katmanını oluşturmaktadır.

ReLU, Softmax, Tanh ve Sigmoid fonksiyonları gibi evrişimli sinir ağlarında yaygın olarak kullanılan birkaç aktivasyon fonksiyonu vardır. Bu fonksiyonlar her birinin belirli bir kullanımı vardır. İkili sınıflandırma CNN modeli için sigmoid ve softmax

fonksiyonları tercih edilirken ve çok sınıflı sınıflandırma için genellikle softmax kullanılmaktadır.

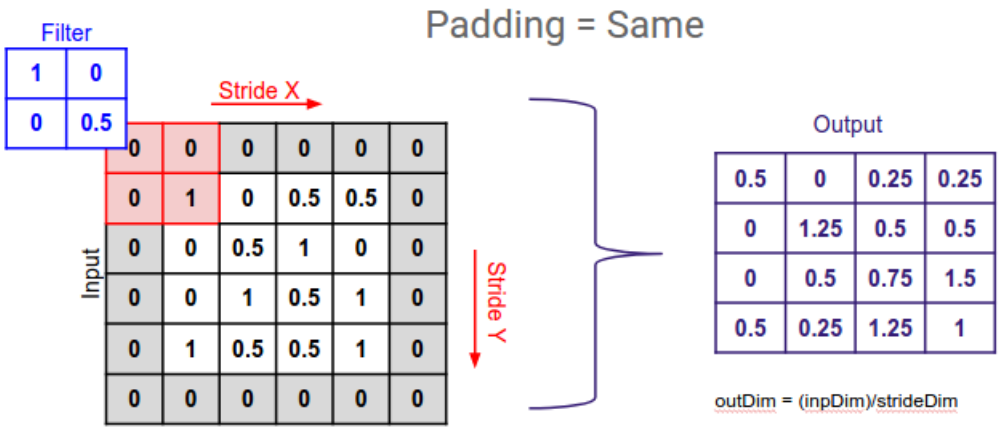
CNN'ler genellikle bilgisayarlı görme kullanılmaktadır, ancak son zamanlarda çeşitli NLP pronlemlerine uygulanmışlardır ve sonuçlar umut vericidir.
CNN'nin genel mimarisi şekil 3.11 'de gösterilmektedir.



Şekil 3.12 CNN ağı genel mimarisi

- **Doldurma (Padding)**

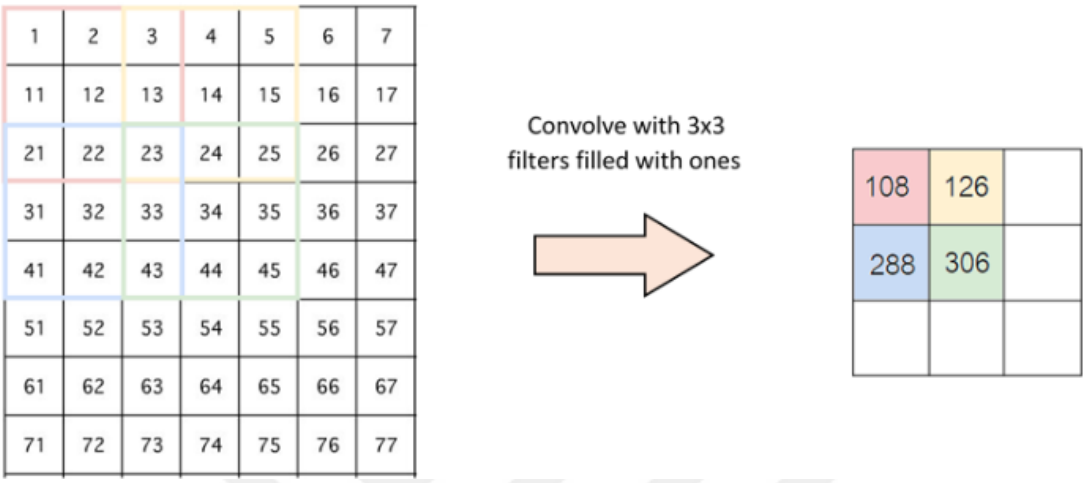
Doldurma, bir CNN'nin çekirdeği tarafından işlenirken bir görüntüye eklenen piksel miktarını ifade ettiğinden, evrişimli sinir ağlarıyla ilgili bir terimdir. Örneğin, bir CNN'deki dolgu sıfıra ayarlanırsa, eklenen her piksel değeri sıfır değerinde olacaktır. Bununla birlikte, sıfır dolgu bire ayarlanırsa, görüntüye sıfır piksel değerine sahip bir piksel kenarlığı eklenmektedir.



Şekil 3.13 CNN' de padding işlemi

- **Adımlar (Strides)**

Adım, giriş matrisi üzerindeki piksel kaymalarının sayısıdır. Örneğin, Adım 1 olduğunda, filtreleri bir seferde 1 piksele taşınmaktadır. Adım 2 olduğunda, filtreleri bir seferde 2 piksele taşınmaktadır. Aşağıdaki şekil, evrişimin 2 adımla çalışacağını göstermektedir.



Şekil 3.14 CNN'de stride işlemi

3.3.4.3.Tekrarlayan Sinir Ağları (RNN)

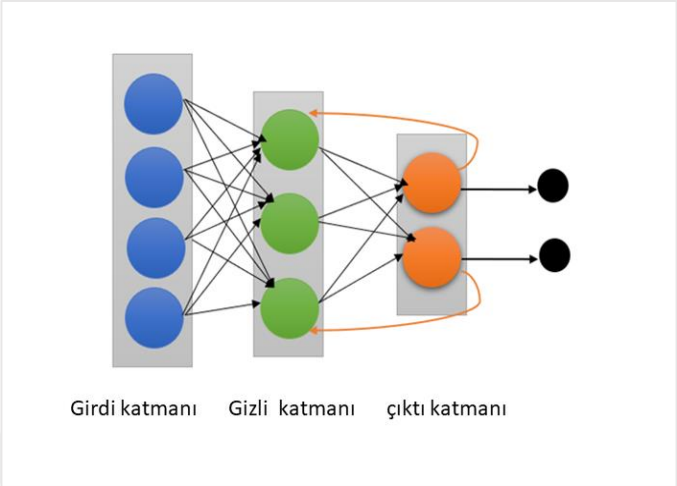
Tekrarlayan sinir ağları, önceki hesaplamaların sonuçlarını ezberleme ve bu bilgiyi mevcut hesaplamada kullanma kapasitesine sahip bir sinir ağı sınıfıdır. Başka bir deyişle, RNN'ler, gizli durumlara sahipken önceki çıktıların girdi olarak kullanılmasına izin vermektedir. Önceki özellikler, girdinin uygun bir bileşimini oluşturmak için RNN modellerini keyfi uzunluktaki girdilerdeki bağlam bağımlılıklarını modellemeye uygun hale getirmektedir.

RNN'ler, Hochreiter & Schmidhuber (1997) tarafından tanıtılmışlar ve diğer çalışmalarda birçok kişi tarafından rafine edilmiş ve popüler hale getirilmiştir.

RNN'ler, dil çevirisi, doğal dil işleme, konuşma tanıma ve resim yazısı gibi sıralı veya zamansal sorunlar için yaygın olarak kullanılmaktadır.
RNN'ler, mevcut girdi ve çıktıyı etkilemek için önceki girdilerden bilgi aldıkları için "hafızaları" ile ayırt edilmektedir.

RNN'ler geleneksel derin sinir ağlarından farklıdır. Geleneksel derin sinir ağları girdi ve çıktıların birbirinden bağımsız olduğunu varsayarken, tekrarlayan sinir ağlarının çıktısı dizi içindeki önceki öğelere bağlıdır.

Tekrarlayan ağların bir diğer ayırt edici özelliği, ağın her katmanında parametreleri paylaşmalarıdır. İleri beslemeli ağlar her düğümde farklı ağırlıklara sahipken, tekrarlayan sinir ağları ağın her katmanında aynı ağırlık parametresini paylaşmaktadır.



Şekil 3.15 RNN ağı genel mimarisi

- **Gradient Descent (Gradyan Azalma)**

Gradient Descent, makine öğrenimindeki en ünlü algoritmalardan biridir. Bu algoritmanın temel işlevi, sistemin ince ayarlı değeri belirlemesine yardımcı olan çok boyutlu hata veya maliyet fonksiyonunun yerel düşük noktasına veya yerel minimumuna yakınlaşarak "boyutluluğun laneti" probleminden kaçınma yeteneğidir veya ağırlık, ağdaki birimlerin her birine atamak ve doğruluğu tekrar hizaya getirmektir.

Basit RNN'ler, önceki katmanlardaki parametreleri öğrenmeyi ve ayarlamayı zorlaştıran gradyanların yok olması/kaybolması (vanishing gradient) probleminden muzdariptir.

RNN'lerin ana dezavantajlarından biri, gradyanların kaybolması problemidir. Gradyanların kaybolması problemi, gradyan tabanlı öğrenme yöntemleri ve geri yayılım ile yapay sinir ağlarının eğitilmesi sırasında karşılaşılmaktadır. Bu tür yöntemlerde, sinir ağının ağırlıklarının her biri, her eğitim yinelemesinde mevcut ağırlığa göre hata fonksiyonunun kısmi türeviyle orantılı bir güncelleme almaktadır. Ve bu güncelleme değerinin kayda değer büyüklükte olmama durumu gradyan yok olma olarak anılır.

Gradyanların kaybolması sorunu, uzun kısa süreli bellek (veya LSTM), artık ağlar (ResNet) ve kapılı tekrarlayan ağlar (GRU) gibi yeni yaklaşımlar önerilerek çözülmüştür.

- **Uzun Kısa Süreli Bellek (LSTM)**

LSTM'ler, uzun vadeli bağımlılıkları öğrenebilen ve ezberleyebilen bir Tekrarlayan Sinir Ağı türüdür. LSTM birimi bir hücre durumu, bir giriş kapısı, bir çıktı kapısı ve bir unutma kapısından oluşmaktadır. Hücre durumu, dizinin işlenmesi boyunca ilgili bilgileri taşıyabilmektedir. Kapılar, hücre durumunda hangi bilgilere izin verildiğine karar veren farklı sinir ağlarıdır.

Kapılar, eğitim sırasında hangi bilgilerin saklanması veya unutulması gerektiğini öğrenebilmektedir. Önceki durumun alakasız kısımlarını unutarak çalışmaktadır, ardından hücre durumu değerlerini seçici olarak günceller ve son olarak, hücre durumunun belirli bölümlerinin çıktısını almaktadır.

Kapılar, daha önce bahsettiğimiz 0 ile 1 arasındaki değerleri sıkıştıran sigmoid aktivasyon fonksiyonları içermektedir. Bu, verileri güncellemek veya unutmak için yararlıdır çünkü 0 ile çarpılan herhangi bir sayı 0'dır, bu da değerlerin kaybolmasına veya unutulmasına neden olmaktadır. Öte yandan, 1 ile çarpılan herhangi bir sayı aynı değerdir, bu nedenle bu değer aynı kalmaktadır veya tutulmaktadır.

a. Unutma kapısı (Forget Gate)

Bu kapı, hangi bilgilerin atılacağına veya saklanacağına karar vermektedir. Önceki gizli durumdan gelen bilgiler ve mevcut girişten gelen bilgiler sigmoid işlevinden geçirilmektedir. 0 ile 1 arasında değerler çıkmaktadır. 0'a yaklaştıkça unutmak, 1'e yaklaştıkça tutmak anlamına gelmektedir.

b. Girdi Kapısı (Input Gate)

Girdi kapısı, hücre durumunu güncellemek için kullanılmaktadır. Önceki gizli durum ve mevcut giriş, 0 ile 1 arasında olacak değerlere dönüştürülerek hangi değerlerin güncelleneceğine karar vermek için bir sigmoid fonksiyonuna geçirilmektedir. 1 önemli ve 0 önemli değil anlamına gelmektedir.

Gizli durum ve şimdiki girişi, ağı düzenlemeye yardımcı olmak için -1 ile 1 arasındaki değerleri sıkıştırmak için tanh işlevine de iletilmektedir. Daha sonra tanh çıktısı sigmoid çıktı ile çarpılmaktadır. Sigmoid çıktısı, tanh çıktısından hangi bilgilerin saklanmasının önemli olduğuna karar vermektedir.

c. Çıktı kapısı (Output Gate)

Çıktı kapısı, bir sonraki gizli durumun ne olacağına karar vermektedir. Gizli durum tahminler için kullanılmaktadır. Önceki gizli durumu ve mevcut girdisi bir sigmoid fonksiyonuna geçirerek çalışmaktadır. Daha sonra yeni değiştirilen hücre durumu tanh işlevine geçirilmektedir. Gizli durumun hangi bilgileri taşıması gerektiğine karar vermek için tanh çıktısı sigmoid çıktı ile çarpılmaktadır. Daha sonra yeni hücre durumu ve yeni gizli, bir sonraki zaman adımına taşınmaktadır.

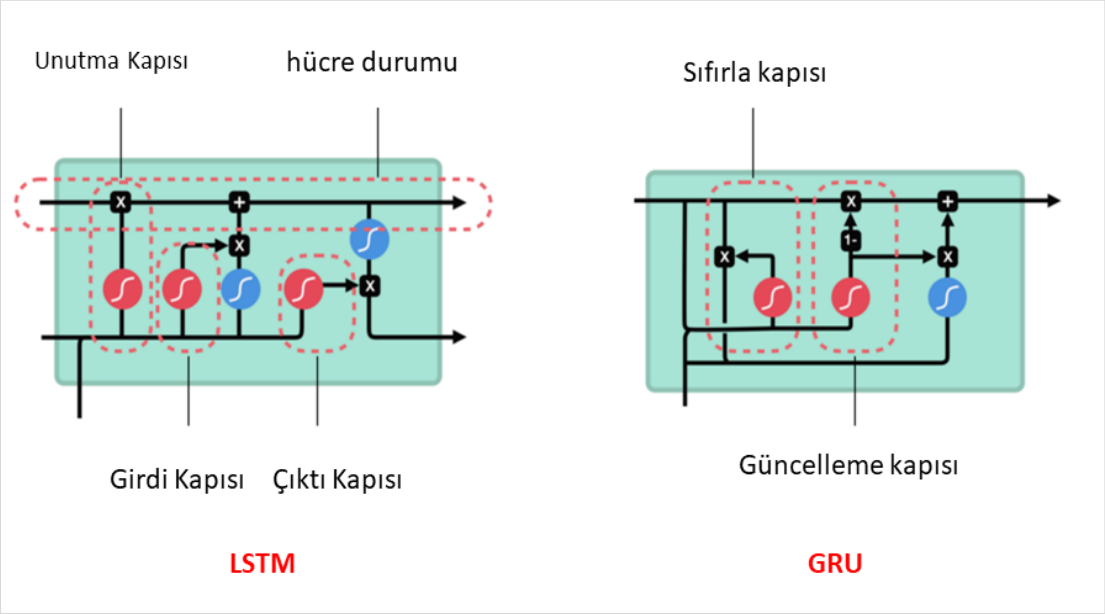
- **Kapılı Tekrarlayan Birimler (Gated Recurrent Unit /GRU):**

GRU, LSTM'ler gibidir, ve RNN modellerinin kısa süreli bellek sorununu çözmeye çalışmaktadır. Ancak, bilgiyi düzenleyen bir "hücre durumu" kullanmak yerine, gizli durumları kullanmaktadır. LSTM'lerde üç kapı yerine iki kapı, bir sıfırlama kapısı ve bir güncelleme kapısından oluşmaktadır. Sıfırlama ve güncelleme kapıları, ne kadar ve hangi bilgilerin tutulacağını kontrol etmektedir.

- **Çift Yönlü Uzun Kısa Süreli Bellek (Bidirectional Long Short Term Neural Network/BLSTM)**

LSTM'ler bağlamı yalnızca bir yönde işleyebildiklerindenbağlamın soldan sağa ve sağdan sola iki yönde okunabildiği çift yönlü LSTM olarak değiştirilmiştir.
Sonuç olarak, bu tür sinir ağı, son zamanlarda farklı NLP görevlerini ele almak için en gelişmiş yöntemlerde kullanılmıştır. Şekil 3.17, BLSTM mimarisinin blok diyagramını göstermektedir.
Şekilde $x = (x_1, x_2, \dots, x_n)$ girdi dizisi vektörünü n ise girdi dizisi uzunluğunu göstermektedir.
BLSTM ağı, giriş dizisinde iki LSTM eğitmektedir. İlki, olduğu gibi girdi dizisinde (ileri yön), ikincisi ise girdi dizisinin ters çevrilmiş bir kopyasında (geri yön).
çıktısı $y = (y_1, y_2, \dots, y_n)$, ileri $H_f^{\rightarrow} = (H_1^{\rightarrow}, H_2^{\rightarrow}, H_3^{\rightarrow}, \dots H_n^{\rightarrow})$

ve geri $H_b^{\leftarrow} = (H_1^{\leftarrow}, H_2^{\leftarrow} H_3^{\leftarrow}, .., H_n^{\leftarrow})$ gizli durumları bir araya getirilerek hesaplanmaktadır.



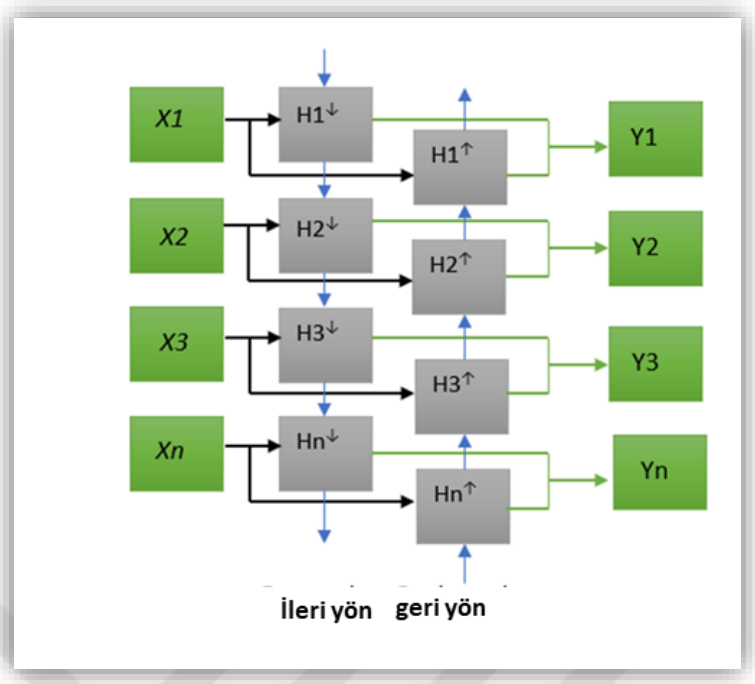
Şekil 3.16 LSTM ve GRU blok diyagramı

3.3.4.4. Üretken Çekişmeli Ağları (Generative Adversarial Networks /GAN)

GAN'lar, eğitim verilerine benzeyen yeni veri örnekleri oluşturan üretken derin öğrenme algoritmalarıdır. GAN 'nın iki bileşeni vardır: sahte veri üretmeyi öğrenen bir üreteci (jeneratör) ve üretecin sahte verisi ile gerçek örnek veriyi ayırt etmeyi öğrenen bir ayırıcıdır. İlk eğitim sırasında, jeneratör sahte veriler üretmektedir ve ayrımcı bunun yanlış olduğunu söylemeyi çabucak öğrenmektedir. GAN, modeli güncellemek için sonuçları üreticiye ve ayrımcıya göndermektedir.

3.3.4.5. Radyal Temelli Fonksiyon Ağları (Radial Basis Function Networks /RBFNs)

RBFN'ler, aktivasyon fonksiyonları olarak radyal temel fonksiyonları kullanan ileri beslemeli sinir ağlarının özel türleridir. Bir girdi katmanına, bir gizli katmana ve bir çıktı katmanına sahiptirler ve genellikle sınıflandırma görevleri için kullanılmaktadır. Girdinin eğitim setindeki örneklere benzerliğini ölçerek sınıflandırma yapmaktadır.



