T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

ÇİZGE EVRİŞİMLİ SİNİR AĞLARI KULLANILARAK METİN SINIFLANDIRMA

Rukiye ÖZDEMİR TEKİR

DANIŞMAN
Doç. Dr. Abdurrahim AKGÜNDOĞDU

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Elektrik-Elektronik Mühendisliği Programı

Haziran, 2023

TEZ KABUL VE ONAYI

Rukiye ÖZDEMİR TEKİR tarafından, **Doç. Dr. Abdurrahim AKGÜNDOĞDU** danışmanlığında hazırlanan **"ÇİZGE EVRİŞİMLİ SİNİR AĞLARI KULLANILARAK METİN SINIFLANDIRMA"** başlıklı bu çalışma, jürimiz tarafından **05/06/2023** tarihinde yapılan sınav sonucunda **oy birliği** ile başarılı bulunarak **Yüksek Lisans Tezi** olarak kabul edilmiştir.

Tez Jürisi

		İmza	**Sonuç**
DANIŞMAN	Doç. Dr. Abdurrahim AKGÜNDOĞDU İstanbul Üniversitesi - Cerrahpaşa Elektrik-Elektronik Mühendisliği Bölümü		☒ Kabul ☐ Ret
ÜYE	Doç. Dr. Yasin ÖZÇELEP İstanbul Üniversitesi - Cerrahpaşa Elektrik-Elektronik Mühendisliği Bölümü		☒ Kabul ☐ Ret
ÜYE	Dr. Öğr. Üyesi Mesut ÇEVİK Altınbaş Üniversitesi Elektrik-Elektronik Mühendisliği Bölümü		☒ Kabul ☐ Ret

Aileme

BÜTÇE DESTEKLERİ

ÇİZGE EVRİŞİMLİ SİNİR AĞLARI KULLANILARAK METİN SINIFLANDIRMA

Bu tez çalışması için herhangi bir kurumdan bütçe desteği alınmamıştır.

TEŞEKKÜR

Tez çalışmam boyunca her anlamda emek ve katkılarından dolayı danışman hocam Sayın Doç. Dr. Abdurrahim AKGÜNDOĞDU'ya ve iş hayatındaki yoğunluğa rağmen tezimi yazabilmek için zaman yaratmama yardımcı olan iş arkadaşlarıma teşekkürü borç bilirim.

Maddi ve manevi her zaman yanımda olan desteklerini hiçbir zaman esirgemeyen aileme, özellikle gösterdiği sabır ve verdiği destekten ötürü sevgili eşim Süleyman TEKİR'e teşekkür ederim.

Haziran 2023 Rukiye ÖZDEMİR TEKİR

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

TABLO LİSTESİ

Sayfa No

SİMGE VE KISALTMA LİSTESİ

Simgeler	Açıklama
V	: Düğüm
E	: Kenar
G	: Çizge (Graf)
D	: TF-IDF değeri hesaplanan belge kümesi
N	: Toplam Belge Sayısı
Λ	: Özdeğerlerin köşegen matrisi
f(t, d)	: t kelimesinin d belgesindeki sayısı
|{d in D: t in d}|	: t kelimesinin geçtiği belge sayısı

Kısaltmalar	Açıklama
DDİ	: Doğal Dil İşleme
DVM	: Destek Vektör Makineleri
ESA	: Evrişimli Sinir Ağları
CNN	: Convolutional Neural Networks
NB	: Naive Bayes
RNN	: Tekrarlayan Sinir Ağları
UKSB	: Uzun Kısa Süreli Bellek
GCN	: Graph Convolutional Networks (Çizge Evrişimli Sinir Ağları)
ÇESA	: Çizge Evrişimli Sinir Ağları
TF-IDF	: Term Frequency-Inverse Document Frequency
RO	: Rastgele Ormanlar
ÇKA	: Çok Katmanlı Algılayıcı
GSİ	: Gizli Semantik İndeksleme
CBOW	: Continuous Bag-of-Words
GloVe	: Global Vektörler
GRU	: Gated Recurrent Unit
GBT	: Generative Pretrained Transformer
YSA	: Yapay Sinir Ağları

GNN : Çizge Sinir Ağları

LSTM : Long Short-Term Memory

vd. : ve diğerleri

vb. : ve benzeri

KNN : K-Nearest Neighboring

API : Application Programming Interface

SMV : Sosyal Medya Verileri

ÖZET

YÜKSEK LİSANS TEZİ

ÇİZGE EVRİŞİMLİ SİNİR AĞLARI KULLANILARAK METİN SINIFLANDIRMA

Rukiye ÖZDEMİR TEKİR

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Elektrik-Elektronik Mühendisliği Programı