Şekil 3.17 BLSTM mimarisi

3.3.4.6. Çok Katmanlı Algılayıcılar (Multilayer Perceptrons /MLPs)

MLP'ler, aktivasyon fonksiyonlarına sahip çoklu algılayıcı katmanlarına sahip bir ileri beslemeli sinir ağları sınıfıdır. Tam bağlantılı bir giriş katmanı ve çıkış katmanından oluşmaktadır. MLP'ler, korelasyonu anlamak ve bir eğitim veri setinden bağımsız ve hedef değişkenler arasındaki bağımlılıkları öğrenmek için modeli eğitmektedir.

3.3.4.7. Kendi Kendini Düzenleyen Haritalar (Self Organizing Maps/SOM)

Kendi Kendini Düzenleyen Haritalar, bir modeldeki rastgele değişkenlerin sayısını azaltan denetimsiz verilerin yardımıyla çalışan bir tür derin öğrenme tekniğidir. SOM'lar önce her düğüm için ağırlıkları başlatarak ve eğitim verilerinden rastgele bir vektör seçerek çalışmaktadır. Ardından, hangi ağırlıkların en olası girdi vektörü olduğunu bulmak için her düğümü incelemektedir. Kazanan düğüm, En iyi eşleşen birim (BMU) olarak adlandırılmaktadır. Bundan sonra SOM'lar BMU'nun komşuları keşfetmekte ve örnek vektöre kazanan bir ağırlık vermektedir. Komşu BMU'dan ne kadar uzaktaysa, o kadar az öğrenmektedir.

3.3.4.8. Kısıtlı Boltzmann Makineleri (Restricted Boltzmann Machines /RBMs)

Kısıtlı Boltzmann Makineleri, bir dizi girdi üzerinden bir olasılık dağılımından öğrenebilen stokastik sinir ağlarıdır. Bu derin öğrenme algoritması, boyut azaltma, özellik öğrenme, işbirlikçi filtreleme, sınıflandırma, regresyon ve konu modelleme için kullanılmaktadır. Görünür birimler ve çıkış düğümleri olmayan Gizli birimler olmak üzere iki katmandan oluşmaktadır.

RBM'ler, tüm görünür birimlere ve gizli birimlere bağlı bir önyargı birimine sahiptir. Tüm düğümler, girdi veya gizli düğüm olmalarına bakılmaksızın diğer tüm düğümlere bağlıdır. Bu, kendi aralarında bilgi paylaşmalarına ve müteakip verileri kendi kendilerine oluşturmalarına olanak tanımaktadır.

3.3.4.9. Otomatik Kodlayıcırlar (Auto Encoders)

Otomatik kodlayıcılar, giriş ve çıkışın aynı olduğu belirli bir ileri beslemeli sinir ağı türüdür. Üç ana bileşenden oluşmaktadır: kodlayıcı, kod ve kod çözücü ve görüntü işleme alanında yaygın olarak kullanılmaktadır. Seyrek, Gürültü Giderme, Büzülme ve Yığılmış gibi çeşitli otomatik kodlayıcı türleri vardır.

4. YÜRÜTÜLEN ÇALIŞMALAR

4.1. Kullanılan Araçlar ve Kütüphaneler

4.1.1. Python

Python, yaygın olarak kullanılan genel amaçlı, üst düzey bir programlama dilidir. 1991 yılında Guido van Rossum tarafından oluşturulmuştur ve Python Software Foundation tarafından geliştirilmektedir. Python kod okunabilirliği vurgulanarak tasarlanmıştır ve sözdizimi programcıların kavramlarını daha az kod satırında ifade etmelerine olanak tanımaktadır.

Python güçlü, esnek ve kullanımı kolay bir dil olduğu için dünyadaki en popüler ve en hızlı büyüyen diller arasında yer almaktadır. Çoklu programlama paradigmalarını desteklediği için birçok organizasyonda kullanılmaktadır. Ayrıca, otomatik bellek yönetimi gerçekleştirmesi Python'un en önemli özelliklerinden biridir. Python'nun avantajlarından bazıları:

- ✓ **Avantajlar:**
- Kapsamlı desteklenen kütüphaneleri,
- Açık kaynak ve topluluk geliştirmesidir,
- Çok yönlü, okuması, öğrenmesi ve yazması kolaydır,
- Kullanıcı dostu veri yapıları,
- Üst düzey dildir,
- Dinamik olarak yazılan dildir,
- Nesne yönelimli dildir,
- Taşınabilmesi ve etkileşimlidir,
- İşletim sistemleri arasında taşınabilmektedir.

4.1.2. Keras

Derin öğrenme, Theano, TensorFlow, Caffe, Mxnet vb. çeşitli kütüphaneler tarafından desteklenmektedir.

Keras, derin öğrenme modelleri oluşturmak için TensorFlow, Theano gibi popüler derin öğrenme kütüphanelerinin üzerine inşa edilmiş en güçlü ve kullanımı kolay Python kütüphanelerinden biridir.

Keras, TensorFlow, Theano veya Cognitive Toolkit (CNTK) gibi açık kaynaklı makine öğrenmesi kütüphanelerinin üzerinde çalışmaktadır. Theano, hızlı sayısal hesaplama görevleri için kullanılan bir Python kütüphanesidir. TensorFlow, sinir ağları ve derin öğrenme modelleri oluşturmak için kullanılan en ünlü sembolik matematik kütüphanesidir.
CNTK, Microsoft tarafından geliştirilen derin öğrenme çerçevesidir. Python, C#, C++ veya bağımsız makine öğrenme araç setleri gibi kütüphanelerini kullanmaktadır. Theano ve TensorFlow çok güçlü kütüphanelerdir ancak sinir ağları oluşturmak için anlaşılması zordur.

Keras, derin öğrenme uygulamaları için onu en uygun seçim haline getiren derin öğrenme modellerini hızlı bir şekilde tanımlamak üzere tasarlanmıştır.
Keras, üst düzey sinir ağı API'ni daha kolay ve daha performanslı hale getirmek için çeşitli optimizasyon tekniklerinden yararlanmaktadır.

Keras aşağıdaki özellikleri desteklemektedir:

- Tutarlı, basit ve genişletilebilir uygulama programlama ara yüzüdür,
- Minimal yapı - herhangi bir ayrıntı olmadan sonuca ulaşmak kolaydır,
- Çoklu platformları ve arka uçları desteklemektedir,
- Hem CPU hem de GPU üzerinde çalışan kullanıcı dostu bir çerçevedir,
- Hesaplamanın yüksek düzeyde ölçeklenebilirliği.

Keras'ın başlıca avantajları şunlardır:

- Daha büyük topluluk desteği,
- Test edilmesi kolaydır,
- Yapay sinir ağları, kullanımı daha kolay hale getiren Python ile yazılmıştır,
- Keras hem evrişimi hem de tekrarlayan ağları desteklemektedir,
- Derin öğrenme modelleri ayrı bileşenlerdir, bu nedenle birçok şekilde birleştirilebilmektedir.

Keras, aynı zamanda, son teknoloji araştırma fikirlerini yinelemeye uygun, oldukça esnek bir kütüphanedir.

4.1.2.1. Keras Mimarisi

Keras API, Model, Katman ve Çekirdek Modüller olmak üzere üç ana kategoriye ayrılabilmektedir. Keras'ın temel veri yapıları, katmanlarından ve modellerinden

oluşmaktadır. Bir "katman", basit bir girdi-çıktı dönüşümü anlamına gelmektedir. Bir "model", yönlendirilmiş bir döngüsel olmayan katman grafiğidir.

Önemli Keras katmanlarından bazıları, Çekirdek Katmanlar, Evrişim Katmanları, Havuz Katmanları ve Tekrarlayan Katmanlardır.

Keras Modelleri iki tipe ayrılmaktadır. Temelde Keras Katmanlarının doğrusal bir bileşimi olan Sıralı Model (Sequential Model) ve temelde karmaşık modeller oluşturmak için kullanılan İşlevsel API (Functional API).

Ayrıca, Keras modelini ve Keras katmanlarını düzgün bir şekilde oluşturmak için birçok yerleşik sinir ağı ile ilgili işlevleri Keras tarafından sağlamaktadır. Fonksiyonlardan bazıları aşağıdaki gibidir:

- **Aktivasyon Modülü**: Yapay sinir ağlarında aktivasyon fonksiyonu önemli bir kavramdır. Keras aktivasyon modülleri, softmax, relu vb. gibi birçok aktivasyon fonksiyonunu içermektedir,
- **Kayıp Modülü:** Kayıp modülü, ortalama kare hata, ortalama mutlak hata, poisson, vb. gibi kayıp fonksiyonları sunmaktadır,
- **Optimizer Modülü:** Optimizer modülü, Adam, SGD, vb. gibi optimizasyon algoritmaları sağlamaktadır,
- **Düzenleyiciler (Regularizers) :** Düzenleyici modülü, L1 düzenleyici, L2 düzenleyici vb. gibi fonksiyonları sağlamaktadır.

4.1.3. TensorFlow 2

TensorFlow 2, uçtan uca, açık kaynaklı bir derin öğrenme çerçevesidir. TensorFlow 2, kapsamlı, esnek ve kütüphaneler ve topluluk kaynakları ekosistemine sahiptir; bu özellikleri nedeniyle, araştırmacıların makine öğrenmesinde en son teknolojiyi kullanmalarına ve geliştiricilerin makine öğrenmesi destekli uygulamaları kolayca oluşturup dağıtmalarına olanak tanımaktadır. TensorFlow 2 aşağıdaki temel yeteneklere sahiptir:

1) CPU, GPU veya TPU üzerinde düşük seviyeli tensör işlemlerini verimli bir şekilde yürütmesi,
2) Rasgele türevlenebilir ifadelerin gradyanını hesaplaması,
3) Yüzlerce GPU'dan oluşan kümeler gibi birçok cihaza ölçekleme hesaplaması,

4) Programları, sunucular, tarayıcılar, mobil ve gömülü cihazlar gibi harici çalıştırma ortamlarına aktarması.

4.1.4. Google Colab

Google Colab veya "Colaboratory", kullanıcıların yapılandırma yapmadan, GPU'lara ücretsiz erişim ve Kolay paylaşım yeteneği ile tarayıcılarında Python kodu yazmasına ve yürütmesine olanak tanımaktadır.

Ayrıca Google Colab verileri analiz etmek ve görselleştirmek için popüler Python kütüphanelerinin tüm gücünden yararlanmaya olanak tanımaktadır.

Colab notebooks, Google'ın bulut sunucularında kod yürütmektedir; bu, kullanıcının makinesinin gücünden bağımsız olarak GPU'lar ve TPU'lar dâhil olmak üzere Google donanımının gücünden yararlanabileceği anlamına gelmektedir.

4.2. Tez Çalışmasının Katkıları

Kısa metin sınıflandırma problemi olarak kabul edilen olan sosyal ağlarda spam tespiti, metnin seyrekliği ve belirsizliği nedeniyle doğal dil işlemede zorlu bir görevdir. Bu sorunu çözmenin en önemli görevlerinden biri güçlü bir metin gösterimi kullanmaktır. Geleneksel kelime gömme modelleri, yoğun vektörlerle kelimeleri temsil ederek veri seyrekliği problemini çözmektedir, ancak bu modellerin bazı problemleri etkili bir şekilde ele almalarını engelleyen bazı sınırlamaları vardır.
Modelin sözlüğünde bulunmayan kelimeleri için herhangi bir vektör temsili sağlayamaması ve "out of vocabulary" problemi olarak bilinen bu yöntemin en yaygın sınırlamalardan biridir.
Bu modellerin karşılaştığı bir diğer problem, modelin, cümle içindeki konumundan bağımsız olarak kelimelerin her birini için yalnızca bir tane vektör vermesidir (bağlamdan bağımsızlık). Bu problemlerin üstesinden gelmek için bu tez dört ana çalışmayı tanıtmaktadır.

1) İlk Çalışma (BERT Tabanlı Yaklaşım)

İlk çalışmada, BERT modelini kullanarak metnin bağlama bağlı temsiline dayanan yeni bir mimari önerilmiştir. Önerilen model, Twitter veri kümesi kullanılarak test edilmiştir. Deneysel sonuçlar, önerilen yöntemin geleneksel ağırlıklandırma

yöntemlerinden, geleneksel kelime gömme tabanlı yöntemlerden ek olarak Twitter platformunda spam tespiti alanında kullanılan mevcut son teknoloji yöntemlerden daha iyi performans gösterdiğini göstermektedir.

2) İkinci Çalışma (CBLSTM /ELMO-BLSTM Tabanlı Yaklaşım)

İkinci çalışmada, derin bağlamsal kelime temsili tabanlı yeni bir model önerilmiştir. Çalışmada, ELMO iyi bilinen modeli, metni bağlamsal temsili amacıyla önerilen çift yönlü uzun kısa vadeli sinir ağının gömme katmanında kullanılmıştır. Önerilen model, CBLSTM (Contextualized Bi-directional Long Short Term Memory neural network) olarak adlandırılmıştır. Önerilen modelde kullanılan bilişenlerin etkisi test edip her bilişenin en iyi sonuçları verebilecek değeri bulmak için bir analiz çalışması gerçekleştirilmiştir. Üç kıyaslama veri kümesi kullanarak elde edilen deneysel sonuçları, önerilen yöntemin yüksek doğruluk elde ettiğini ve sosyal ağlarda spam tespiti konusunda mevcut son teknoloji yöntemlerden daha iyi performans gösterdiğini göstermektedir.

3) Üçüncü Çalışma (RoBERTa-BLSTM /Transformer Tabanlı Yaklaşımlar)

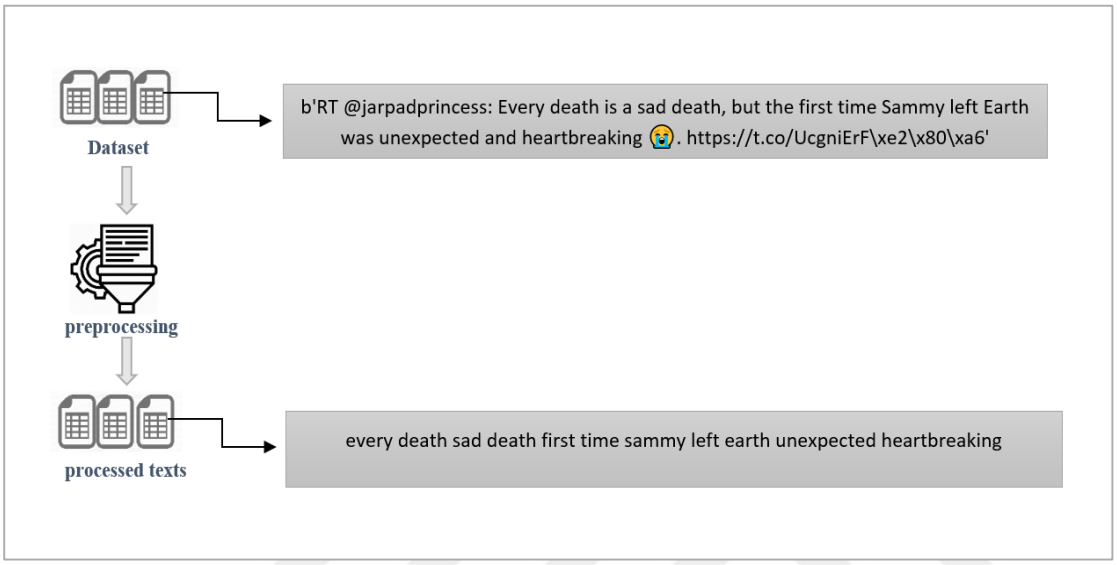
Üçüncü çalışma kapsamında iki alt çalışma gerçekleştirilmiştir. Birinci alt çalışma, Twitter platformunda spam tespiti için RoBERTa tabanlı çift yönlü tekrarlayan sinir ağ modeli tanıtmaktadır. RoBERTa modeli, yığılmış BLSTM ağının performansını iyileştirmek için bağlamsal kelime temsillerini öğrenmek için kullanılmaktadır. Ayrıca, spam tespit probleminin çözümü açısından en yaygın transformer tabanlı modelleri değerlendirmek için karşılaştırmalı bir çalışma yapılmıştır. İkinci alt çalışmada, bir önceki çalışmada önerilen modeli değerlendirmek için bir analiz çalışması yapılmış ve model daha önce bahsedilen üç kıyaslama veri seti kullanılarak test edilmiştir. Kullanılan veri setindeki deneysel sonuçlar, RoBERTa-BLSTM modelin sosyal ağlardaki spam tespiti için kullanılan tüm yaygın modellerden daha iyi performans gösterdiğini belirtmektedir.

4) Dördüncü Çalışma (ALBERT4Spam/ALBERT Tabanlı Yaklaşım)

Dördüncü çalışmada, ALBERT tabanlı bir derin öğrenme modeli olan, ALBERT4Spam olarak adlandırılan ve sosyal ağlarda spam tespiti amaçlayan bir modeli tanıtılmaktadır. Çalışmada önerilen RNN tabanlı derin sinir ağ mimarisinin performansını iyileştirmek için, ALBERT iyi bilinen model, kelime temsillerini

öğrenmek amacıyla kullanılmıştır. Önerilen ALBERT4Spam modelinde kullanılan nöron sayısı, katman sayısı, aktivasyon fonksiyonu, ağırlık başlatıcı, öğrenme oranı, optimize edici ve bırakma gibi birden çok hiper parametreler, modelin en iyi performansına ulaşmak için rastgele arama algoritması kullanılarak ince ayar (Hyperparameter fine-tuning) yapılmıştır.

Kullanılan üç veri kümesi üzerindeki deneysel sonuçları, önerilen modelimizin sosyal



Şekil 4.1 bir tweet'in ön işlemesine bir örnek

ağlardaki spam tespiti için kullanılan tüm yaygın modellerden üstün sonuçlarla daha iyi performans gösterdiğini belirtmektedir.

4.2.1. Kullanılan Veri Kümeleri

Bu çalışmada üç kıyaslama veri kümesi kullanılmıştır. Bunlardan ilki Chen ve diğerleri tarafından (Chen, Yeo vd. 2015) yapılan çalışma kapsamında yayınlanan Twitter veri kümesidir. Orijinal veri kümesi 100000 tweet ID ve etiketlerini içermektedir, bu nedenle, tweet bilgilerini toplamak için tweet kimliğine dayalı Python kod tabanlı bir tarayıcı geliştirip uygulanmıştır. Bazı tweetlerin Twitter veya kullanıcılar tarafından silinmesi nedeniyle sadece 58.159 tweet'e ulaşılmıştır. Daha sonra veri kümesi dengesiz olduğundan, sınıflandırıcının performansını iyileştirmek için çıkarılan veri kümesine veri büyütme uygulanmıştır. Ardından veri setinin, %80'i eğitim için ve %20'si ise test için ayrıştırılmıştır. İkinci veri kümesi, Alberto ve diğerleri tarafından (Alberto, Lochter vd. 2015) yapılan makalesinde kullanılan YouTube yorum veri setidir. Orijinal veri kümesi, spam ve spam olmayan mesajları dâhil olmak üzere beş grup YouTube yorumu içermektedir. Son olarak, üçüncü veri kümesi, UCI repository

da bulunan SMS spam veri kümesidir. Bahse geçen veri kümelerin açıklaması Çizelge 4.1 'de sunulmuştur.

Çizelge 4.1 kullanılan veri kümeleri

Veri kümesi	# Spam	# spam olmayan	Toplam
Twitter	38205	27822	66027
YouTube	941	935	1876
SMS spam	747	4827	5574

4.2.2. Veri Önişleme

Veri önişleme, metin sınıflandırma probleminde önemli bir rol oynamaktadır. Sosyal medyadan çıkarılan kısa metinlerin boşluklar, noktalama işaretleri, özel karakterleri, URL'ler vb. içerebilmektedir. Dolayısıyla, sınıflandırma algoritmalarının daha iyi performans gösterebilmesi için metni daha sindirilebilir bir forma dönüştürülmelidir. Bu amaçla, Python kodu kullanılarak bağlantıları (URL) kaldırma, noktalama işaretlerini kaldırma, emoji ve özel karakterleri kaldırma, durak kelimelerin çıkarılma ve harflerin tamamının küçük harflere dönüştürülme gibi farklı önişlemleri gerçekleştirilmiştir. Şekil 4.1, bir tweet'in ön işlemesinin grafiksel bir örneği göstermektedir.