Danışman : Doç. Dr. Abdurrahim AKGÜNDOĞDU

Metin sınıflandırma, doğal dil işlemede yaygın olarak kullanılan önemli bir görevdir. Bir metnin sınıfını belirlemek, metin verilerinin anlamını çözümlemek ve bu verilerden yararlı bilgiler elde etmek için gereklidir. Metin sınıflandırma; konu etiketleme, spam tespiti ve duygu analizi gibi uygulamalarda kullanılabilir. Metinlerin sınıflandırılması işlemi belirli anahtar kelimelere dayalı kural tabanlı yaklaşımlarla yapılabileceği gibi sınıflandırma için yapay zekâ temelli bazı sınıflandırma algoritmaları da kullanılabilir. Yapay zekâ temelli metin sınıflandırma algoritmaları, yapısal olmayan metin verilerini işleyebilmek için, metinlerin özelliklerini çıkararak sayısal bir vektör olarak temsil etmeyi gerektirir. Bu işleme vektör temsili denir.

Bu tez çalışmasında, metin sınıflandırma için kullanılan vektör temsil yöntemleri incelenmiş ve yapay zekâ temelli 4 algoritma ile iki Türkçe veri seti üzerinde deneyler yapılmıştır. Veri setlerinden biri TTC4900 adlı 7 kategorili bir haber metni veri seti, Sosyal Medya Verileri 2 kategorili bir duygu analizi veri setidir. Deneylerde Naive Bayes (NB), Evrişimli Sinir Ağları (ESA), Tekrarlayan Sinir Ağları (RNN) ve Çizge Evrişimli Sinir Ağları (GCN) modelleri kullanılmıştır. NB, ESA ve RNN modellerinde metinleri sayısallaştırmak için TF-IDF ve

Sayma Vektörleştirici (Kelime Çantası) yöntemleri kullanılmıştır. Bu modellerin performansları GCN modeli ile karşılaştırılmıştır.

GCN modeli ise metin sınıflandırma alanında yeni ve farklı bir yaklaşım sunmaktadır. Bu yaklaşımda metinler çizge yapısı şeklinde temsil edilerek, kelimeler arasındaki bağlantılar ve anlamsal ilişkiler öğrenilmeye çalışılmaktadır. Böylece, metin sınıflandırma problemi düğüm sınıflandırma problemine indirgenmektedir. Metinleri çizge yapısı şeklinde temsil etmek için, belgelerden oluşan büyük bir çizge oluşturulmaktadır. Bu çizgede düğümler kelimeleri veya belgeleri, kenarlar ise kelimeler veya belgeler arasındaki ilişkileri göstermektedir. Kenarların ağırlıkları ise TF-IDF veya PMI gibi istatistiksel yöntemlerle hesaplanmaktadır. Oluşturulan bu çizge yapısı GCN modeline girdi olarak verilmekte ve model parametreleri eğitilmektedir. Bu çalışmada GCN modeli eğitimi için 200 epoch tanımlanmış ancak erken durdurma mekanizması ile eğitimde erken durdurma sağlanmıştır.

Deney sonuçlarına göre, GCN modelinde TTC4900 veri seti için yüksek doğruluk sağlanmıştır. Sosyal Medya Verileri veri setinde diğer modellere nispeten doğruluk düşüktür. Ancak veri setinde eğitim verileri için farklı bir ayrım yapıldığında yüksek doğruluk oranına ulaşılmıştır. Bunun veri setindeki kayıtlardan ve ön işleme problemlerinden kaynaklandığı düşünülmektedir. Bu çalışmada elde edilen sonuçlar GCN modelinin metin sınıflandırma problemine etkili bir çözüm sunabileceğini göstermektedir.

Haziran 2023 , 76 sayfa.

Anahtar kelimeler: Metin Sınıflandırma, Çizge Evrişimli Sinir Ağları, Metin Madenciliği

ABSTRACT

M.Sc. THESIS

TEXT CLASSIFICATON USING GRAPH CONVOLUTIONAL NETWORKS

Rukiye OZDEMIR TEKIR

Istanbul University-Cerrahpaşa

Institute of Graduate Studies

Department of Electrical and Electronics Engineering

Electrical and Electronics Engineering

Supervisor : Assoc. Prof. Dr. Abdurrahim AKGUNDOGDU

Text classification is an important and commonly used task in natural language processing. Determining the class of a text is necessary to analyze the meaning of text data and obtain useful information from it. Text classification can be used in applications such as topic labeling, spam detection, and sentiment analysis. The process of classifying texts can be done using rule-based approaches based on specific keywords, as well as artificial intelligence-based classification algorithms. Artificial intelligence-based text classification algorithms require extracting the features of texts and representing them as a numerical vector to process non-structural text data. This process is called vector representation.

In this thesis study, vector representation methods used for text classification were examined, and experiments were conducted on two Turkish datasets using four artificial intelligence-based algorithms. One of the datasets is a 7-category news text dataset called TTC4900, and the other is a sentiment analysis dataset with 2 categories called Social Media Data. Naive Bayes (NB), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Graph Convolutional Neural Networks (GCN) models were used in the experiments. TF-IDF and Bag-

of-Words (Count Vectorizer) methods were used to digitize the texts in the NB, CNN, and RNN models. The performance of these models was compared with the GCN model.

The GCN model presents a new and different approach in the field of text classification. In this approach, texts are represented in the form of a graph structure, and the connections and semantic relationships between words are learned. Thus, the text classification problem is reduced to a node classification problem. To represent texts in the form of a graph structure, a large graph consisting of documents is created. In this graph, nodes represent words or documents, and edges represent relationships between words or documents. The weights of the edges are calculated using statistical methods such as TF-IDF or PMI. This graph structure is given as input to the GCN model, and the model parameters are trained. In this study, 200 epochs were defined for training the GCN model, but early stopping mechanism was used to stop the training early.

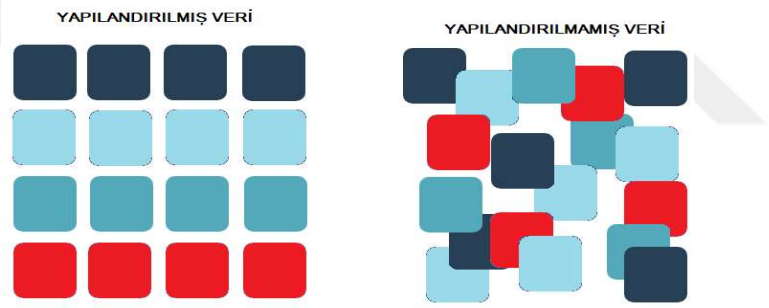
According to the experimental results, the GCN model achieved high accuracy for the TTC4900 dataset. The accuracy for the Social Media Data dataset is relatively low compared to other models. However, when a different distinction is made for the training data in the dataset, a high accuracy rate is achieved. This is thought to be due to the records in the dataset and preprocessing problems. The results obtained in this study show that the GCN model can provide an effective solution to the text classification problem.

June 2023, 76 pages.

Keywords: Text Classification, Graph Convolutional Neural Networks, Text Mining

1. GİRİŞ

İnternetin yaygınlaşması ve web teknolojilerinde yaşanan gelişmeler ile insanların global düzeyde iletişim kurmasını sağlayan sosyal ağ kavramı hayatımıza girmiştir. Son kullanıcıların internet ortamına veri gönderebilmesini sağlayan sosyal ağlar, her geçen gün internet üzerindeki resim ve metin verilerinde artışa sebep olmaktadır. Ayrıca günümüzde kütüphanelerdeki kitaplar ve ansiklopediler, bunların çok küçük bir oranı hariç, internet üzerinde yer almaktadır. Sosyal ağlar, forum sayfaları ve bahsi geçen kitaplar sayesinde, internet büyük bir bilgi kaynağıdır. Ancak internet üzerindeki bilgiler düzensiz olduğundan; internet, ulaşılmak istenen bilgiye ulaşmanın zor olduğu büyük veri yığını haline gelmiştir. Büyük veri yığınından bilgiye ulaşmak için veriler arasındaki ilişkilerin bilinmesi gerekir. Bu ilişkiye erişmenin ilk aşaması veriyi kategorilere ayırmaktır.



Şekil 1.1 Yapılandırılmış ve yapılandırılmamış veri

Veri tabanı tablosunda yer alan veriler gibi belirli formatta olan veriler, bilgisayar programları yardımıyla kolayca işlenebilir. Ancak, formatı belli olmayan metin verilerinin kategorilere ayrılmadan önce yapılandırılması gerekmektedir. Yapılandırılmış veriler, yazılımların ve insanların kolaylıkla erişim sağlayabileceği standartlaştırılmış tipteki verilerdir. Bu tür verilerdeki öznitelikler ayrıştırılmıştır. Öznitelikleri ayrıştırılmış veri sınıflandırma algoritmalarına girdi olarak verilebilir. Metin verilerinin yapısal hale getirilerek, içeriğine göre etiket ve kategorilere ayrılması metin sınıflandırma olarak adlandırılır. Metin sınıflandırma; duygu analizi, spam tespiti, konu etiketleme ve niyet tespitinde kullanılabilir. Metin sınıflandırması ticari işletmeler için de önemlidir. Pazarlama, reklam, müşteri ilişkileri yönetimi ve diğer işletme faaliyetleri için kullanılabilir.