4.2.3. Özellik Çıkarma

Özellik çıkarma, orijinal olanlardan yeni özellikler oluşturarak bir veri kümesindeki özellik sayısını azaltmaktadır. Bu aşamada genel hesaplama maliyetini veya eğitim süresini azaltmak ve modelin doğruluğunu artırmak için mevcut özellikleri daha düşük boyutlu uzaya dönüştürülmektedir.

Bu çalışmada, sosyal spam mesajları sınıflandırma görevinin performansı üzerinde kelime temsilinin önemini göstermek için birkaç özellik çıkarma yöntemi denenmiştir. İlk olarak, Count Vctorizer ve TF-IDF gibi geleneksel BoW tabanlı ağırlıklandırma yöntemlerini test edilmiştir, ardından word2vec, Glove ve FastText gibi en popüler geleneksel Word embedding yöntemleri kullanılmıştır. Son olarak, özellik çıkarma aşamasında ELMo, BERT, DistilBERT, RoBERTa, DistilRoBERTa, ELECTRA ve ALBERT gibi bağlamsal önceden eğitilmiş modeller kullanılmıştır.

4.3. Önerilen Modeller

4.3.1. İlk Çalışmada Önerilen Model

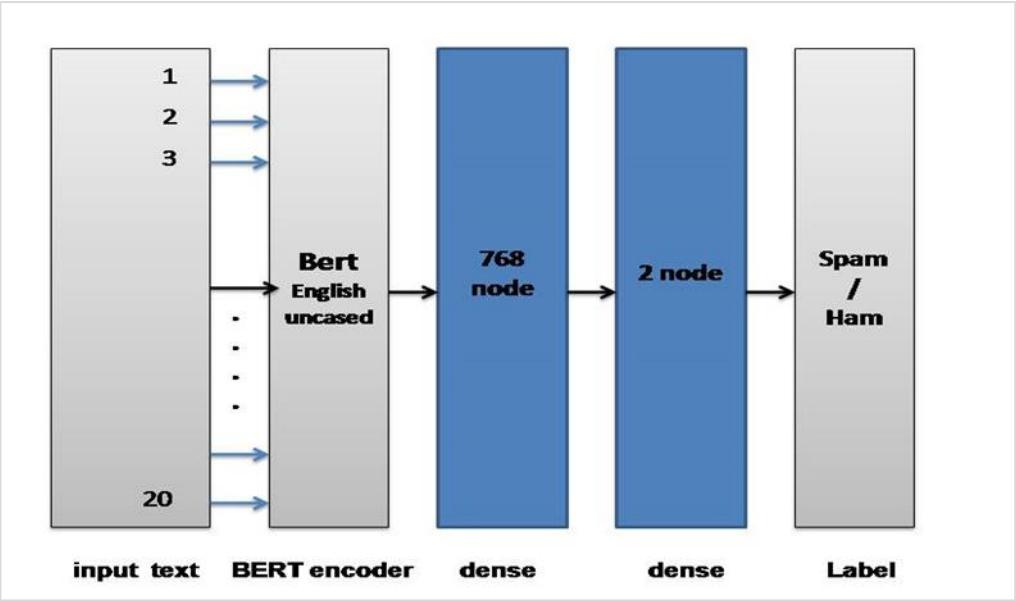
İlk çalışmada, metin temsili için BERT modeline dayalı yeni bir mimari önerilmiştir. Şekil 4.2'de gösterildiği gibi önerilen mimari beş katmandan oluşmaktadır, onlar, giriş katmanı, 20 tokena eşit bir dizi uzunluğuna sahip BERT kodlayıcı katmanı, ardından 768 düğümlü ve Relu aktivasyon fonksiyonlu bir gizli katmanı ve son olarak metnin spam olup olmadığını sınıflandırmak amacıyla kullanılan ve sigmoid aktivasyon fonksiyonuna sahip olan bir çıktı katmanıdır. Çizelge 4.2, önerilen modelde kullanılan hiper parametreleri göstermektedir.

Bu çalışmada, önerilen yöntem ile karşılaştırmak amacıyla, metin sınıflandırma problemlerinde kullanılan en yaygın makine öğrenmenin sınıflandırıcılarından biri olan Naive Bayes (NB) sınıflandırıcısı ile CountVectorizer ve TF-IDF gibi en popüler BoW tabanlı ağırlıklandırma yöntemlerini denenmiştir.

Ayrıca, word2vec, Glove ve fastText gibi en yaygın kelime temsili yöntemleriyle üç farklı derin sinir ağı mimarisi uygulanmış ve denenmiştir. Evrişimsel sinir ağı (CNN), Uzun kısa süreli bellek sinir ağı (LSTM) ve çift yönlü uzun kısa vadeli sinir ağı (BLSTM) olmak üzere üç tane derin öğrenme mimarisi önerip uygulanmıştır. Çizelge 4.3, her mimaride kullanılan hiper parametreleri göstermektedir.

Çizelge 4.2 Birinci çalışmada önerilen modelde kullanılan parametreleri

Parametre	Değer
İyileştirici (Optimizer)	Adam
Aktivasyon fonksiyonu	RELU, Sigmoid
Dizi uzunluğu	20
Kayıp fonksiyonu	Binary cross entropy
Batch boyutu	64
No. Epochs	3
Öğrenme oranı (Learning rate)	0.000001
Optimizasyon metrikleri	Doğruluk,F1-skor, kesinlik, duyarlılık



Şekil 4.2 İlk çalışmada önerilen modelin blok diyagramı

Çizelge 4.3 Word embedding tabanlı modellerde kullanılan parametreleri

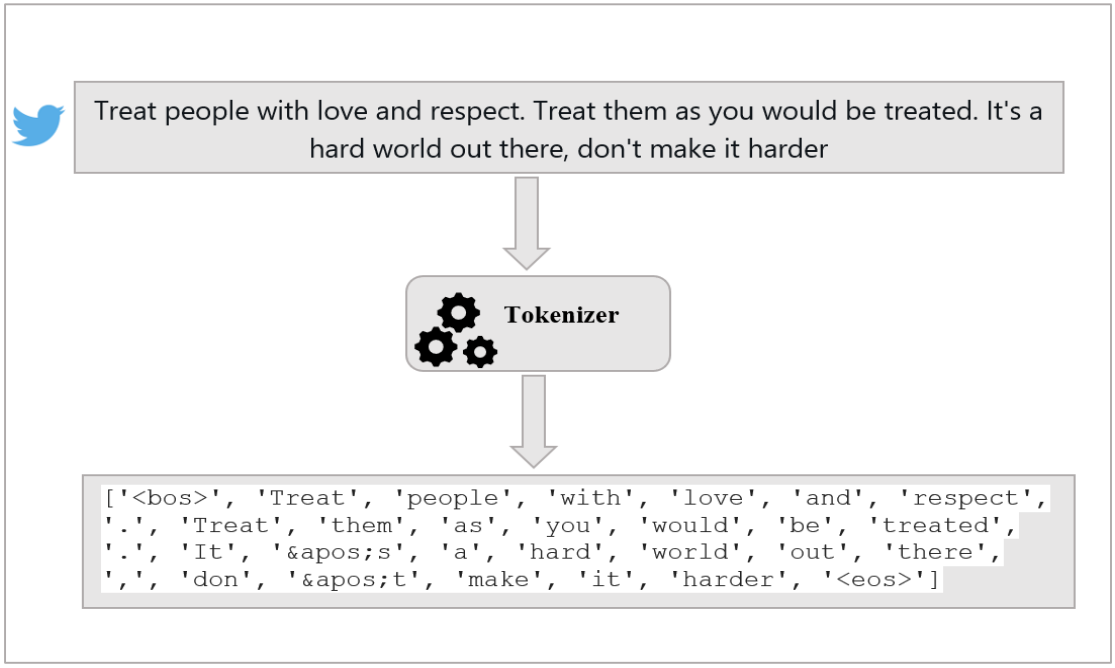
Parametre	CNN	LSTM	BLSTM
İyileştirici	Adam	Adam	Adam
Birim sayısı	-	300	300
Dropout	0.2	0.2	0.2
Recurrent Dropout	-	-	0.2
Filtre sayısı	32	-	-
Kernel boyutu	3	-	-
Pooling	Global max	-	-
Kayıp fonksiyonu	Binary cross entropy	Binary cross entropy	Binary cross entropy
Batch boyutu	32	64	64
Epoch	30	30	30
Optimizasyon metriği	Doğruluk	Doğruluk	Doğruluk

4.3.2. İkinci Çalışmada Önerilen Model

İkinci çalışmada, embedding katmanında metin bağlamsal temsili kullanan çift yönlü tekrarlayan sinir ağına dayalı yeni bir derin öğrenme mimarisi önerilmiştir. Önerilen model, CBLSTM (Contextualized Bi-directional Long Short Term Memory neural network) olarak adlandırılmıştır. Modeldeki her bir bileşeninin en iyi sonuçlar

verebilecek değeri bulabilmek için bir analiz çalışması yapılmıştır. Ayrıca, CBLSTM'in sonuçları karşılaştırmak için aynı veri setlerine geleneksel ağırlıklandırma yöntemlerine ek olarak GloVe ve Fast Text gibi geleneksel Word embedding yöntemleri uygulanıp test edilmiştir.

Önceden eğitilmiş ELMO modeli, metin verilerini derin bir bağlamsal formda temsil etmek için kullanılmıştır. ELMO modeldeki çıkışı, sosyal ağlardaki spam ve spam olmayan mesajları ayırt etmek için tekrarlayan sinir ağı tabanlı mimariye bir girdi olarak kullanılmıştır. Şekil 4.3, bir tweet tokenizasyon işlemi için bir örnek göstermektedir. Burada BOS'un bir cümlenin başlangıcı ve EOS'un bir cümlenin sonunu gösterdiğini belirtmekte fayda vardır.



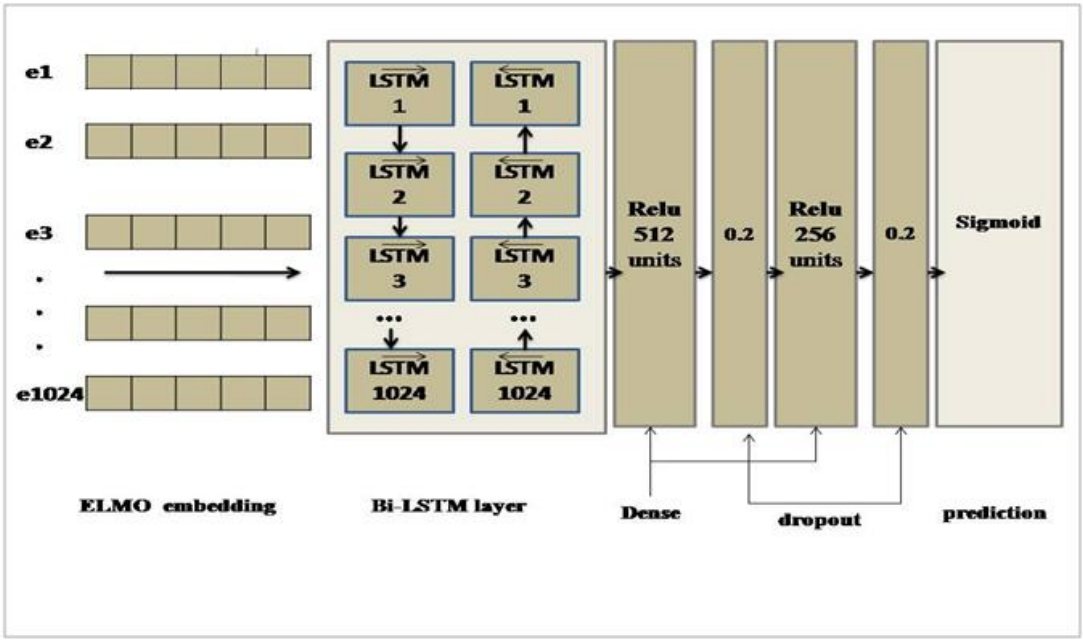
Şekil 4.3 ELMO belirteci kullanan bir tweetin tokenizasyon örneği

Önerilen CBLSTM mimarisi, bir analiz (ablation study) çalışması yapılarak deneysel olarak oluşturulmuştur.

Önerilen sinir ağı mimarisi, en iyi sonuçları elde etmek için katman sayısı ve kullanılan hiperparametreleri kademeli olarak değiştirilecek şekilde ampirik olarak oluşturulmuştur. Önerilen modelde çift yönlü bir LSTM mimarisi kullanılmış olsa da önerilen modelden elde edilen sonuçları karşılaştırmak amacıyla evrişimsel sinir mimarisi de test edilmiştir. Önerilen modelde, farklı bir bırakma ve yoğun katman seçimi ile değerlendirilmiş olup en iyi sonuçları verebilecek değerleri kullanılmıştır. Ayrıca, gizli katmanlarda ReLu aktivasyon fonksiyonu olarak, çıktı katmanında ise

Sigmoid aktivasyon fonksiyonu olarak kullanılmıştır. Söz konusu problemi ikili sınıflandırma problemi olduğundan ikili çapraz entropi kayıp fonksiyonu olarak kullanılmıştır.

Çizelge 4.4, Twitter veri seti kullanılarak yürütülen hiperparametre analiz çalışmasının bir kısmını göstermektedir. Şekil 4.4'te yer alan model mimarisine ulaşıldığında en iyi sonuçlar elde edilmiştir. Önerilen model, bir girdi katmanı, embedding katmanı, çift yönlü LSTM katmanı, iki yoğun katman, iki bırakma katmanı ve bir çıktı katmanından oluşmaktadır.



Şekil 4.4 ikinci çalışmada önerilen modelin blok diyagramı

Son olarak, mesajları spam ve spam olmayan mesajlar olarak sınıflandırmak için bir nöron ve sigmoid aktivasyon fonksiyonuna sahip bir çıktı katman kullanılmıştır. Sigmoid aktivasyon fonksiyonu S(x), çıkış değerlerini 0 ile 1 arasında yer almaktadır ve aşağıdaki formülle tanımlanmaktadır:

$$S(x) = 1/(1+e^{(-x)}) \tag{4.2}$$

Çizelge 4.4 İkinci çalışmada önerilen modeli oluşturmak için yürütülen analiz çalışmasının bir kısmı

Model	**Dropout değeri**	**Sınıflandırma doğruluğu (Twitter veri kümesi)**
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) /Dropout/Dense(256,Relu)/ Dropout/Dense output(sigmoid)	0.2	0.9672
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512,Relu)/Dropout / Dropout /Dense output(sigmoid)	0.2	0.9705
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512,Relu)/ Dropout / Dense output(sigmoid)	0.2	0.9641
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512, Relu)/ Dense(256,Relu)/ Dropout / Dense output(sigmoid)	0.2	0.9655
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512, Relu)/ Dropout /Dense(256,Relu)/ Dense output(sigmoid)	0.2	0.9509
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512, Relu)/ Dropout /Dense(256,Relu)/ Dropout /Dense output(sigmoid)	0.2	**0.9780**
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense (512, Relu)/ Dropout /Dense(256, Relu)/ Dropout /Dense output(sigmoid)	0.3	0.9698
Input/embedding/BLSTM (1024-unit, dropout & recurrent dropout=0.2) / Dense (512, Relu)/ Dropout /Dense (256, Relu)/ Dropout /Dense output(sigmoid)	0.4	0.9717
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense (512, Relu)/ Dropout /Dense(256, Relu)/ Dropout /Dense output(sigmoid)	0.5	0.9599
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense (512, Relu)/ Dropout /Dense(256, Relu)/ Dropout /Dense output(sigmoid)	0.6	0.9624
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense (512, Relu)/ Dropout /Dense(256,Relu)/ Dropout /Dense output(sigmoid)	0.7	0.9615
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512, Relu)/ Dropout /Dense(256,Relu)/ Dropout /Dense output(sigmoid)	0.8	0.9416
Input/embedding/BLSTM (1024 unit, dropout & recurrent dropout=0.2) / Dense(512, Relu)/ Dropout /Dense(256,Relu)/ Dropout /Dense output(sigmoid)	0.9	0.9594
Input/embedding/Conv1D((filters=32, kernel size=3,Relu)/ pooling (GlobalMaxPooling1D /Dense(512,Relu)/ Dropout /Dense(256,Relu)/ Dropout /Dense output(sigmoid)	0.2	0.9265

RNN ağları genellikle aşırı uyum probleminden muzdarip olduğundan, modelin muhtelif yerlerinde bırakma katmanları uygulanmıştır.

Çizelge 4.5, önerilen modelin eğitimi sırasında kullanılan hiper parametreleri sunmaktadır. Optimize edici olarak Adam'ın kullanıldığını, batch boyutunun 32'ye eşit olduğunu ve modelin 10 epoch eğitildiğini belirtmekte fayda var.

Çizelge 4.5 ikinci çalışmada önerilen modelde kullanılan parametreler

Parametre	**Değer**
İyileştirici	Adam
Kayıp fonksiyonu	Binary cross entropy
Batch boyutu	32
Epoch	10
Optimizasyon metriği	Doğruluk, kayıp
LSTM birimi sayısı	1024*2
Dropout	0.2
Aktivasyon fonksiyonu	Sigmoid

4.3.3. Üçüncü Çalışmada Önerilen Model

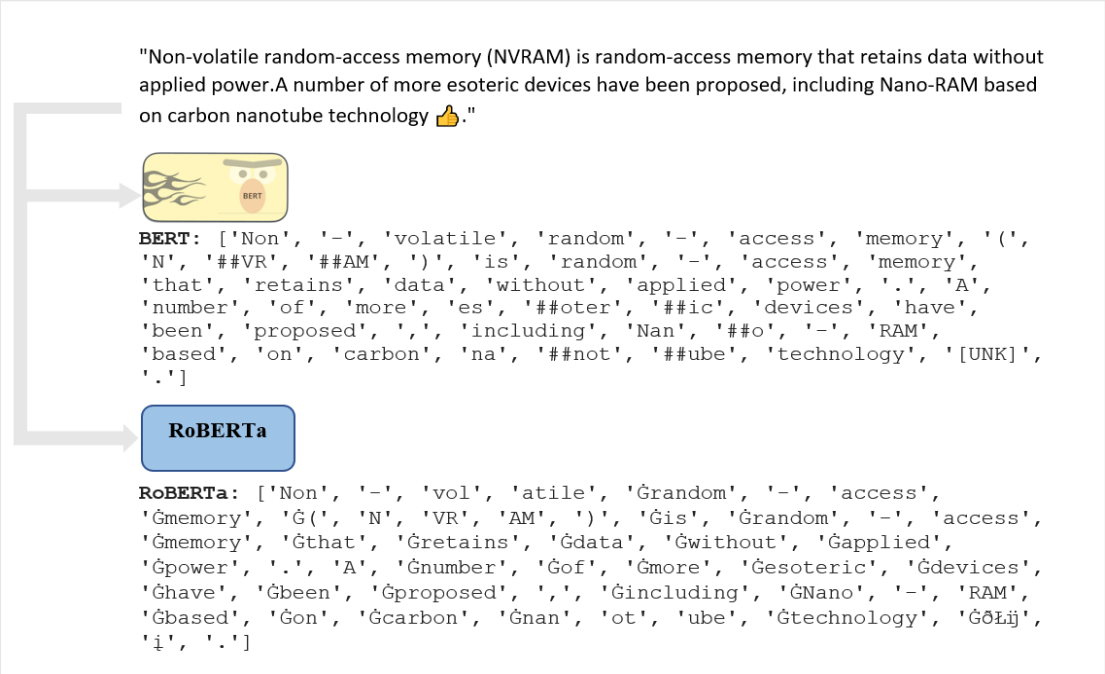
Üçüncü çalışmada, en popüler transformer tabanlı yöntemler ve bunların sosyal ağlarda spam tespiti alanındaki uygulamalarının karşılaştırmalı bir çalışması yapılmıştır. Ardından sosyal ağarlardaki spam tespit etmek için yeni bir hibrit derin sinir ağ modeli önerilmiştir. Daha fazla bağlam sağlamak için bir yığılmış çift yönlü uzun kısa vadeli sinir ağ mimarisi, RoBERTa önceden eğitilmiş modeli melezleşerek metin bağlamsal kelime temsili amacıyla kullanılmıştır. Önerilen model kullanılan üç kıyaslama veri seti kullanılarak eğitilip test edilmiştir.