Geleneksel metin sınıflandırma yöntemleri çoğunlukla kelime çantası (Bag-of-Words) yöntemini kullanan özellik mühendisliğine ve sınıflandırma algoritmalarına dayanır [1]. Metin sınıflandırmanın özellik mühendisliği ile ya da tamamen manuel bir şekilde yapılması zaman alıcı ve maliyetli olabilir. Bu nedenle, büyük veri kümeleri ile çalışılırken otomatik kategorizasyon yöntemleri kullanılmalıdır. Yapısal veriler, bilgisayar programları aracılığıyla otomatik olarak sınıflandırılabilir. Metin verileri yapısal değildir. Metin verilerinin işlenip özelliklerin çıkarılarak yapısal hale dönüştürülmesi için veri madenciliğinin bir alt alanı olan metin madenciliği ortaya çıkmıştır. Metin madenciliği, yapısal olmayan doğal dil metinlerinden yapısal veri çıkarma amacını taşır. Metin madenciliği ve doğal dil işleme (DDİ) çalışmaları birbirleriyle ilişkili alanlardır. DDİ, dilbilim bilgisi temelinde ilerlerken, metin madenciliği metinleri istatistiksel bir yaklaşımla analiz ederek sonuçlara ulaşmayı hedefler [2]. Ancak DDİ metin madenciliği alanında da kullanıma sahiptir. Metin madenciliği alanında DDİ özellik çıkarımı için kullanılabilir.

Metin madenciliğinin amacı, makinelerin anlayabileceği şekilde metni sayısal olarak ifade etmektir. Metin verisinden özellik çıkararak metni vektör gibi sayısal bir değere dönüştürebilmek için TF-IDF (Term Frequency-Inverse Document Frequency) ve kelime temsil yöntemleri gibi geleneksel DDİ yaklaşımları ve BERT gibi transfer öğrenme yöntemleri kullanılmaktadır. Bu sayısal verilerden bilgi keşfi için sınıflandırma ve kümeleme yöntemleri kullanılır. Sınıflandırma yöntemleri yapay zekâya dayalı yapılacaksa makine öğrenmesine dayalı geleneksel yöntemler ve derin öğrenme yöntemleri kullanılmaktadır. Destek vektör makineleri (DVM), K-En Yakın Komşu ve Karar ağaçları makine öğrenmesine dayalı geleneksel metin sınıflandırma yöntemlerindendir. Derin öğrenme yöntemleri ise Evrişimli Sinir Ağları (ESA), Tekrarlayan Sinir Ağları (RNN) ve Uzun Kısa Süreli Bellek (UKSB) gibi algoritmaların kullanıldığı sinir ağlarıdır. Bu yöntemler, metindeki ardışık kelimelerin anlamsal ve dil bilgisel özelliklerini etkili bir şekilde yakalayabilir. Ancak, metin sınıflandırma alanında kullanılan bir başka derin öğrenme yöntemi olan Çizge Sinir Ağları (GNN) metnin genel yapısını analiz etmek ve daha kapsamlı bir bakış açısıyla metni değerlendirmek için kullanılabilir [3]. Son yıllarda, Çizge Evrişimli Sinir Ağları (Graph Convolutional Networks - GCN) metin sınıflandırma alanında kullanılmaya başlanan bir GNN algoritması olarak karşımıza çıkmaktadır. Bu yaklaşım metni bir çizge olarak temsil etme fikrine dayanır. Bu yaklaşımda, her doküman bir çizge olarak düşünülür. Çizgenin düğümleri kelimeler veya cümlelerdir, çizge kenarları ise kelimeler veya cümleler arasındaki ilişkilerdir. Özellik çıkarma

yöntemleri ile oluşturulan çizge yapısı, çizge evrişimli sinir ağları ile işlenir ve metindeki kelimeler arası bağlamsal bilgiler yakalanır. Bu yaklaşım, kelimelerin sırasının önemli olmadığı ve kelimeler arasındaki ilişkilerin daha önemli olduğu durumlarda etkilidir. Çizge evrişimli sinir ağları, düğümlerin yerel ve global bağlamlarını dikkate alır. Bu nedenle, GCN kelime sırasının değil; kelimeler arası bağlamın önemli olduğu metin sınıflandırma görevleri için elverişlidir. Farklı uzunluktaki metinleri işleyebilmesi ve dokümandaki metin ve görüntüler gibi birden fazla modaliteden öğrenme yapabilmesi GCN'yi doğal dil işleme (DDİ-NLP) alanında önemli bir konuma getirir.

1.1. Tezin Amacı

İnternet, e-postalar, haberler, dijital kütüphaneler ve makaleler özelikleri belli olmayan yapılandırılmamış metin verileri içerir [4]. Bu tür kaynaklardan bilgiye kolayca erişebilmek için kaynaklardaki verilerin doğru bir şekilde sınıflandırılması gerekmektedir. Kümeleme ve sınıflandırma, bilgiye erişme yolunda ilk adımdır. Elektronik belgelerdeki bilginin ayıklanması ve sınıflandırılması araştırmacıların aradıkları bilgiye daha hızlı ve kolay bir şekilde erişimini sağlar. Bir makalenin okunmadan hangi alanda yazıldığının bilgisine sahip olmak, milyonlarca elektronik kaynak arasından aranılan bilgiyle ilgili kaynaklara ulaşmak metin sınıflandırmanın hedefidir.

Metinlerde otomatik sınıflandırma metin madenciliği, makine öğrenmesi ve derin öğrenme algoritmaları ile sağlanır. Metin madenciliği, doğal dil metninde yer alan yapılandırılmamış (metin) verilerin çeşitli yöntem, araç ve tekniklerin kullanılarak analiz edilmesidir [5]. Metin madenciliği metni yapılandırılmış verilere dönüştürmek için doğal dil işleme tekniklerini kullanır. Doğal Dil İşleme; dokümanların anlamsal olarak temsil edilmesini sağlayarak, dokümanların sınıflandırılması ve içerisinden bilgi çıkarılması gibi işlemleri gerçekleştirir. Anlamsal temsil, metnin bağlamını belirlemek için dil bilimsel kuralları ve mantıksal yapısını kullanarak metindeki cümleleri ve paragrafları analiz etmeyi içerir [6]. Cümleler ve paragraflardan isimler ve fiilleri ortaya çıkarmaktır [7]. İstatistiksel yöntemlere göre bu kelimelere bakılarak metin belirli kategorilere atanır. Metin sınıflandırmada kural tabanlı ve istatistiksel yaklaşımlar, kullanılabilecek kuralların kısıtlı olması ve dilin karmaşıklığı nedeniyle büyük metin verileri üzerinde yapılan çalışmalarda başarısız sonuçlar verebilir.

Verinin boyutunun, hızının ve çeşitliliğinin sürekli olarak artması metin verilerinin otomatik olarak sınıflandırılması gereksinimini ortaya çıkarmıştır. DVM ve Naive Bayes gibi makine öğrenmesi yöntemleri metin verilerinin sınıflandırılması konusunda başarılı sonuçlar vermiştir. Geleneksel makine öğrenmesi yöntemleri boyut azaltma, veri seyrekliği ve sınırlı genelleştirme kabiliyeti gibi dezavantajlara sahiptir [8]. Ayrıca makine öğrenmesi yöntemleri, belirli bir kelime dağılımına göre metinleri işlemekte ve metinlerin özniteliklerini belirlemektedir. Son yıllarda ESA, RNN, UKSB gibi derin öğrenme modelleri metin sınıflandırma alanında sıklıkla kullanılmaktadır [1]. Kelimelerin çizge gösterimi ardışık olmayan kelimelerin arasındaki anlamı tespit edebilme avantajına sahiptir [9].

Bu tez çalışmasının amacı, metin sınıflandırma konusunda mevcut çalışmaları gözden geçirmek ve Çizge Evrişimli Sinir Ağları (GCN) ile metin sınıflandırma yönteminin Türkçe metinlerdeki başarımını geleneksel metin sınıflandırma yöntemleri ile karşılaştırmaktır. Metin sınıflandırma yapılırken, sınıflandırma algoritmaları olarak Naive Bayes sınıflandırıcı, Evrişimli Sinir Ağları, Tekrarlayan Sinir Ağları ve Çizge Evrişimli Sinir Ağları kullanılacak olup TTC4900 ve Türkçe Sosyal Medya Verileri gibi Türkçe veri setleri ile modeller eğitilecektir.

1.2. Tezdeki Bölümler

Bu tez çalışmasındaki bölümler şu şekildedir:

Bölüm 2'de; metin madenciliğinden, metin sınıflandırmadan ve kullanılan yöntemlerden genel olarak bahsedilmektedir. Ayrıca Derin Sinir Ağları ve Çizge Sinir Ağları hakkında genel bilgiler verilmektedir.

Bölüm 3'te; metin sınıflandırma için bu tez çalışmasında kullanılan yöntemler açıklanmaktadır. Ayrıca oluşturulan model mimarileri karşılaştırılarak, modellerde kullanılan kayıp (loss) fonksiyonlarından bahsedilmektedir.

Bölüm 4'te; kullanılan yöntemin deneysel sonuçları, sayısal rakamlarla ifade edilip grafiklerle gösterilmektedir. Sınıflandırma performansı makine öğrenmesi ve derin öğrenmede kullanılan diğer metin sınıflandırma yöntemleri ile karşılaştırılmaktadır.

Bölüm 5'te; kullanılan modellerin metin sınıflandırmadaki yeri değerlendirilmektedir.

Bölüm 6'da; nihai sonuçlar yorumlanmaktadır.