RoBERTa modeli, Roberta tokenizasyon yaklaşımı nedeniyle metinden özellikler çıkarmak için bu çalışmada kullanılmıştır. RoBERTa modeli, bayt düzeyinde bayt çifti kodlaması (byte-level byte-pair-encoding) veya BPE tokenizasyon olarak adlandırılan yöntemi kullanmaktadır. Belirteç oluşturucu, dizeyi birden çok alt dizeye bölmektedir, daha sonra alt dizeler modelin sözlüğünde değilse, her dize kendi sözlüğüyle temsil edilene kadar tekrar bölünmektedir. Ayrıca, bayt düzeyinde bir yaklaşıma sahip olması ve her baytı ayrı ayrı işleme yeteneği nedeniyle belirteç oluşturucu, BERT gibi diğer tokenizasyon yaklaşımlarında kullanılan [UNK] belirtecini kullanmak yerine emoji gibi karmaşık karakterleri birden çok baytın bir kombinasyonu olarak

kodlayabilmektedir. Üstelik belirteç oluşturucu boşlukları, belirteçlerin parçaları gibi ele almak üzere eğitilmiştir. Bu nedenle bir kelime, cümlenin başında (boşluksuz) olsun veya olmasın farklı şekilde kodlanmaktadır.

Batch boyutu ve maksimum dizi uzunluğu 128'e eşit olarak seçilmiştir, bu nedenle veri setindeki metin 128 token veya daha az olacak şekilde kesilmiş olup önerilen sinir ağı mimarisine bir girdi olarak beslenmiştir.

Şekil 4.5, BERT ve RoBERTa tokenizasyonu arasındaki farkı göstermektedir.



Şekil 4.5 BERT ve RoBERTa tokenizasyon işlemi

Spam mesajları etkili bir şekilde ayırt etmek için yığılanmış çift yönlü uzun kısa süreli belleğe (LSTM) dayalı derin bir sinir ağı mimarisi kullanılmıştır. Yığılmış BLSTM'de, birinci BLSTM katmanının çıktısı, ikinci BLSTM katmanının girdisi olmaktadır ve ikinci katmanın çıktısı, bu katmanın ileri ve geri gizli durumları bir araya getirilerek oluşturulmaktadır. Böyle, ağa ek bağlam eklenebilmek ve sorunun daha hızlı öğrenilmesiyle sona erilebilmektedir. Önerilen modeli oluşturmak ve performansını değerlendirmek için, RoBERTa modelinden bağlamsal embedding alan girdi katmanıyla başlatılmıştır, ardından iki BLSTM katmanının, yoğunluk ve çıktı katmanları izlenmiştir. Ayrıca, farklı değerlere sahip bırakma katmanı test edilmiştir. Ondan sonra, üç veri kümesi kullanılarak test yaparken BLSTM katmanlarının sayısını kademeli olarak artırılmıştır. Şekil 4.6'de görülebileceği gibi, mesajların metin

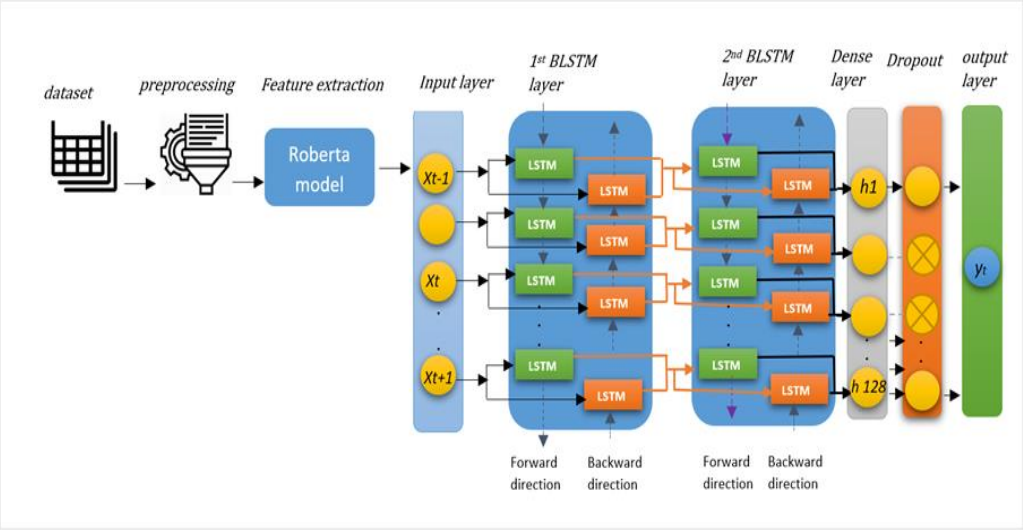
bağlamsal kelime temsili, RoBERTa modeli kullanılarak oluşturulmuştur. Bu aşamada, elde edilen sayısal girdi metin formu, derin sinir ağı mimarisi tarafından ele alınmaya hazır hale gelmektedir. Girdi metninin özellik vektörleri, daha sonra sinir ağının girdi katmanına girmek ve ek bağlam sağlamak için birinci BLSTM katmanına beslenmektedir. Birinci BLSTM katmanın çıktısı, daha fazla bağlam eklemek ve aynı zamanda sınıflandırmanın etkinliğini arttırmak ile beraber problemin öğrenilmesini hızlandırmak için kullanılan ikinci BLSTM katmanına bir girdi olarak kabul edilmiştir. Bundan sonra elde edilen çıktı aynı şekilde sonraki BLSTM katmanlarına iletilmektedir. Sonunda, BLSTM fazdan çıkan özellik vektörü, 128 nörondan oluşan ve ReLu Doğrusal aktivasyon fonksiyonunu kullanan yoğun katmana beslenmektedir. Sınıflandırma görevi, spam ve spam olmayan mesajları sınıflandırmak amacıyla kullanan, bir nörondan oluşan ve sigmoid aktivasyon fonksiyonuna sahip olan yoğun çıktı katmanı tarafından yapılmaktadır.

RNN ağlarının genellikle aşırı uyum probleminden muzdariptir. Dolaysıyla, aşırı uyum problemlerinden kaçınmak için önerilen modelin çıkış katmanının yanı sıra BLSTM katmanlarındaki bellek birimlerinin girişinden önce bırakma katmanları uygulanmıştır.

Çizelge 4.6, önerilen mimaride kullanılan parametreleri göstermektedir.

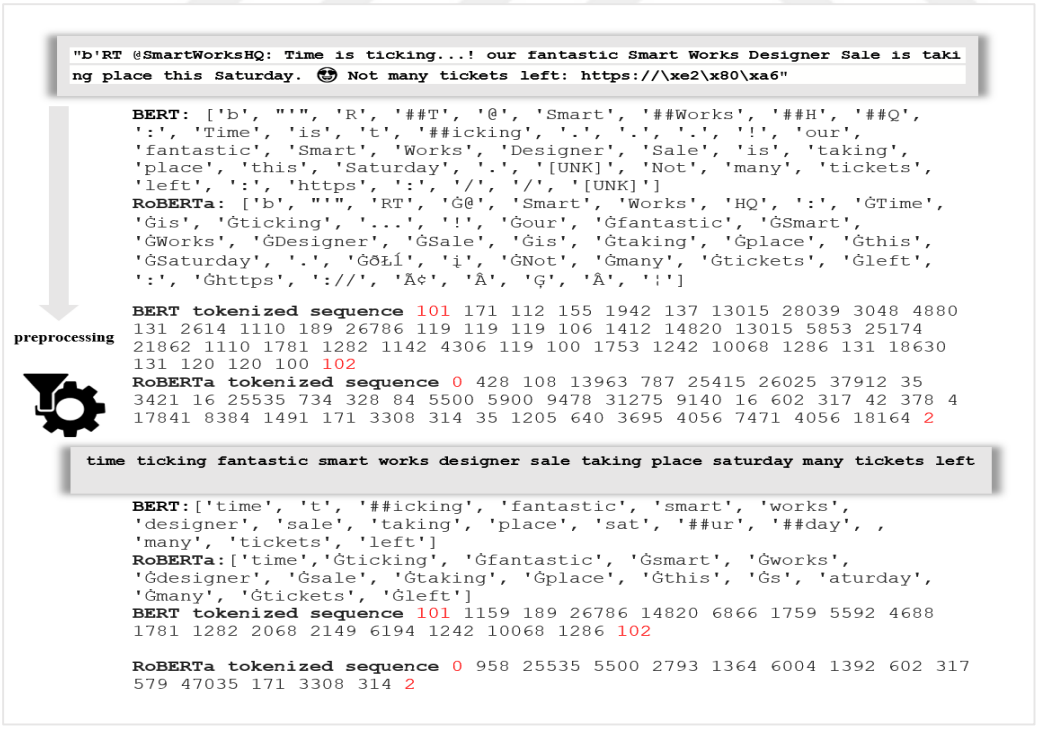
Çizelge 4.6 Üçüncü çalışmada önerilen modelde kullanılan parametreler

Parametre	Değer
İyileştirici	Adam
Aktivasyon fonksiyonu	Relu, Sigmoid
Kayıp fonksiyonu	Binary cross entropy
Learning rate	0.001
Her katmandaki LSTM birimi sayısı	128
Dropout	0.25
Batch boyutu	256
Maksimum dizi uzunluğu	128
Eğitim turu sayısı (epoch)	10
Optimizasyon metriği	Doğruluk, kayıp



Şekil 4.6 Üçüncü çalışmada önerilen modelin blok diyagramı

Metnin temsilini iyileştirmede ön işleme adımları çok büyük önem taşıdığından bahsetmekte fayda vardır, dolayısıyla, Şekil 4.7 ön işlemeli ve ön işlemesiz tokenizasyon süreci arasındaki farkı göstermektedir.



Şekil 4.7 Ön İşlemeli ve ön işlemesiz Tokenizasyon süreci

4.3.4. Dördüncü Çalışmada Önerilen Model

Dördüncü çalışmada, sosyal ağlarındaki spam mesajları sorununu çözmek için tekrarlayan sinir ağı (RNN), ALBERT önceden eğitilmiş modeli ile hibritleştirilmiştir. Önerilen model, veri hazırlama ve özellik çıkarma fazı ve sınıflandırma fazı olmak üzere iki alt fazdan oluşmaktadır. Çalışmada önerilen derin sinir ağı modelinde tokenizasiyon yaklaşımı nedeniyle öznitelik çıkarıcı olarak ALBERT modelini kullanılmıştır. ALBERT model, sözcük dağarcığındaki sözcükleri işlemek için veriye dayalı bir belirteç olan Sentencepiece tokenizere dayanmaktadır. Bayt düzeyinde bir yaklaşıma sahip olması ve her baytı ayrı ayrı işleme yeteneği nedeniyle, ALBERT belirteci, emoji gibi karmaşık karakterleri, diğer yaklaşımlar belirteçlerde kullanılan [UNK] veya [OOV] belirteçlerini kullanmak yerine birden çok baytın bir kombinasyonu olarak kodlayabilmektedir. Öte yandan, sınıflandırma aşaması BLSTM tabanlı derin sinir ağ mimarisine dayanmaktadır. BLSTM katman kullanıldığında tek bir model eğitmek yerine iki model eğitilmektedir. İlk model sağlanan girdinin sırasını öğrenir iken ikinci model bu sıranın tersini öğrenmektedir. Başka bir deyişle, bağlam soldan sağa ve sağdan sola iki yönde okunabilir, bu da bir girdi dizisinin her bileşenin hem geçmişten hem de günümüzden bilgi içerdiği anlamına gelmektedir.

4.3.4.1. Önerilen Modeldeki Parametrelerin İnce Ayarlanması

Hiper parametrelerin, derin öğrenme modelin performansı üzerinde doğrudan etkisi vardır. Bu nedenle, üstün performans elde etmek için bu parametreleri optimize etmek önemlidir. Hiper parametreleri optimize etmek için farklı stratejiler vardır.

En yaygın kullanılan stratejiler grid search ve rastgele arama (random search). Bu çalışmada, önerilen modelin hiper parametreleri optimize etmek için rastgele arama algoritması kullanılmıştır. Rastgele arama algoritması kullanılarak çoklu hiper parametre değerleri optimize edilmiş olup en iyi sonuçları veren model seçilmiştir. LSTM katmanlarının sayısı, bırakma varlığı, bırakma değeri, yoğun katman sayısı, yoğun katmanların her birinde kullanılan nöron sayısı, öğrenme hızı, ağırlık başlatıcısı ve aktivasyon fonksiyonlar gibi birçok hyper parametre değeri rastgele algoritması kullanılarak optimize edilmiştir. Rastgele algoritması yürütürken kullanılan hiper parametrelerin her birinin değer aralığı çizelge 4.7'de gösterilmiştir.

Çizelge 4.7 Ayar sırasında denenmiş hiper parametre değerleri

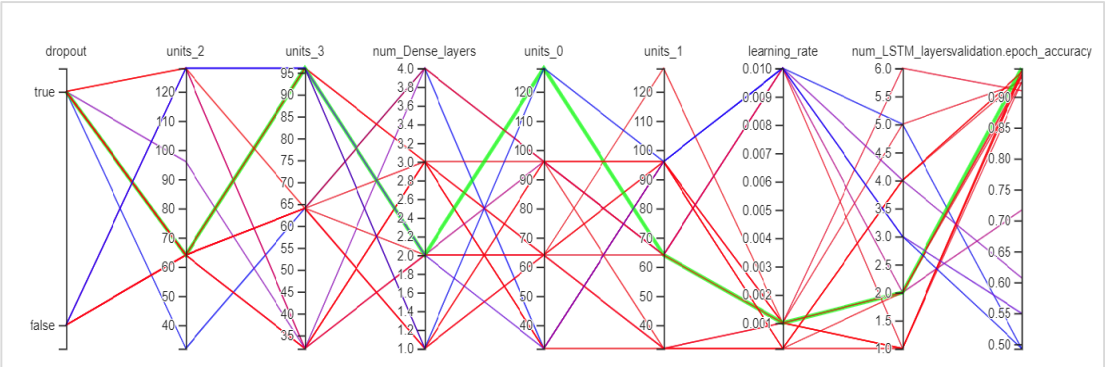
Hiper parametre	**Değer aralığı**
LSTM katmanlarının sayısı	1-6
Bırakma varlığı	[True, False]
Bırakma değeri	[0.1,0.2,0.3,0.4,0.5]
Yoğun katmanların sayısı	1-4
Her Yoğun katmandaki nöron sayısı	[32, 64, 96, 128]
Öğrenme oranı	[0.01, 0.001, 0.0001]
Ağırlık başlatıcı (weight initializer)	['uniform', 'lecun_uniform', 'normal', 'zero', 'glorot_normal', 'glorot_uniform', 'he_normal', 'he_uniform']
Aktivasyon fonksiyonları	[relu, tanh]

Rastgele arama algoritması, her denemede dört yürütme kapsayan 20 deneme yapacak şekilde ayarlanmıştır ve her yürütmede model 5 epoch için eğitilmiştir. Sınıflandırma doğruluğu, yürütme sırasında izlendi ve rastgele algoritmasındaki amaç fonksiyonunu, doğrulama doğruluğu maksimum edecek şekilde tanımlanmıştır. Rastgele aramanın benimsenmesi için kullanılan yapılandırma ayarları çizelge 4.8'de gösterilmektedir.

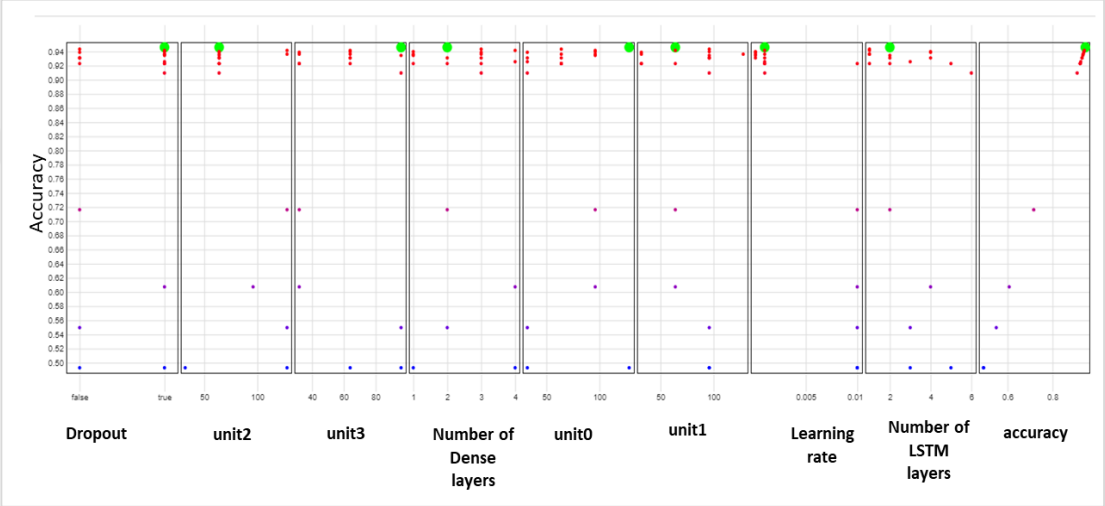
Çizelge 4.8 rastgele aramada kullanılan yapılandırma ayarları

Deneme sayısı	20
Deneme başına yürütme	4
Deneme başına epok	5

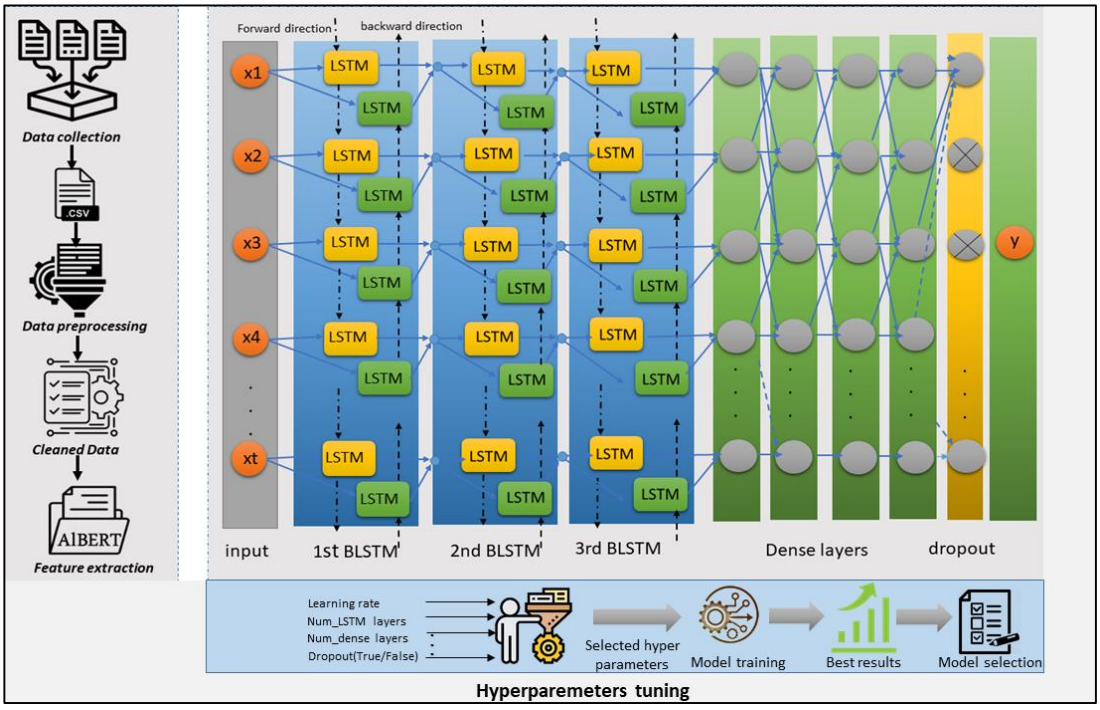
Son olarak, kullanılan üç veri kümesinde en iyi doğrulama doğruluğunu veren en iyi modeli elde edilmiştir. Seçilen model, her biri 256 birimine sahip olan üç BLSTM katmanı, her biri 32 nörona sahip olan dört yoğun katman, bıraka katmanının ve çıkış katmanından oluşmaktadır. Önerilen modelde dropout değeri 0.2, ağırlık başlatıcısı 'he_uniform' olarak seçilmiş ve Relu fonksiyonu yoğun katmanlarda aktivasyon fonksiyonu olarak kullanılmıştır.