2. KAVRAMSAL ÇERÇEVE

Sınıflandırma işlemi, verinin bilgiye dönüşme sürecinde verilerin analiz edilmesi ve özetlenmesi için temel bir adımdır. Verilerin otomatik olarak sınıflandırılması için istatistiksel yöntemler, kural tabanlı yöntemler, makine öğrenmesi ve derin öğrenme yöntemleri kullanılabilir. Metin verisi gibi yapısal olmayan verilerin makine öğrenmesi ve derin öğrenme yöntemleri ile sınıflandırılabilmesi için verinin yapısal hâle dönüştürülmesi gerekir. Bu nedenle metin verileri ön işleme tabi tutulmalıdır. Tezin bu kısmında metin sınıflandırmadan ve ön işlem adımlarından bahsedilecektir. Ayrıca metin sınıflandırmada en çok kullanılan son teknoloji (state-of-the-art) derin öğrenme yaklaşımlarından bahsedilecektir.

2.1. METİN MADENCİLİĞİ

Veri, sayılar, harfler, metinler, görüntüler veya sesler gibi farklı biçimlerde temsil edilebilir. Ancak veri tek başına anlamlı değildir. Ölçme, sayım, deney ve gözlem gibi çeşitli veri toplama yöntemleri ile elde edilen veri; sayısal ise nicel veri olarak adlandırılırken, sayısal olmayan veri nitel veri olarak adlandırılır.

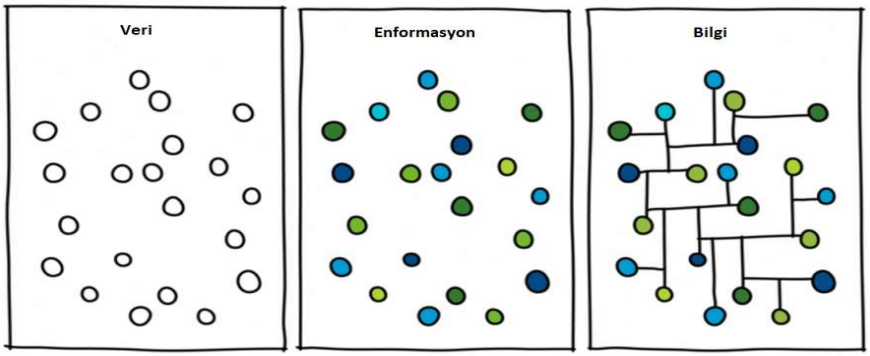
Veri yönetimi açısından iki tip veri vardır:

- **Yapılandırılmış Veri:** Belirli bir formatta düzenlenmiş ve kolayca işlenebilen verilerdir. Standart bir formatta olan bu tür verilere bilgisayarlar ve insanlar kolaylıkla erişim sağlar. Yapılandırılmış veriye; bir Excel tablosu, veri tabanı tablosu vb. veriler örnek olarak gösterilebilir.

- **Yapılandırılmamış Veri:** Düzenlenmemiş, şekilsiz ve işlenmesi zor verilerdir. Örnek olarak, bir metin belgesi, bir web sayfası, bir e-posta mesajı ve video dosyaları gibi veriler yapılandırılmamış veriye örnek olarak gösterilebilir.

Verilerin, bilgisayar yazılımları tarafından işlenebilmesi için yapılandırılmış olması gerekmektedir. Yapılandırılmamış veriyi yapılandırılmış hale dönüştürmek manuel bir şekilde olabileceği gibi algoritmalar ve yazılımlar aracılığıyla da yapılabilir.

Veriler yapılandırılsa bile tek başına bir anlam ifade etmezler. Anlam kazanabilmesi için verilerin enformasyon ya da bilgiye dönüşmesi gerekir. Verilerin analiz edilmesi, özetlenmesi, gruplanması belirli sınıflara ayrılması enformasyon olarak tanımlanabilir. Örnek olarak, haber verilerinin olduğu veri kümesinden bir verinin spor haberi olduğunu anlamak bir enformasyondur. Bilgi ise, enformasyonun daha geniş bir bağlama yerleştirilmesi, anlamlı bir ilişki kurulması ve bir anlam çıkarılmasıdır. Veri, enformasyon ve bilgi, bilgi yönetimi alanında önemli kavramlardır. Karar verme süreçlerinde bilgi ve enformasyon önemli rol oynar.



Şekil 2.1 Veri, enformasyon ve bilgi ilişkisi

Veriler, genellikle bir anlam taşımazlar ve işlenmemiş, organize edilmemiş veya anlamlandırılmamış olabilirler. Ancak, verilerin anlamlı hale getirilmesi, organize edilmesi ve işlenmesiyle bilgi ortaya çıkar. Bilgi, verilerin işlenmesi ve anlamlandırılmasıyla elde edilen sonuçtur.

Metin madenciliği; yapısal olmayan büyük metin yığınlarından önceden bilinmeyen, faydalı ve düzenli bilgilerin elde edilmesini esas alan bir veri madenciliği çalışmasıdır [10]. Büyük veri kümelerindeki metin verilerini otomatik olarak işlemek, analiz etmek ve anlamlı bilgiler çıkarmak için kullanılan bir dizi yöntem ve tekniklerin bütünüdür. Metin madenciliği, veri madenciliğinin bir parçası olarak düşünülmesine rağmen olay örüntülerinden değil de doğal dil metinlerinden çıkarım yapar [11].

Metin madenciliği veri madenciliğinde kullanılabilecek yapısal verilerin hazırlanmasından sorumludur. Yapısal olmayan metin verilerinin yapısal hale getirilmesi ve analiz edilebilir hale getirilmesi, veri madenciliğinde önemli bir adımdır. Metin madenciliği, doğal dil işleme, istatistik, veri madenciliği, yapay zekâ ve makine öğrenimi gibi disiplinlerle birlikte çalışır. İstatistiksel yöntemler, metin verilerinin analiz edilebilir hale getirilmesinde önemli bir rol oynar. Bu yöntemler, kelime sıklıkları, kelime vektörleri, n-gramlar gibi metin özelliklerini

kullanarak yapısal olmayan metin verisinden yapısal veriler elde edilmesine yardımcı olur. Bu yapısal veriler, bilgisayar yazılımları ve çeşitli makine öğrenmesi algoritmaları tarafından işlenebilir.

Metin madenciliği ve DDİ birlikte düşünülen iki alandır. Bu iki alanın malzemesi doğal dildir. Ancak Dilbilim ve Doğal Dil İşleme (DDİ), metinleri daha derinlemesine analiz etmek ve dilbilimsel açıdan temsil etmek amacıyla çeşitli hesaplama teknikleri kullanır. Metin madenciliği ise, metinden yüksek kaliteli bilgi çıkarma süreçlerine odaklanır [12]. Metin madenciliği, metin verilerinin içinde saklı olan anlamları ve yapıları keşfetmeye yardımcı olur. Bu sayede, büyük veri kümelerindeki anlamlı bilgilerin keşfedilmesi ve çıkarılması daha kolay hale gelir.

Metin madenciliği; sosyal medya analizi, pazar analizi, müşteri ilişkileri yönetimi, sahte bilgi yönetimi, duygu analizi, belge ve metin sınıflandırma gibi alanlarda kullanılabilir [13].

2.2. METİN SINIFLANDIRMA

Metin sınıflandırma, bir dizi metne birkaç farklı kategori atama işlemidir [14]. Metin sınıflandırması, yalnızca verinin sınıflara atanması işlemiyle sınırlı olmayıp, veri ön işleme, dönüştürme ve boyut azaltma gibi ek prosedür gerektiren bir süreçtir [15].

Metin sınıflandırma süreci genellikle aşağıdaki adımları içerir:

1. **Veri Toplama:** Metin sınıflandırma için ilk adım, sınıflandırma işleminde kullanılacak verilerin toplanmasıdır. Bu veriler genellikle etiketli veri setlerinden oluşur ve her bir belge için doğru kategorinin belirtildiği bir etiket içerir.

2. **Veri Ön İşleme:** Toplanan veriler genellikle metin ön işleme adımlarından geçirilir. Bu adımlar arasında metin temizleme (noktalama işaretlerini ve gereksiz karakterleri kaldırma), metin normalleştirme (kelimeleri küçük harfe dönüştürme), gereksiz kelimeleri kaldırma (sık kullanılan ancak anlamsız kelimeleri kaldırma) ve stemming (kelimelerin köklerini bulma) yer alır.

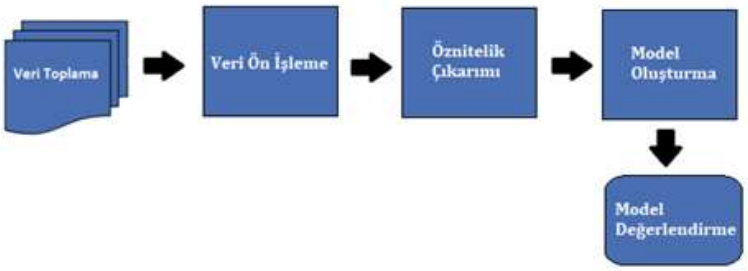
3. **Öznitelik Çıkarımı:** Ön işlemden geçirilen metinlerden öznitelikler çıkarılır. Bu öznitelikler genellikle kelime sayısı, kelime frekansı, kelime dağılımı gibi metin belgesinin yapısına ilişkin özellikleri içerir. Öznitelik çıkarımı için farklı yöntemler

kullanılabilir, en yaygın kullanılan yöntemlerden biri Kelime Çantası (Bag-of-Words) yöntemidir.