Şekil 4.8 Yürütülen rastgele arama hiper parametrelerinin ince ayarı için paralel koordinatlar görünümü



Şekil 4.9 Yürütülen rastgele arama hiper parametrelerinin ince ayarı için dağılım grafiği matris görünümü



Şekil 4.10 dördüncü çalışmada önerilen modeli

Çizelge 4.9 YouTube veri kümesi kullanılarak gerçekleştirilen hiper parametre ince ayarlama sonuçları

Deneme sayısı	bırakma	Birim 2	Birim 1	Birim 3	Dense katman sayısı	Birim 0	Learning rate	Eğtim doğruluk	Doğrulama doğruluk	Eğtim kaypı	Test kaypı
0	true	-	64	-	4	32	0.001	0.92516	0.92553	0.19694	0.19296
1	true	32	96	64	4	32	0.0001	0.50056	0.49291	0.70626	0.70383
2	true	64	96	96	3	32	0.001	0.87653	0.90957	0.34381	0.2838
3	false	64	32	96	3	64	0.01	0.95692	0.94326	0.11603	0.18043
4	true	64	64	32	3	64	0.01	0.92294	0.92287	0.21103	0.19901
5	false	64	64	64	2	64	0.01	0.94559	0.93085	0.15676	0.19206
6	false	64	32	64	3	32	0.001	0.94448	0.93085	0.15121	0.19475
7	false	64	32	32	3	32	0.01	0.94426	0.93883	0.16144	0.17078
8	true	128	64	32	3	96	0.001	0.94848	0.93617	0.14338	0.19642
9	true	64	32	64	1	64	0.001	0.94559	0.93617	0.14904	0.17147
10	true	96	32	32	4	96	0.01	0.61115	0.60727	0.64572	0.64864
11	false	128	96	32	2	96	0.0001	0.69776	0.71631	0.58824	0.58255
12	true	64	64	96	2	128	0.001	0.94581	0.94592	0.15504	0.15537
13	**true**	**32**	**32**	**32**	**4**	**32**	**0.001**	**0.9853**	**0.9521**	**0.0470**	**0.1944**
14	true	64	64	64	1	96	0.001	0.94715	0.93972	0.14341	0.19378
15	true	128	64	64	4	96	0.001	0.95514	0.94149	0.12171	0.1697
16	true	64	96	64	1	64	0.001	0.92449	0.92287	0.26203	0.26102
17	true	64	96	96	1	96	0.001	0.95203	0.9344	0.13678	0.18747
18	false	128	96	96	2	32	0.01	0.55008	0.54965	0.66753	0.65661
19	false	128	64	96	1	128	0.01	0.501	0.49291	0.69329	0.69345

Çizelge 4.10 En iyi modelin yapısı

Parametre	**Değer**
# LSTM katmanlarının sayısı	3
# LSTM katmanlarındaki birim sayısı	256→256→256
# Yoğun katman sayısı	4
# Yoğun katmanlardaki birim sayısı	64→32→32→32
Aktivasyon fonksiyonu	ReLu
Weight initializer	he_uniform
Dropout	True
Dropout değeri	0.2
Çıkış Aktivasyon fonksiyonu	Sigmoid
Öğrenme oranı	0.001
Optimizer	Adam
Kayıp fonksiyonu	Binary crossentropy
Batch size	128
# epoch	10

5. ARAŞTIRMA BULGULARI

5.1. Değerlendirme Metrikleri:

Doğruluk (Accuracy), F1-skoru (F1-score), Duyarlılık (Recall), ve Kesinlik (Precision) gibi metrikleri önerilen yöntemlerin performansı ölçmek için standart değerlendirme metrikler olarak kullanılmıştır.

$$\boldsymbol{Accuracy} = (\boldsymbol{TN} + \boldsymbol{Tp})/(\boldsymbol{Tp} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN}) \quad (5.1)$$

$$\boldsymbol{Kesinlik}\,(\boldsymbol{P}) = \boldsymbol{TP}/(\boldsymbol{TP} + \boldsymbol{FP}) \quad (5.2)$$

$$\boldsymbol{Recall}\,(\boldsymbol{R}) = \boldsymbol{TP}/(\boldsymbol{TP} + \boldsymbol{FN}) \quad (5.3)$$

$$\boldsymbol{F1 - Skor} = (\boldsymbol{2} * \boldsymbol{P} * \boldsymbol{R})/(\boldsymbol{P} + \boldsymbol{R}) \quad (5.4)$$

Ayrıca, MCC metriği kullanılmıştır. MCC, Matthews korelasyon katsayısı anlamına gelmektedir. MCC ikili sınıflandırmanın kalitesini ölçmek için kullanılmaktadır ve -1 ile +1 arasında bir değer döndürmekte. Aşağıdaki formülü kullanılarak doğrudan karışıklık matrisinden hesaplanabilmektedir:

$$\text{MCC} = \frac{\text{TP}*\text{TN} - \text{FP}*\text{FN}}{\sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}} \quad (5.5)$$

- **Karışıklık Matrisi**

Karışıklık matrisi (hata matrisi olarak da bilinmektedir), bir sınıflandırma algoritmasının performansını özetlemek için kullanılan bir tekniktir. Matrisin her satırı gerçek bir sınıftaki örnekleri temsil ederken, her sütun tahmin edilen bir sınıftaki örnekleri temsil eder (veya bunun tersi de geçerlidir).

Çizelge 5.1 Hata (karışıklık) matrisi

		Tahmin	
	Toplam = P + N	Pozitif (PP)	Negatif (PN)
Gerçek	Pozitif (P)	Gerçek pozitif (TP)	Yanlış negatif (FN)
	Negatif (N)	Yanlış pozitif (FP)	Gerçek negatif (TN)

Herhangi bir ikili karışıklık matrisin şablonu, dört tür sonucu yani gerçek pozitif, yanlış negatif, yanlış pozitif ve gerçek negatif olarak tahmin edilen veri örnek sayısı içermektedir. 2×2 karışıklık matrisi aşağıdaki gibi formüle edilebilmektedir:Burada TP gerçek pozitiflerin sayısıdır, TN gerçek negatiflerin sayısıdır, FP yanlış pozitiflerin sayısıdır ve FN yanlış negatiflerin sayısıdır. Başka bir değiş ile tez çalışmasında TP, spam olarak doğru bir şekilde sınıflandırılan tweet sayısı gösteririken TN, spam olmayan olarak doğru bir şekilde sınıflandırılan tweet sayısı göstermektedir.

Birinci ve ikinci çalışmalardaki tüm deneyleri, Intel Xeon E5-2680 8 çekirdek ve 64 GB RAM kullanılarak gerçekleştirilmiştir. Öte yandan, üçüncü ve dördüncü çalışmadaki deneyleri, GPU run time ile Google Colaboratory, Python3 kullanılarak gerçekleştirilmiştir.

5.2. İlk Çalışmada Deneysel Sonuçlar:

İlk çalışmada, önerilen yöntem ile karşılaştırmak amacıyla, metin sınıflandırma problemlerinde kullanılan en yaygın makine öğrenimi sınıflandırıcılarından biri Naive Bayes (NB) sınıflandırıcı ile CountVectorizer ve TF-IDF gibi en popüler BoW tabanlı ağırlıklandırma yöntemlerini denenmiştir.

Ayrıca, word2vec, Glove ve fastText gibi en yaygın Word embedding yöntemleriyle üç farklı derin sinir ağı mimarileri, yani Evrişim sinir ağı (CNN), Uzun kısa süreli bellek sinir ağı (LSTM) ve çift yönlü uzun kısa vadeli sinir ağı (BLSTM) mimarileri uygulanılıp test edilmiştir. Çizelge 5.2, önerilen modelin karışıklık matrisini göstermektedir; Çizelge 5.3, Twitter veri kümesine uygulandığında önerilen yöntemi kullanılarak elde edilen sonuçları göstermektedir. Çizelge 5.4, Şekil 5.1. ve Şekil 5.2, farklı sinir ağ mimarileri ile geleneksel kelime gömme yöntemleri, Twitter ve SMS spam veri seti kullanılarak test edildiğinde, bu yöntemler arasında bir karşılaştırma göstermektedir. Çizelge 5.5 ve Şekil 5.3 önerilen yöntem ile geleneksel ağırlıklandırma dayalı yöntemler arasındaki karşılaştırma sonuçlarını göstermektedir.

Çizelge 5.2 ilk çalışmada karışıklık matrisi

	Tahmin edilen sınıf %	
Gerçek sınıf %	**Spam**	**Ham**
	98.3	1.7
	2.1	**97.9**

Çizelge 5.3 İlk çalışmada Twitter veri setinde önerilen yöntemin test sonuçları

Doğruluk	**Kesinlik**	**Duyarlılık**	**F1-skor**
98.1	98	98.1	98.05

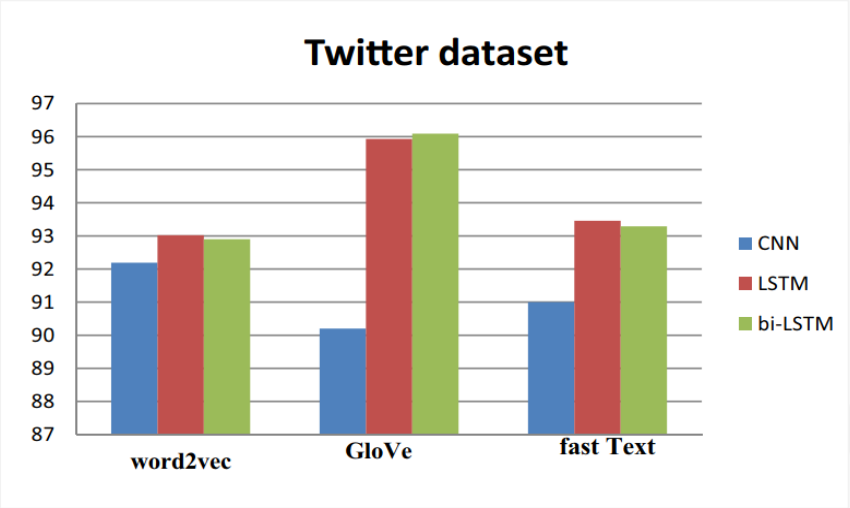
Çizelge 5.4 İlk çalışmada Word embedding tabanlı modellerinin doğruluk sonuçları

Word embedding yöntemi		Word2vec			GloVe			Fasttext		
Sinir ağı mimarisi		CNN	LSTM	BLSTM	CNN	LSTM	BLSTM	CNN	LSTM	BLSTM
Doğruluk (%)	Twitter veri seti	92.19	93.02	92.9	90.2	95.93	96.09	91	93.46	93.29
	SMS spam veri seti	99.16	99.04	98.69	96.77	98.03	97.85	98.81	99.28	99.22

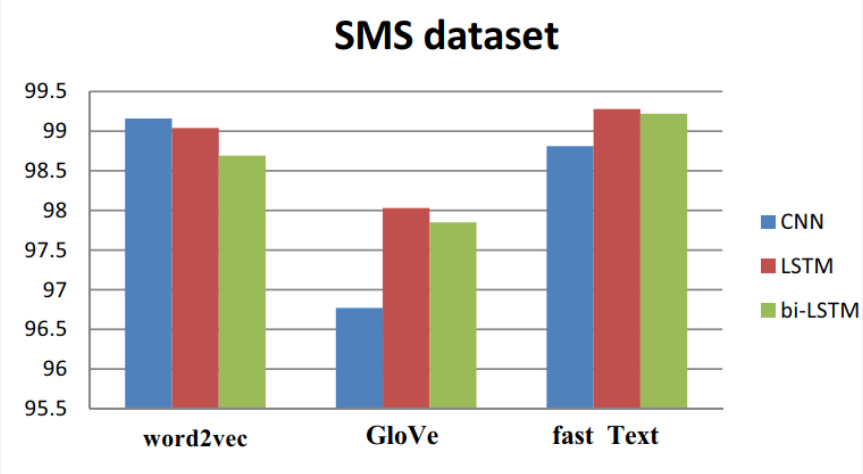
Çizelge 5.5 İlk çalışmada NaiveBayes sınıflandırıcı ile geleneksel ağırlıklandırma melezleştiren yöntemlerinin performans değerlendirmesi

Veri seti	Metot	Doğruluk	Precision	Recall	F1-score
Twitter Veri seti	Count vectors	0.927	0.95	0.92	0.94
	kelime seviyesi TF-IDF	0.866	0.92	0.84	0.88
	N-gram seviyesi TF-IDF	0.818	0.80	0.92	0.86
	Karakter seviyesi TF-IDF	0.811	0.92	0.74	0.82

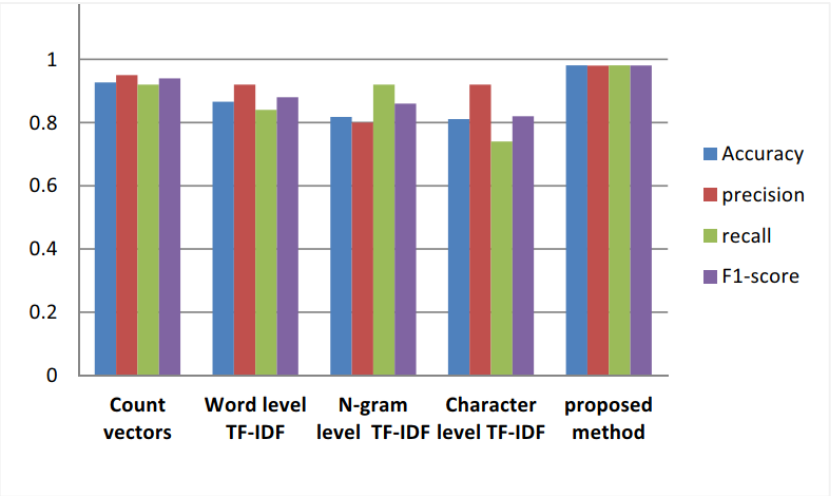
Veri seti	Metot	Doğruluk	Precision	Recall	F1-score
SMS spam Veri seti	Count vectors	0.974	0.87	0.93	0.90
	kelime seviyesi TF-IDF	0.980	1.00	0.87	0.93
	N-gram level TF-IDF	0.976	1.00	0.81	0.89
	Karakter seviyesi TF-IDF	0.980	0.98	0.85	0.91



Şekil 5.1 Twitter veri setinde farklı sinir ağı mimarileri ile Word embedding yöntemlerinin doğruluk karşılaştırması



Şekil 5.2 SMS veri setinde farklı sinir ağı mimarileri ile Word embedding yöntemlerinin doğruluk karşılaştırması



Şekil 5.3 İlk çalışmada önerilen yaklaşımın Twitter veri setinde diğer geleneksel ağırlıklandırma yöntemleriyle performans karşılaştırması

5.3. İkinci Çalışmada Deneysel Sonuçlar

İkinci çalışmada, önerilen kelime temsil yöntemini, doğrusal regresyon ve Naive Bayes makine öğrenme sınıflandırıcıları ile melekleştirilen olan Count Vectorizer, kelime seviyesinde TF-IDF, N-gram seviyesinde TF-IDF ve karakter seviyesi TF-IDF gibi bazı geleneksel özellik ağırlıklandırma yöntemleriyle karşılaştırılmıştır.

Çizelge 5.6, Çizelge 5.7, Çizelge 5.8, Şekil 5.4, Şekil 5.5 ve Şekil 5.6, bu yöntemler, Twitter, YouTube ve SMS veri kümelerine uygulandığında elde edilen deneysel sonuçları sırasıyla göstermektedir. Öte yandan, modelimizi Glove ve fastText gibi geleneksel Word embedding yöntemleri kullanan benzer modellerle karşılaştırılmıştır. Çizelge 5.9 ve Çizelge 5.10, GloVe ve FastText modellerinde kullanılan parametreleri gösterirken, Çizelge 5.11 ve Şekil 5.7, Twitter, YouTube ve SMS veri kümelerine önceki modeller uygulandığında elde edilen deneysel sonuçları göstermektedir. Son olarak Çizelge 5.12 bu çalışmada önerilen yöntemin test sonuçları göstermektedir.

Çizelge 5.6 Twitter veri setine geleneksel ağırlıklandırma yöntemleri uygulandığında elde edilen deneysel sonuçlar

Metot	Sınıflandırıcı	Doğruluk
Count vectors	Naïve Bayes	92.5
Kelime seviyesi TF-IDF	Naïve Bayes	86.75
N-gram seviyesi TF-IDF	Naïve Bayes	82.5
karakter seviyesi TF-IDF	Naïve Bayes	81.8

Metot	Sınıflandırıcı	Doğruluk
Count vectors	Linear regresyon	97.2
Kelime seviyesi TF-IDF	Linear regresyon	93.3
N-gram seviyesi TF-IDF	Linear regresyon	83.5
karakter seviyesi TF-IDF	Linear regresyon	91.2

Çizelge 5.7 Geleneksel ağırlıklandırma yöntemleri, YouTube veri setine uygulandığında elde edilen deneysel sonuçlar

Metot	Sınıflandırıcı	Doğruluk
Count vectors	Naïve Bayes	90.4
Kelime seviyesi TF-IDF	Naïve Bayes	89.6
N-gram seviyesi TF-IDF	Naïve Bayes	87.4
Karakter seviyesi TF-IDF	Naïve Bayes	91.1
Count vectors	Linear regression	91.5
Kelime seviyesi TF-IDF	Linea rregression	90
N-gram seviyesi TF-IDF	Linear regression	85.1
Karakter seviyesi TF-IDF	Linear regression	92.3

Çizelge 5.8 Geleneksel ağırlıklandırma yöntemleri, SMS veri setine uygulandığında elde edilen deneysel sonuçlar

Metot	Sınıflandırıcı	Doğruluk
Count vectors	Naïve Bayes	97.4
Kelime seviyesi TF-IDF	Naïve Bayes	96.84
N-gram seviyesi TF-IDF	Naïve Bayes	96.63
Karakter seviyesi TF-IDF	Naïve Bayes	98.13
Count vectors	Linear regresyon	98.1
Kelime seviyesi TF-IDF	Linear regresyon	97.2
N-gram seviyesi TF-IDF	Linear regresyon	96.13
Karakter seviyesi TF-IDF	Linear regresyon	98.14

Çizelge 5.9 GloVe modelinde kullanılan parametreler

Model parametresi	Değer
Optimize Edici	Adam
Kayıp fonksiyonu	Binary cross entropy
Parti boyutu	32
Epoch	30
Optimizasyon metriği	Doğruluk
LSTM birimi sayısı	100
Dropout	0.25

Çizelge 5.10 Fasttext modelinde kullanılan parametreler

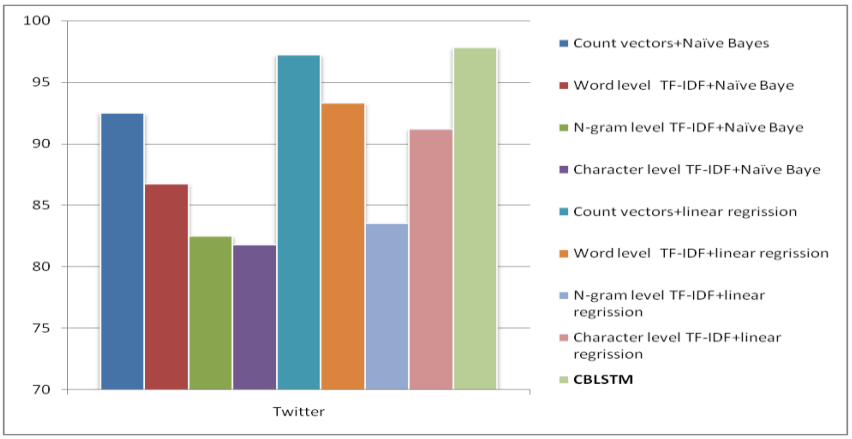
Model parametresi	**Value**
Optimize Edici	Adam
Kayıp fonksiyonu	Binary cross entropy
Batch boyutu	64
Epoch	30
Optimizasyon metriği	Doğruluk
LSTM birimi sayısı	300
Dropout	0.25

Çizelge 5.11 Önerilen model ile geleneksel Word embedding tabanlı modellerinin performans karşılaştırması

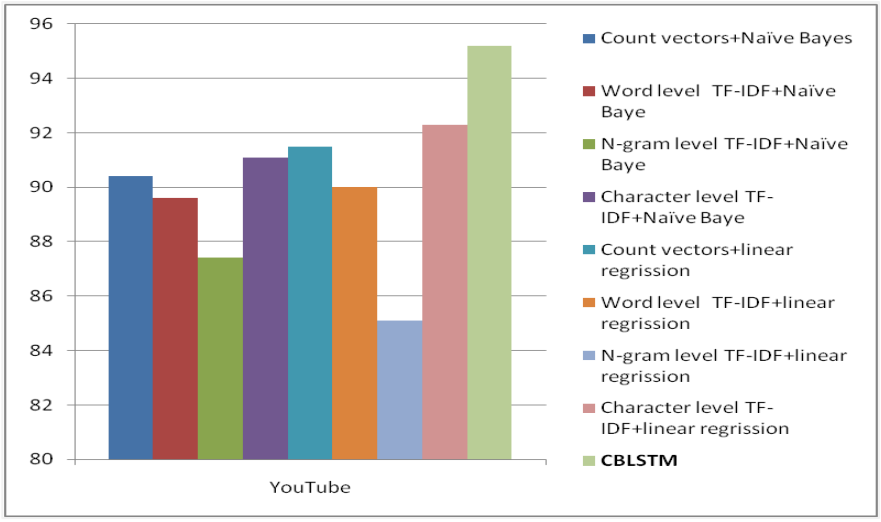
Model	**Veri seti**	**Doğruluk**
Glove	Twitter	95.01
FastText	Twitter	92.82
Glove	YouTube	92.8
FastText	YouTube	92.7
Glove	SMS	98.28
FastText	SMS	98.81

Çizelge 5.12 İkinci çalışmada önerilen yöntemin test sonuçları

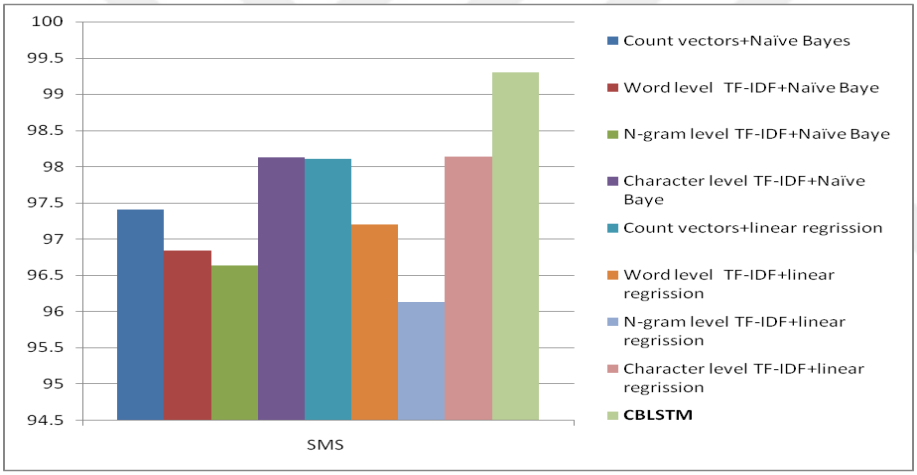
Veri seti	**Doğruluk**
Twitter	97.8
YouTube	95.21
SMS	99.30



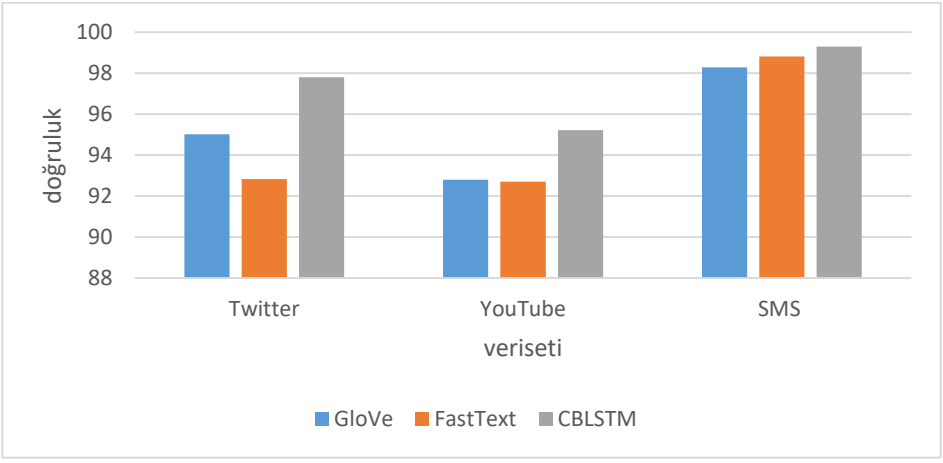
Şekil 5.4 Twitter veri seti kullanıldığında geleneksel ağırlıklandırma yöntemleriyle performans karşılaştırması



Şekil 5.5 YouTube veri seti kullanıldığında geleneksel ağırlıklandırma yöntemleriyle performans karşılaştırması



Şekil 5.6 SMS veri seti kullanıldığında geleneksel ağırlıklandırma yöntemleriyle performans karşılaştırması



Şekil 5.7 Geleneksel kelime yerleştirme modelleriyle performans karşılaştırması

5.4.Üçüncü Çalışmada Deneysel Sonuçlar

Üçüncü çalışmada, en popüler transformer tabanlı yöntemleri ve bunların sosyal ağlarda spam tespit alanındaki uygulamaların karşılaştırmalı bir çalışması yapılmıştır. Ardından sosyal ağlarda spam tespit etmek için derin yığınlı çift yönlü bir sinir ağ mimarisi önerilmiştir. Bundan sonra, daha fazla bağlam sağlamak için yığılmış çift yönlü uzun kısa vadeli sinir ağ mimarisi ile RoBERTa önceden eğitilmiş modeli melezleştirilmiştir. Başka bir değişiyle, tweet'lerin bağlamsal kelime temsiline dayanarak Twitter'daki spam tweet'leri tespit etmek için RoBERTa dayalı yeni bir Hibrit derin sinir ağ modeli önerilmiştir.

Karşılaştırma amacıyla BERT, DistilBERT, ALBERT, ELECTRA, RoBERTa ve DistilRoBERTa gibi Transformer tabanlı farklı modeller de test edilmiştir.

Genel olarak Transformer tabanlı modelleri iki yaklaşım kullanılarak benimsenebilmektedir. Transformer tabanlı modelleri kullanmanın birinci yaklaşım, ince ayar (fine-tuning) yaklaşımı olarak adlandırılmaktadır. Bu yaklaşımda, modele köklü değişiklikler yapmadan göreve özel veriler üzerinde önceden eğitilmiş modelde ince ayar yapılmaktadır. Transformer tabanlı modelleri kullanmanın ikinci yaklaşım, özellik tabanlı yaklaşım olarak bilinmektedir. Bu yaklaşımda model, başka bir görevde kullanılmak üzere eğitim öncesi yerleştirmenin (embedding) çıkarılmasına izin vermektedir.

Bu çalışmada her iki yaklaşımı da kullanılmıştır. İlk olarak, önceden eğitilmiş modeller doğrudan sınıflandırıcı olarak kullanılmıştır. Daha sonra önceden eğitilmiş modeller, önerilen sinir ağı mimarisi içindeki kelime temsillerini çıkarmak için kullanılmıştır.

Önerilen model değerlendirmek için doğruluk ve MCC olmak üzere iki standart metrik kullanılmıştır.

Deneyler iki aşamada gerçekleştirilmiştir. Şekil 5.8 gösterdiği gibi bu çalışmada yürütülen deneyleri iki aşma olarak gerçekleştirilmiştir. İlk deneysel aşamasında, BERT, AlBERT, RoBERTa, DistilBERT, DistilRoBETa ve ELECTRA gibi en popüler transformer tabanlı modelleri, ince ayar yaklaşımı kullanılarak benimsenmiştir. Bahse geçen transformer tabanlı modelleri, Twitter veri kümesindeki

spam mesajları sınıflandırmak için kullanılmıştır. Daha sonra, kullanılan modellerin performansını karşılaştırmak için MCC gibi farklı değerlendirme metrikleri kullanılmıştır. İkinci aşamada, birinci aşmada en iyi sonuç veren modelleri, özellik çıkartma tabanlı yaklaşımı kullanılarak benimsenmiştir. Başka bir değişiyle, bu modeller, kullanılan veri setindeki metinden bağlamsal temsili çıkarmak için özellik çıkarıcı olarak kullanılmıştır. Daha sonra, çıkartılan özellikleri, tekrarlayan sinir ağı tabanlı mimariye beslemiştir. Önerilen modeller arasında en yüksek performansı bulmak için farklı değerlendirme metrikleri kullanılmıştır. Son olarak, önerilen modelin bazı parametreleri en yüksek performansa ulaşmak için bir analiz çalışması gerçekleştirilerek ayarlanmıştır. Çizelge 5.13 ve Şekil 5.9, ilk aşamada tweet'leri sınıflandırmak için kullanıldığında modellerin performans karşılaştırmalarını göstermektedir. Öte yandan, Çizelge 5.14 ve Şekil 5.10, ikinci aşamada önerilen yapay sinir ağ modeli ile birlikte kullanıldığında modellerin performans karşılaştırmalarını göstermektedir. Ayrıca, Çizelge 5.15 ve Şekil 5.11, farklı dropout değerleri kullanıldığında elde edilen doğruluk sonuçları göstermektedir.

Önerilen model kullanılarak elde edilen sonuçları, geleneksel Word embedding yöntemleri metin temsili olarak kullanılan aynı modeli ile karşılaştırılmıştır. Bu amaçla, word2vec, Global vector (GloVE) ve fastText gibi geleneksel Word embedding yöntemleri kullanılarak metin temsili çıkartmış olup önerilen modele beslenmiştir.

Çizelge 5.13 Modeler arasındaki performans karşılaştırması

Model	Doğruluk	MCC	Eğitim Kaybı	Değerlendirme Kaybı
BERT (bert-base-cased)	0.9901	0.979	0.00103	0.0442
DistilBERT(distilbert-base-cased)	0.9851	0.969	0.00214	0.0492
Roberta (roberta-based)	0.9913	0.9823	0.00017	0.0371
DistilRoberta (distilroberta-base)	0.9845	0.9683	0.02316	0.0559
Electra (electra-base)	0.9852	0.970	0.00083	0.0591

Çizelge 5.16, önerilen model kullanılarak elde edilen sonuçları, RNN-fastText, RNN-word2vec ve RNN-GloVE modelleri kullanılarak elde edilen sonuçları ile karşılaştırılmıştır. Burada, RNN-fastText modelinde, önerilen modelde kullanılan

RoBERTa metin temsil yöntemi yerinde fastText metin temsil yöntemi kullanılmıştır. RNN-word2vec de, önerilen modelde kullanılan RoBERTa metin temsil yöntemi yerinde word2vec metin temsil yöntemi kullanılmıştır. RNN-GloVE modelinde ise, önerilen modelde kullanılan RoBERTa metin temsil yöntemi yerinde GloVE metin temsil yöntemi kullanılmıştır.

Çizelge 5.14 Önerilen sinir ağı mimarisi ile kullanılan modellerin performans karşılaştırması

Model	Doğruluk	Kayıp	Değerlendirme Kaybı	Değerlendirme Doğruluk
BLSTM-BERT	0.9947	0.0142	0.0142	0.9773
BLSTM-Roberta	0.9946	0.0155	0.0793	0.9824
BLSTM-DistilRoBERTa	0.9916	0.0233	0.0665	0.9818
BLSTM-ALBERT	0.9866	0.0352	0.0753	0.9755

5.4.1. Gerçekleştirilen Analiz Çalışması

En iyi performansa ulaşmak için bir analiz çalışması ve parametre ayarlaması yapılmıştır. Önerilen model, Twitter veri seti, YouTube yorumları veri seti ve SMS spam veri seti olmak üzere üç kıyaslama veri seti üzerinde test edilmiştir. Önerilen modelin etkinliğini ve verimliliğini, hiper parametreleri değiştirerek ve üç veri kümesi üzerinde değerlendirilerek test edilmiştir. Çizelge 5.17, farklı dropout değerlerine sahip kullanılan temel modelin (iki yığınlanmış BLSTM katmanı) test sırasında elde edilen doğruluk sonuçlarını sunmaktadır. Öte yandan, Çizelge 4.18, farklı BLSTM katmanları kullanılarak değerlendirme yaparken doğruluk sonuçları göstermektedir. Çizelgelardaki sonuçların, modelin birden çok kez eğitilmesi ve değerlendirilmesiyle elde edilen ortalama sonuçları temsil ettiğini belirtmekte fayda vardır.

5.4.2. Diğer Yöntemlerle Karşılaştırması

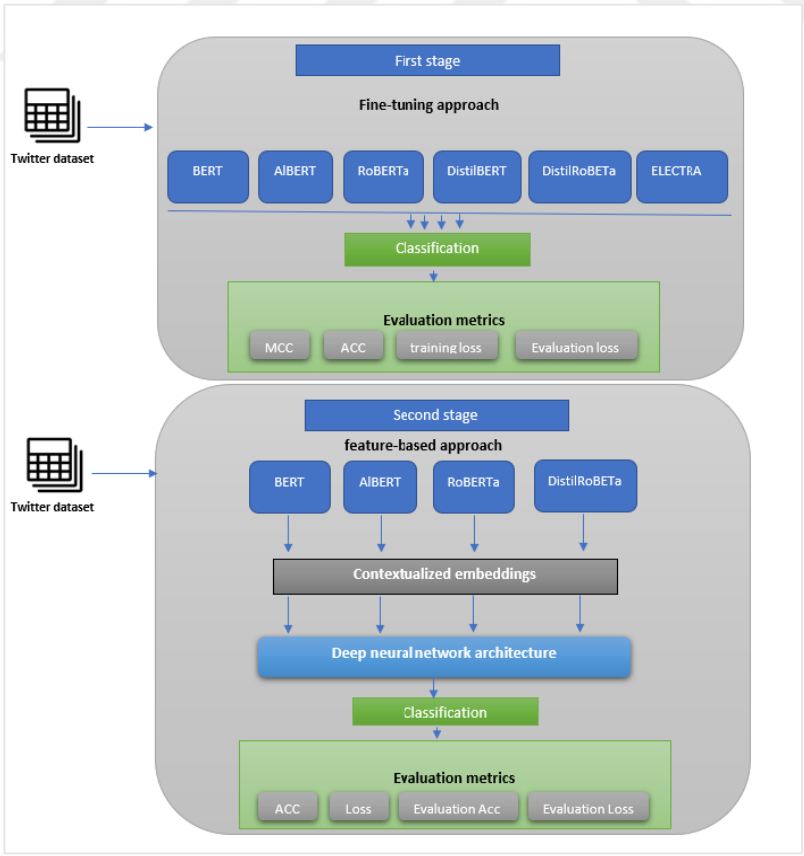
Seçilen kelime temsil yönteminin önemini kanıtlamak için farklı geleneksel word embedding tabanlı modeller test edilmiştir. Hemen hemen aynı parametrelere sahip bir model, word2vec, Global vector (GloVE) ve fastText gibi farklı kelime temsil yöntemleri ile melezleştirilmiştir. Çizelge 5.19,elde edilen sonuçları göstermektedir.

Çizelge 5.15 Dropout'un farklı değerleri ile elde edilen doğruluk

Dropout	Doğruluk	Doğrulama Doğruluk
0.2	0.9959	0.9768
0.25	0.995	0.9824
0.3	0.9937	0.9815
0.4	0.9927	0.9793
0.5	0.9876	0.98
0.6	0.9886	0.9782
0.7	0.9952	0.9768

Çizelge 5.16 Geleneksel Word embedding tabanlı yöntemleri ile üçüncü çalışmada önerilen yöntem arasındaki doğruluk karşılaştırması

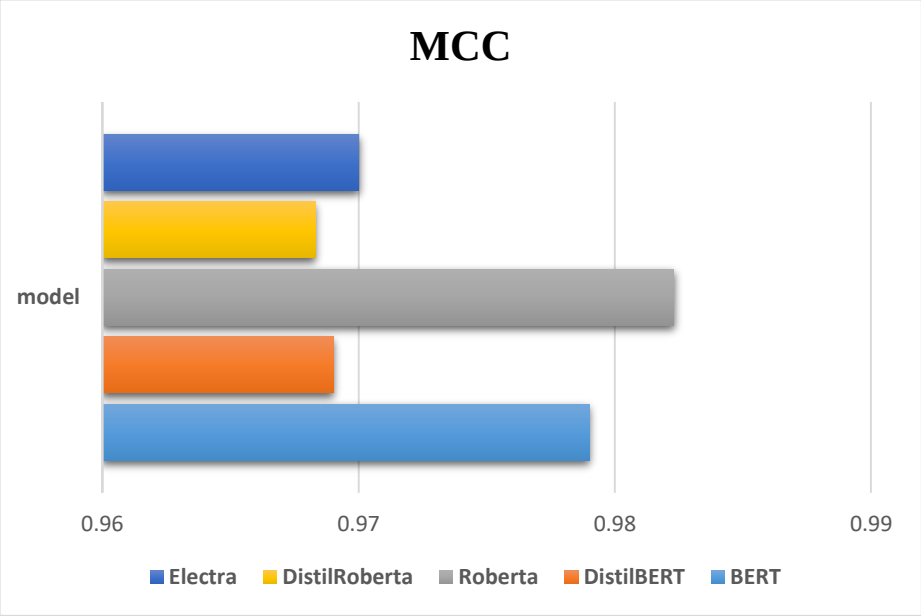
Metot	Doğruluk
RNN-fastText	0.912
RNN-word2vec	0.859
RNN-GloVE	0.945
Önerilen yöntem	0.9824



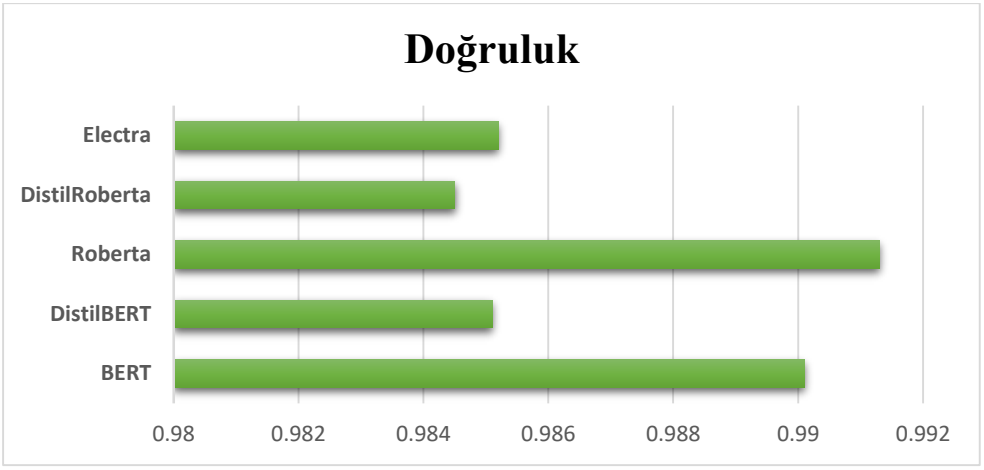
Şekil 5.8 Üçüncü çalışmada Yapılan deneylerin şematik diyagramı

Çizelge 5.17 Farklı dropout değerlerine sahip modelin doğruluk sonuçları

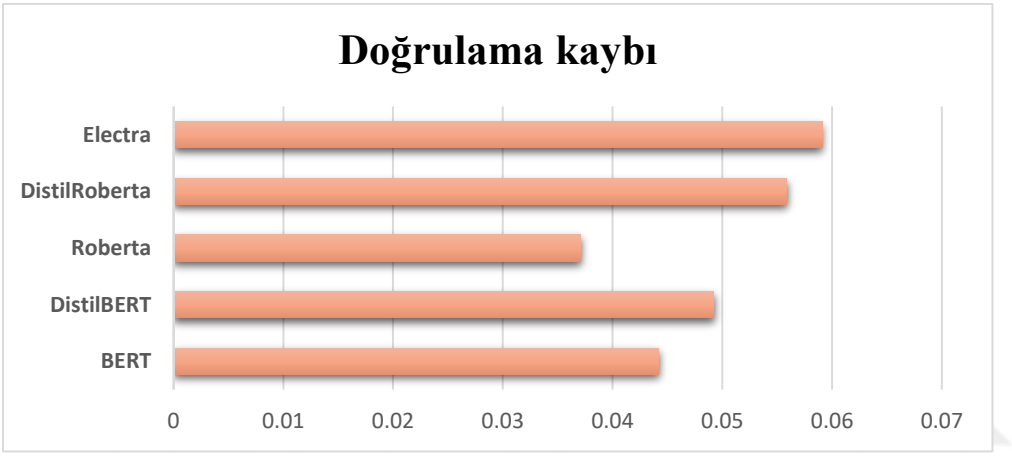
Dropout Değeri	**Doğruluk %**		
	Twitter veri kümesi	YouTube veri kümesi	SMS spam veri kümesi
0.2	97.68	92.53	99.28
0.25	98.15	94.41	99.74
0.3	98.12	93.1	99.19
0.4	97.93	93.74	99.01
0.5	97.23	93.48	99.28
0.6	97.82	92.81	98.83
0.7	97.96	92.61	99.2



5.9.a



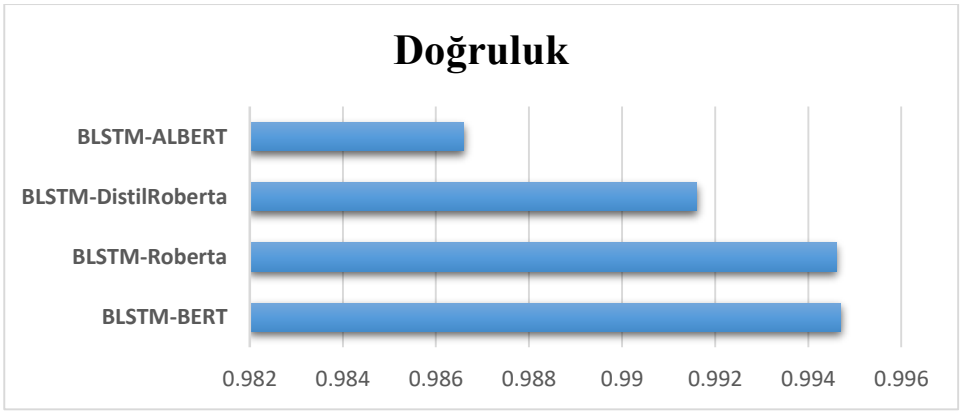
5.9.b



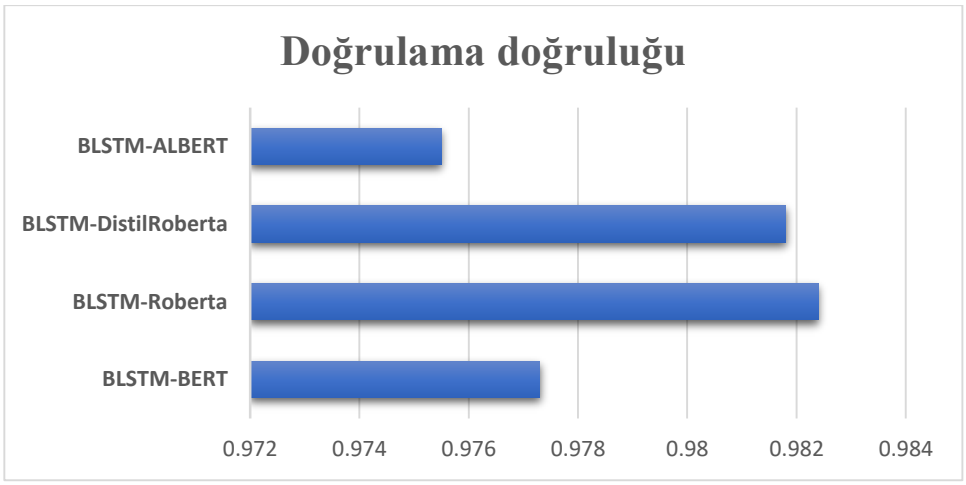
5.9.c

Şekil 5.9 Modeller arasındaki performans karşılaştırması;

5. 9.a MCC açısından,5. 9.b doğruluk açısından, 5. 9.c doğrulama kaybı açısından

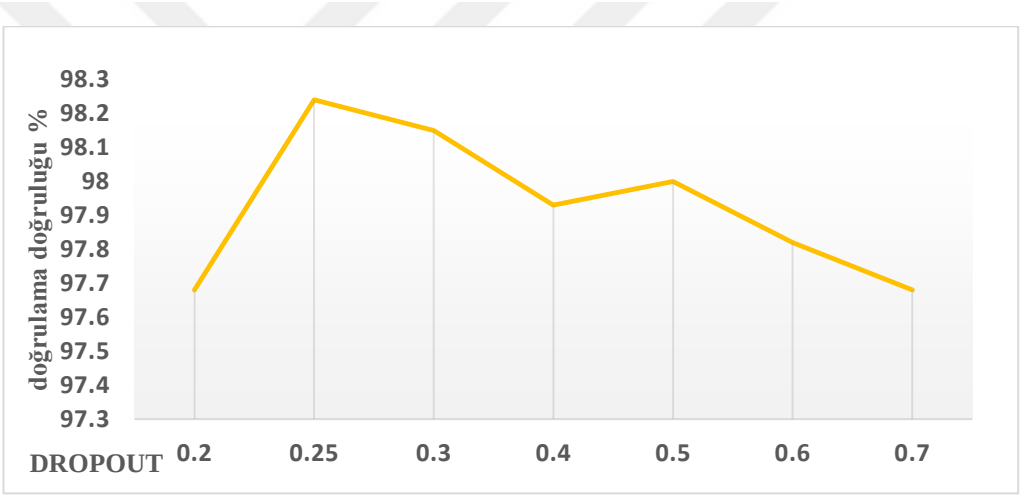


5.10.a

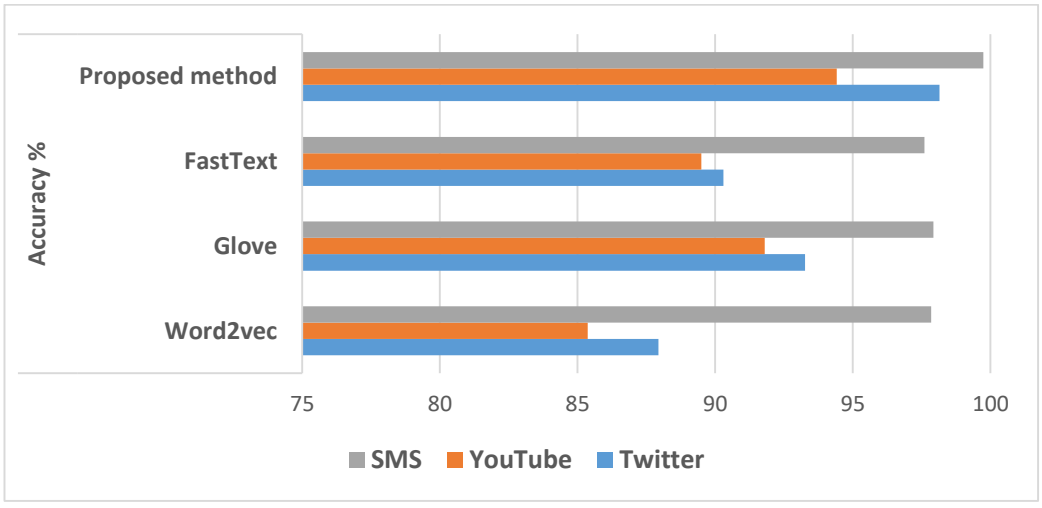


5.10.b

Şekil 5.10 BLSTM tabanlı modellerin performans karşılaştırması; 5.10. a doğruluk açısından,5.10. b doğrulama doğruluğu açısından



Şekil 5.11 Farklı dropout değeri açıdan model değerlendirilmesi



Şekil 5.12 Geleneksel kelime yerleştirme tabanlı yöntemlerle performans karşılaştırması

Çizelge 5.18 Farklı sayıda BLSTM katmanıyla doğruluk sonuçları

BLSTM sayısı	**Doğruluk %**		
	Twitter veri seti	YouTube veri seti	SMS spam veri seti
2	98.15	94.41	99.74
3	97.32	92.15	99.01
4	97.86	93.35	99.28
5	97.88	92.96	99.19
6	97.79	92.55	99.1
7	97.83	92.56	98.65
8	97.75	92.29	98.47

Çizelge 5.19 Üçüncü çalışmada önerilen yöntem diğer Word embedding tabanlı yöntemler ile sonuçlar karşılaştırması

Veri seti	**Doğruluk %**			
	Word2vec	**Glove**	**FastText**	**Önerilen yöntem**
Twitter	87.94	93.26	90.3	**98.15**
YouTube	85.37	91.8	89.5	**94.41**
SMS	97.84	97.92	97.6	**99.74**

5.5. Dördüncü Çalışmada Deneysel Sonuçları

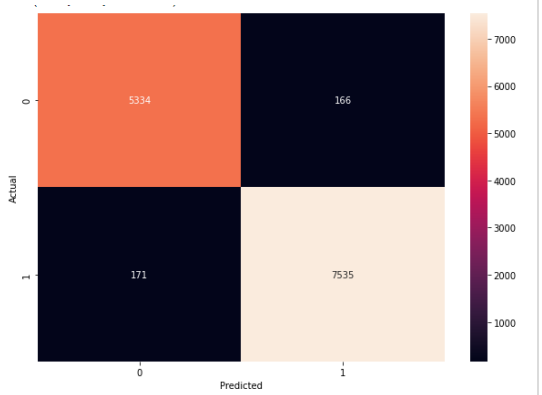
Dördüncü çalışmada, daha fazla bağlam sağlamak için ALBERT olarak bilinen modeli, metin bağlamsal kelime temsili yöntem olarak kullanılmıştır. Başka bir değiş ile yığılmış çift yönlü uzun kısa vadeli sinir ağ mimarisi ile ALBERT modeli melezleştiren yeni bir model önerilirmiştir. Ayrıca, en iyi performansa ulaşmak için bir analiz çalışması ve hiper parametre ince ayarlaması yapılmıştır ve modeli daha önce bahsedilen üç kıyaslama veri seti kullanılarak test edilmiştir.

Üç veri kümesi için en iyi sonuçları verebilecek bir derin öğrenme modeli oluşturmak için kullanılabilecek en uygun hiper parametre değerleri seçmek için rastgele arama algoritması benimsenmiştir. Rastgele aram algoritması sonucunda, çizelge 4.10 'de yer alan yapıya sahip bir model elde edilmiştir. Daha sonra, önerilen model 10 epoch olarak eğitildiğinde çizelge 5.20 'de yer alan en iyi sonuçlara ulaşılmıştır. Şekil 4.8, yürütülen rastgele arama hiperparametreleri ince ayar işlemi için paralel koordinatlar görünümünü göstermektedir. Yatay eksenler, ayarlanmış hiperparametreleri temsil ederken dikey eksenler, her hiperparametre için seçilen değeri temsil eder. Ayrıca, Şekil 4.9, yürütülen rastgele arama hiperparametreleri ince ayar işlemi için Scatter koordinatlar görünümünü göstermektedir. Şekildeki her noktanın bir denemeyi göstermektedir. Son olarak, Şekil 5.13, önerilen modelin kullanılan veri setleri üzerinde uygulanmasıyla elde edilen sınıflandırma raporlarını ve karışıklık matrislerini göstermektedir.

Çizelge 5.20 Dördüncü çalışmada elde edilen sonuçları

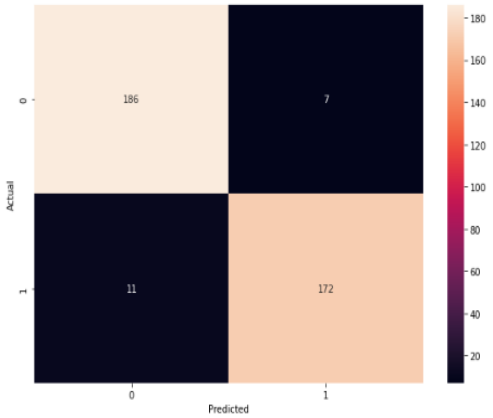
Veri kümesi	Doğruluk	Kesinlik	Duyarlılık	F1-skor
Twitter	0.982	0.98	0.98	0.98
YouTube	0.9548	0.96	0.94	0.95
SMS	0.9912	0.98	0.97	0.97

	precision	recall	f1-score	support
0	0.97	0.97	0.97	5500
1	0.98	0.98	0.98	7706
accuracy			0.97	13206
macro avg	0.97	0.97	0.97	13206
weighted avg	0.97	0.97	0.97	13206



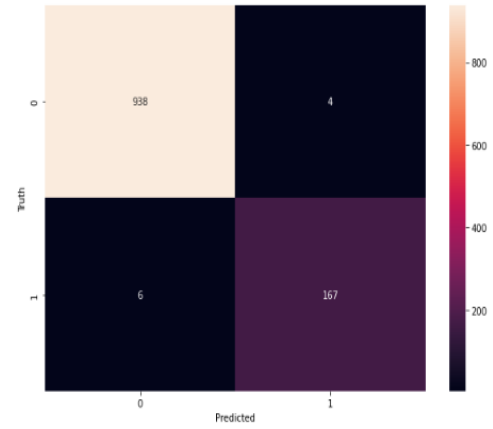
Twitter veri kümesi

	precision	recall	f1-score	support
0	0.94	0.96	0.95	193
1	0.96	0.94	0.95	183
accuracy			0.95	376
macro avg	0.95	0.95	0.95	376
weighted avg	0.95	0.95	0.95	376



YouTube veri kümesi

	precision	recall	f1-score	support
0	0.99	1.00	0.99	942
1	0.98	0.97	0.97	173
accuracy			0.99	1115
macro avg	0.99	0.98	0.98	1115
weighted avg	0.99	0.99	0.99	1115



SMS veri kümesi

Şekil 5.10 Önerilen model kullanılarak elde edilen sınıflandırma raporları ve karışıklık matrisleri

6. SONUÇ VE TARTIŞMA

Bu bölümde, önerilen dört çalışmadan elde edilen sonuçları tartışılacaktır.
Hatırlamak gerekirse, ilk çalışmada Twitter veri kümesindeki metni temsil etmek için BERT kodlayıcıyı kullanmayı önerilmiştir. Ardından önerilen model performansı hem geleneksel ağırlıklandırma yöntemleriyle hem de geleneksel Word embedding yöntemleriyle karşılaştırılmıştır.

Twitter veri setinde spam tespiti için geleneksel word embedding yöntemleri arasındaki karşılaştırmaları sunan Şekil 5.4'de görülebileceği gibi, GloVe temsiline dayalı model diğer modellere kıyasla en iyi sonuçları vermiştir. Yapay sinir ağı tabanlı yapısına gelince, LSTM tabanlı yapının tüm modeller arasında en iyi yapı olduğunu görülmektedir.

SMS veri kümesi kullanılarak test edildiğinde ise, Şekil 5.2 gösterdiği gibi, FastText gömmeye dayalı LSTM yapısı, değir yapılara kıyasla en yüksek sınıflandırma doğruluğu elde etmiştir.

Çizelge 5.5'te görüldüğü gibi geleneksel ağırlıklandırma yöntemler arasında, Count Vectorizer tabanlı NaiveBayes (NB) sınıflandırıcısı performansı Twitter veri seti üzerinde en iyi performans gösterirken, kelime düzeyinde TF-IDF tabanlı NaiveBayes (NB) SMS veri seti üzerinde en iyi sonuçları elde etmiştir. Öte yandan, Çizelge 5.3'de gösterildiği gibi önerilen yöntem, %98.1 sınıflandırma doğruluğu ve %98.05 F1-Score'una ulaşan olağanüstü bir performansla önceki tüm yöntemleri geride bırakmıştır.

İkinci çalışmada ise, çift yönlü uzun kısa vadeli sinir ağı tabanlı bir modelde metin temsil etmek için derin bağlama dayalı bir kelime temsili kullanan ve CBLSTM olarak adlandırılan bir model önerilmiştir. Deneysel sonuçlar, geleneksel özellik ağırlıklandırma yöntemleri ile GloVe ve FastText gibi bazı geleneksel Word embedding yöntemleri karşılaştırılmıştır.
Önerilen modeli oluşturmak için katman sayısı, aktivasyon ve kayıp fonksiyonları ve uygun bırakma değerleri açısından en uygun seçimi sağlamak amacıyla Çizelge 4.4'de gösterildiği gibi bir analiz çalışması gerçekleştirilmiştir. Sonuçları analiz ederek, aşağıdaki gözlemleri sonuçlandırabiliriz:

• İlk olarak, BLSTM tabanlı mimarisinin, BLSTM katman yerine CNN katmanına sahip olan benzer bir mimariye göre daha iyi sonuç verdiğini gözlemlenmiştir.
• Önerilen model, Çizelge 4.4'ün altıncı satırındaki önerilen mimari ve bırakma değeri 0.2'ye eşit olarak ayarlandığında en iyi sonuçları elde etmiştir.
İkinci çalışmada önerilen model, Çizelge 5.12'de gösterildiği gibi Twitter, YouTube ve SMS veri seti ile test edildiğinde üstün başarılı sonuçları elde etmiştir.
Word embedding tabanlı modellerle elde edilen sonuçları (Çizelge 5.11) geleneksel ağırlıklandırma yöntemleriyle elde edilen sonuçlarla (Çizelge 5.6, Çizelge 5.7 ve Çizelge 5.8) karşılaştırılmıştır. Word embedding yöntemlerin, üç veri kümesinde geleneksel ağırlıklandırma yöntemlerinden daha iyi performans gösterdiğini gözlemlenmiştir. Öte yandan, CBLSTM modelimizi kullanılarak elde edilen sonuçların (Şekil 4.4, Şekil 4.5 ve Şekil 4.6 gösterdiği gibi), geleneksel kelime yerleştirme tabanlı modellerin sonuçlarına göre daha iyi olduğunu görülmüştür. Örneğin, GloVe modeliyle karşılaştırıldığında, GloVe tabanlı modeli kullanıldığında sınıflandırma doğruluğu %95.1'e ulaştığı iken CBLSTM kullanıldığında %97.8'e yükselmiştir (yani Performans %2.7 artmıştır). Aynı şekilde YouTube veri seti kullanıldığında, sınıflandırma doğruluğu %92,8'den %95,21'e yükselmiştir. SMS veri setine gelince, sınıflandırma doğruluğu %98.28'den %99.30'a kadar yükselmiştir. Benzer şekilde önerilen model sonuçları, fastText tabanlı model sonuçlarıyla karşılaştırıldığında, sınıflandırma doğruluğun, Twitter veri setinde %4.98, YouTube veri setinde %2.51 ve SMS spam veri setinde %0.94 bir artışı uğradığını gözlemlenmiştir.

Önceki sonuçlardan, özellikle sosyal ağlarda olduğu gibi kısa metinleri işlerken, sınıflandırma sürecinin performansının iyileştirmesinde kelime temsilinin ne kadar önemli olduğunu da anlaşılabilmektedir. Derin bağlama dayalı kelime (deep contextualized Word representation) temsiline dayalı mimarisinin, diğer metin temsil yöntemlerine dayalı mimarilerinden daha iyi performans verebileceğini sonuca varılmıştır.

Bu çalışmadan çıkarabileceğimiz en önemli sonuçlar şunlardır:
1) Sınıflandırma performansını artırmada önemli bir etkiye sahip olduğu için, sosyal ağlardaki verileri ele alırken ve analiz ederken metinsel verileri bağlamsal bir biçimde temsil etmenin büyük önem taşımaktadır.

2) Sosyal ağların kısa metin sınıflandırma problemi çözmek için karakter tabanlı bir temsilin ne kadar önemli olduğunu görülmüştür. Bunun sebebi de, ağ eğitilirken görülmeyen tokenler için sağlam temsiller oluşturmak üzere morfolojik sözcükleri kullanmasına izin vererek bilinen OOV (out of vocabulary) probleminden kaçınmaktadır.

3) Çift yönlü tekrarlayan sinir ağlarına dayalı mimarinin kısa metin sınıflandırma problemi çözmek için ne kadar etkili olduğunu bariz bir şekilde de anlaşılmaktadır.

Üçüncü çalışmada, transformer tabanlı modelleri, derin öğrenme mimarisi ile melezleştirilen yeni yaklaşım önerilmiştir. Yığılmış BLSTM sinir ağının performansını iyileştirmek için Roberta önceden eğitilmiş modeli kullanılarak Tweet metninin bağlamsal kelime temsili çıkarılmıştır. Ayrıca önerilen model sonuçları karşılaştırmak amacıyla BERT, Roberta, DistilBERT, DistilRoBERTa, Electra ve AlBERT gibi en modern transformer tabanlı yöntemleri de test edilmiştir.

Ayrıca, Twitter, YouTube ve SMS veri kümeleri olmak üzere üç kıyaslama veri kümesinde en iyi sonuçları elde etmek için bir analiz çalışması yapılmıştır.

Üçüncü çalışmadaki sonuçları analiz ederek, aşağıdaki gözlemleri sonuçlandırılabilmiştir:

İlk olarak, kullanılan Transformer tabanlı modelleri, MCC, sınıflandırma doğruluğu, eğitim kaybı ve doğrulama kaybı açısından karşılattırıldığında, RoBERTa modeli, en iyi performans göstermiş olup BERT modelin ikinci sıra aldığını gözlemlenmiştir.

Ayrıca, Çizelge 5.14'teki sonuçları, Roberta modelinin öznitelik çıkarıcı olarak kullanıldığında diğer modellere göre daha başarıl olduğunu de göstermiştir. Son olarak, Çizelge 5.15 ve Şekil 5.11'de görüldüğü gibi RoBERTa tabanlı modelin farklı Dropout değerleri ile değerlendirildiğinde, Dropout değeri 0.2'yi olarak ayarlandığında en iyi sonuçları elde edilmiştir.
Önerilen modelin performansını değerlendirmek ve veri temsil yöntemi önemini göstermek için, önerilen sinir ağı mimarisini farklı veri temsil yöntemleriyle melezleştirilerek test edilmiştir. Elde edilen sonuçlar, Çizelge 5.16'da görülebileceği gibi, kelime temsil yöntemin düzgün seçilmesinin, sınıflandırıcı performansı üzerindeki etkili olduğunu doğrulamıştır ve önerilen bağlamsal kelime temsili tabanlı

yöntemin, diğer geleneksel temsil tabanlı yöntemlerden daha iyi performans verdiğini göstermiştir.

Analiz çalışmasına bakıldığında, dropout değeri 0.25 olarak ayarlandığında tüm veri setleri kullanılarak en iyi sınıflandırma doğruluğu elde edilmiştir. Dolaysıyla, BLSTM katman sayısı artırılırken dropout değeri 0.25 olarak ayarlanmıştır. Birçok deneylerden sonra, üstün sonuçlar vermek için iki yığılmış BLSTM katmanının yeterli olduğunu gözlemlenmiştir.

Ayrıca, diğer popüler kelime temsil yöntemleri ile karşılaştırmak için, önerilen modelde kullanılan mimarisi, GloVE, FastText ve word2vec gibi üç geleneksel Word embedding yöntemi ile birleştirilerek test edilmiştir.

Çizelge 5.19 ve Şekil 5.12'deki sonuçlardan, kelime temsil yönteminin sınıflandırma sürecinde önemli bir rol oynadığı sonucuna varılabilmektedir. Örneğin, diğer geleneksel Word embedding yöntemlerin yerine RoBERTa tabanlı yöntemi kullanıldığında, doğruluk sonuçların Twitter veri seti için yaklaşık %4-10, YouTube veri seti için %1-8 ve SMS veri seti için ise %1-2 oranında iyileştiğini gözlenmiştir.

Dördüncü çalışmada, bağlamsal gömmeyi (özellik çıkarıcı olarak ALBERT modelini kullanarak) RNN tabanlı derin öğrenme mimarisiyle melezleştiren bir model önerilmiştir. Önerilen modelin, kullanılan üç kıyaslama veri kümesi kullanılarak test edildiğinde üstün sonuçlar verdiğini görülmüştür. Çizelge 5.20'de görüldüğü gibi önerilen model Twitter, YouTube ve SMS veri setleri kullanılarak test edildiğinde sırasıyla %98, %95 ve %99'un üzerinde bir doğruluğa ulaşmıştır.

Hiper parametre ince ayarı (Fine-tuning), önerilen modelin en iyi performansına ulaşmada önemli bir rol oynamaktadır. Bu çalışmada önerilen mimarinin hiper parametrelerini ayarlayarak, modelin iyi bir performans vermesi için BLSTM katman sayısının 3 katman geçmemesi gerektiğini bulunmuştur. Ayrıca, üstün sonuçlara ulaşmak için öğrenme oranının mükemmel değerinin 0.001'e ve bırakma değerinin 0.2'ye eşit olarak ayarlanması gerektiğini bulunmuştur. Yoğun (Dense) katmanlardaki aktivasyon fonksiyonu ise deneyler sırasında ReLu olarak seçilmiştir. Problemimiz ikili sınıflandırma problemi olduğu için çıkış katmanında varsayılan aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanılmıştır.

6.1. Sonuç ve Gelecekteki Çalışmalar

Spam mesaj sorunu, sosyal ağlardaki yaygın sorunlardan biridir. Bu mesajların algılanması kısa metin sınıflandırma problemlerine girmektedir. Sosyal paylaşım sitelerindeki kısa metinlerin seyrekliği ve belirsizliği nedeniyle, Spam algılama sorunu, doğal dil işleme alanında zorlu bir görev olarak kabul edilmektedir. Bu nedenle, sınıflandırmanın etkinliğini arttırmanın en önemli adımı, kısa metin için sağlam bir temsili kullanmaktır.

Geleneksel Word embedding modelleri, yoğun vektörlerle kelimeleri temsil ederek veri seyreklik problemi çözmektedir, ancak bu modellerin bazı problemlere maruz kaldığından dolayı kısa metin etkili bir şekilde ele almalarını engellenebilmektedir. Problemlerin birincisi, modelin sözlüğünde bulunmayan kelimeler için herhangi bir vektör temsili sağlayamadığı "out of vocabulary" problemi olarak kabul edilmektedir. İkinci problem ise modellerin, cümle içindeki konumundan bağımsız olarak her bir kelimenin yalnızca bir vektör ile temsil ettiğini bağlamdan bağımsızlık problemi olarak kabul edilmektedir.

Bu problemlerin üstesinden gelmek için, bu tez çalışmasında, modern doğal dil işleme modelleri özellikle metin bağlamsal temsilleri, derin öğrenme teknikleri ile hibirtleştirilerek sosyal ağlardaki spam mesajlarını tespit etme yönelik farklı yaklaşımlar önerilmiştir.

Bu çalışmada yapılan deney sonuçları, kelime temsilinin, sosyal ağlarda olduğu gibi kısa metinleri işlemek için ne kadar önemli olduğunu bariz bir şekilde fark edilebilmektedir.
Ayrıca, önerilen modellerin sonuçları diğer yaklaşımlarla karşılaştırıldığında, önerilen metin temsili yöntemlere dayalı modellerin üstün başarılı gösterdiğini gözlemlenmiştir.

Gelecekteki çalışmalarda, spam algılama alanında mesaj içeriğinin yanı sıra başka özellikleri de kullanmayı amaçlanmaktadır. Ayrıca, Emoji kullanmanın algılama süreci üzerindeki etkisini de incelemeyi hedeflenmektedir. Üstelik çeşitli gelişmiş NLP modelleri test edip kullanılabilmekte ve önerilen modellerin yerine daha gelişmiş derin öğrenme mimarisi de kullanılabilmektedir. Öte yandan, resimlere gömülü spam metinlerini tespit etmek için bir yöntem de geliştirilebilmektedir. Son olarak, önerilen

yöntemler, duygu analizi (sentiment analysis), review spam detection, yazar kimliği tespiti (authorship identification), nefret söylemi tespiti (hate speech detection), review spam tespiti (review spam detection) vb. gibi diğer yaygın alanlarda da test edilecektir.

KAYNAKLAR

Adewole, K. S., vd. (2020). "Twitter spam account detection based on clustering and classification methods." The Journal of Supercomputing **76**(7): 4802-4837.

Aiyar, S. ve N. P. Shetty (2018). "N-gram assisted youtube spam comment detection." Procedia computer science **132**: 174-182.

Alberto, T. C., vd. (2015). Tubespam: Comment spam filtering on youtube. 2015 IEEE 14th international conference on machine learning and applications (ICMLA), IEEE.

Alom, Z., vd. (2020). "A deep learning model for Twitter spam detection." Online Social Networks and Media **18**: 100079.

Alsaffar, D., vd. (2019). Machine and deep learning algorithms for Twitter spam detection. International Conference on Advanced Intelligent Systems and Informatics, Springer.

Ameen, A. K. ve B. Kaya (2018). Spam detection in online social networks by deep learning. 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), IEEE.

Barushka, A. ve P. Hajek (2020). "Spam detection on social networks using cost-sensitive feature selection and ensemble-based regularized deep neural networks." Neural Computing and Applications **32**(9): 4239-4257.

Bojanowski, P., vd. (2017). "Enriching word vectors with subword information." Transactions of the Association for Computational Linguistics **5**: 135-146.

Brooke Auxir , M. A. (2021). Social Media Use in 2021, Pew Research Center.

Carson, J. M., vd. (2020). "Artificial intelligence approaches to predict coronary stenosis severity using non-invasive fractional flow reserve." Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine **234**(11): 1337-1350.

Chaudhary, V. ve A. Sureka (2013). Contextual feature based one-class classifier approach for detecting video response spam on youtube. 2013 Eleventh Annual Conference on Privacy, Security and Trust, IEEE.

Chen, C., vd. (2016). "Statistical features-based real-time detection of drifted twitter spam." IEEE Transactions on Information Forensics and Security **12**(4): 914-925.

Chen, W., vd. (2015). Real-time twitter content polluter detection based on direct features. 2015 2nd International Conference on Information Science and Security (ICISS), IEEE.

Chen, W., vd. (2017). "A study on real-time low-quality content detection on Twitter from the users' perspective." PloS one **12**(8): e0182487.

Chowdury, R., vd. (2013). A data mining based spam detection system for youtube. Eighth international conference on digital information management (ICDIM 2013), IEEE.

Clark, K., vd. (2020). "Electra: Pre-training text encoders as discriminators rather than generators." arXiv preprint arXiv:2003.10555.

Concone, F., vd. (2019). Twitter Spam Account Detection by Effective Labeling. ITASEC.

Devlin, J., vd. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

El-Mawass, N. ve S. Alaboodi (2015). "Hunting for spammers: Detecting evolved spammers on twitter." arXiv preprint arXiv:1512.02573.

Ghanem, R. ve H. Erbay (2020). "Context-dependent model for spam detection on social networks." SN Applied Sciences **2**(9): 1-8.

Ghanem, R. ve H. Erbay (2022). "Spam detection on social networks using deep contextualized word representation." Multimedia Tools and Applications: 1-16.

Gupta, H., vd. (2018). A framework for real-time spam detection in Twitter. 2018 10th International Conference on Communication Systems & Networks (COMSNETS), IEEE.

Ha, C., vd. (2019). "Eliminating overfitting of probabilistic topic models on short and noisy text: The role of dropout." International Journal of Approximate Reasoning **112**: 85-104.

Hidalgo, J. M. G. ve U. Zurutuza (2018). Novel Comment Spam Filtering Method on Youtube: Sentiment Analysis and Personality Recognition. Current Trends in Web Engineering: ICWE 2017 International Workshops, Liquid Multi-Device Software and EnWoT, practi-O-web, NLPIT, SoWeMine, Rome, Italy, June 5-8, 2017, Revised Selected Papers, Springer.

Imam, N., vd. (2019). "A Semi-Supervised Learning Approach for Tackling Twitter Spam Drift." International Journal of Computational Intelligence and Applications **18**(02): 1950010.

Inuwa-Dutse, I., vd. (2018). "Detection of spam-posting accounts on Twitter." Neurocomputing **315**: 496-511.

Jain, G., vd. (2019). "Optimizing semantic LSTM for spam detection." International Journal of Information Technology **11**(2): 239-250.

Jain, G., vd. (2019). "Spam detection in social media using convolutional and long short term memory neural network." Annals of Mathematics and Artificial Intelligence **85**(1): 21-44.

Kabakus, A. T. ve R. Kara (2019). “TwitterSpamDetector: A Spam Detection Framework for Twitter." International Journal of Knowledge and Systems Science (IJKSS) **10**(3): 1-14.

Kandasamy, K. ve P. Koroth (2014). An integrated approach to spam classification on Twitter using URL analysis, natural language processing and machine learning techniques. 2014 IEEE Students' Conference on Electrical, Electronics and Computer Science, IEEE.

Kanodia, S., vd. (2018). A novel approach for youtube video spam detection using markov decision process. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE.

Karakaşlı, M. S., vd. (2019). Dynamic Feature Selection for Spam Detection in Twitter. International Telecommunications Conference, Springer.

KEMP, S. (2021). 103. Kemp, S. (2021)."Digital 2021: Turkey", https://datareportal.com/reports/digital-2021-turkey, P. 16-103.

Kingma, D. P. ve J. Ba (2014). "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980.

Kumar, A., vd. (2019). Fuzzy string matching algorithm for spam detection in twitter. International Conference on Security & Privacy, Springer.

Lan, Z., vd. (2019). "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942.

Lee, S. ve J. Kim (2014). "Early filtering of ephemeral malicious accounts on Twitter." Computer Communications **54**: 48-57.

Liu, Y., vd. (2019). "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692.

Madisetty, S. ve M. S. Desarkar (2018). "A neural network-based ensemble approach for spam detection in Twitter." IEEE Transactions on Computational Social Systems **5**(4): 973-984.

Mateen, M., vd. (2017). A hybrid approach for spam detection for Twitter. 2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST), IEEE.

McCann, B., vd. (2017). "Learned in translation: Contextualized word vectors." arXiv preprint arXiv:1708.00107.

Mikolov, T., vd. (2013). "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781.

Miller, Z., vd. (2014). "Twitter spammer detection using data stream clustering." Information Sciences **260**: 64-73.

Pennington, J., vd. (2014). Glove: Global vectors for word representation. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).

Raj, R. J. R., vd. (2020). A Multi-classifier Framework for Detecting Spam and Fake Spam Messages in Twitter. 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), IEEE.

Samsudin, N. A. M., vd. (2019). "Youtube spam detection framework using naïve bayes and logistic regression." Indonesian Journal of Electrical Engineering and Computer Science **14**(3): 1508-1517.

Sanh, V., vd. (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." arXiv preprint arXiv:1910.01108.

Sedhai, S. ve A. Sun (2017). "Semi-supervised spam detection in Twitter stream." IEEE Transactions on Computational Social Systems **5**(1): 169-175.

Sun, N., vd. (2020). "Near real-time twitter spam detection with machine learning techniques." International Journal of Computers and Applications: 1-11.

Tajalizadeh, H. ve R. Boostani (2019). "A novel stream clustering framework for spam detection in Twitter." IEEE Transactions on Computational Social Systems **6**(3): 525-534.

Tur, G. ve M. N. Homsi (2017). Cost-sensitive classifier for spam detection on news media Twitter accounts. 2017 XLIII Latin American Computer Conference (CLEI), IEEE.

Uysal, A. K. (2018). "Feature selection for comment spam filtering on YouTube." Data Science and Applications **1**(1): 4-8.

Vaswani, A., vd. (2017). Attention is all you need. Advances in neural information processing systems.

Wang, J., vd. (2017). Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. IJCAI.

Wang, X., vd. (2019). "Drifted Twitter spam classification using multiscale detection test on KL divergence." IEEE Access **7**: 108384-108394.

Watcharenwong, N. ve K. Saikaew (2017). Spam detection for closed Facebook groups. 2017 14th International Joint Conference on Computer Science and Software Engineering (JCSSE), IEEE.

Wu, T., vd. (2017). Twitter spam detection based on deep learning. Proceedings of the australasian computer science week multiconference.

Wu, T., vd. (2017). "Detecting spamming activities in twitter based on deep-learning technique." Concurrency and Computation: Practice and Experience **29**(19): e4209.

Wu, T., vd. (2018). "Twitter spam detection: Survey of new approaches and comparative study." Computers & Security **76**: 265-284.

Yang, C., vd. (2013). "Empirical evaluation and new design for fighting evolving twitter spammers." IEEE Transactions on Information Forensics and Security **8**(8): 1280-1293.

Zheng, X., vd. (2015). "Detecting spammers on social networks." Neurocomputing **159**: 27-34.

ÖZGEÇMİŞ

Adı Soyadı	:Rezan BAKIR
Doğum Tarihi	:
Yabancı Dil	:İnglizce

Eğitim Durumu:

Lisans: Şam Üniversitesi / Otomasyon ve Bilgisayar mühendisliği Bölümü, 2009

Yüksek Lisans: Şam Üniversitesi /Ağlar ve Bilgisayar mühendisliği Bölümü, 2015

Çalıştığı Kurum/Kurumlar ve Yıl/Yıllar :

Software Developer	AE Enerji ANONIM şirketi	2022- devam
Sistem ve ağ yöneticisi	Şam Üniversitesi / Elasad Üniversite Hastanesi	2012- 2015
Bilgisayar eğitimi bölüm başkanı	Fırat barajı genel organizasyonu	2010-2012

Yayınları (SCI) :

Ghanem, R., & Erbay, H. (2022). Spam detection on social networks using deep contextualized word representation. *Multimedia Tools and Applications*, 1-16.

Yayınları (Diğer) :

Ghanem, R., & Erbay, H. (2020). Context-dependent model for spam detection on social networks. *SN Applied Sciences*, *2*(9), 1-8.

Araştırma Alanları:

Derin öğrenme,Makine öğrenmesi, Doğal dil işleme, Veri madenciliği,Sosyal medya analizi,Siber güvenliği.