4. **Model Oluşturma:** Çıkarılan öznitelikler kullanılarak bir sınıflandırma modeli oluşturulur. Bu model, eğitim verilerindeki örnekleri kullanarak belirli bir kategoriyi tanımlayan kuralları öğrenir ve bu kuralları doğru bir şekilde uygulayabilen bir yapıya sahip olur.

5. **Model Değerlendirme:** Oluşturulan model, test verilerindeki örnekler için doğru tahminler yapabilme yeteneği açısından değerlendirilir. Bu değerlendirme için farklı metrikler kullanılabilir, en yaygın kullanılan metriklerden biri doğruluk (accuracy) metriğidir.

6. **Tahmin:** Oluşturulan ve değerlendirilen model, yeni metin belgelerini sınıflandırmak için kullanılabilir. Model, yeni bir metin belgesini alır ve bu belgenin hangi kategoriye ait olduğunu tahmin eder.



Şekil 2.2 Metin Sınıflandırma Süreci

Tek etiketli ve çok etiketli olmak üzere iki metin sınıflandırma türü vardır. Tek etiketli sınıflandırma metni ön tanımlı etiketlerden birine atar. Duygu analizi, spam filtreleme ve dil tespiti tek etiketli sınıflandırma örneğidir. Örneğin; bir e-posta ya spamdır ya da değildir. Çok etiketli sınıflandırma metni birden fazla sınıfa atar. Haber sınıflandırma, tıbbi teşhis ve sosyal medya gönderilerindeki etiketler çok etiketli sınıflandırmaya örnek olarak verilebilir.

Metin sınıflandırmada; kural tabanlı yaklaşımlar ve geleneksel makine öğrenmesi yöntemlerinin yanında derin öğrenme yöntemleri de kullanılabilir. Makine öğrenmesi, istatistiksel yöntemler ve derin öğrenme birbiriyle bağlantılı kavramlardır ancak literatürde bu yöntemlerin başlıklandırılması konusunda farklılıklar vardır. Bazı kaynaklar [16] istatistiksel yaklaşımı ayrı bir çatı altında toplamıştır. Bazı kaynaklarda ise, makine öğrenmesi yaklaşımları

istatistiksel yöntemler başlığı altında da incelenmektedir. Bu tez çalışmasında metin sınıflandırma yöntemleri, Metin Sınıflandırma Modelleri başlığı altında incelenecektir.

Metin sınıflandırmada, sınıflandırma yapılacak dil ön işleme adımlarında çok önemlidir. Ön işleme ve öznitelik çıkarımında çoğu yöntemin İngilizce dili için oluşturulmuş olması diğer dillerde metin sınıflandırmayı zorlaştırır. Ön işleme adımları, metin verilerini makine öğrenimi modeli tarafından anlaşılabilir bir forma dönüştürmeyi amaçlar. Bu adımlar arasında metin temizleme, kelime kökü bulma ve kelimeleri vektörleştirme işlemleri yer alır. Ancak bu adımların çoğu İngilizce dili için geliştirilmiştir ve diğer dillerde doğru sonuçlar vermeyebilir. Örneğin, kelime kökü bulma işlemi dilin yapısına bağlı olarak değişiklik gösterir. İngilizce dili için geliştirilen bir kelime kökü bulma algoritması, Türkçe gibi sondan eklemeli bir dilde doğru sonuçlar vermeyebilir. Aynı şekilde, öznitelik çıkarımı yöntemleri de dilin yapısına bağlı olarak değişiklik gösterir. Örneğin, n-gram modeli gibi yöntemler İngilizce dili için iyi sonuçlar verse de diğer dillerde aynı performansı göstermeyebilir. Bu nedenle, metin sınıflandırma yapılacak dilin yapısını anlamak ve bu yapıya uygun önişleme ve öznitelik çıkarımı yöntemlerini kullanmak çok önemlidir. Aksi halde, metin sınıflandırma modeli yanlış sonuçlar üretebilir.

Metin sınıflandırma, spam tespiti, konu modelleme ve duygu analizi uygulamalarında sıklıkla kullanılır. Bu tezde, konu modelleme ve duygu analizi üzerine iki sınıflı ve çok sınıflı metin sınıflandırma konuları üzerine çalışılmıştır.

2.2.1. Türkçe Metinlerin Sınıflandırılması

Türkçe ve Fince gibi sondan eklemeli bir dillerde metin sınıflandırmasını diğer dillere nispeten zordur. Çünkü Türkçede kelime kökleri genellikle sabit kalır ve anlam, kelimenin sonuna eklenen eklerle belirtilir. Fincede ise kök sabit kalırken gramer ekleri kelimenin sonuna eklenir. Bu nedenle, Türkçede veya Fincede bir kelimenin sonuna eklenen bir ek, kelimenin anlamını değiştirerek sınıflandırmasının sonucunu etkileyebilir. İngilizce bir cümlenin anlamı Türkçede sadece bir kelime ile verilebilir. Örneğin İngilizcede “I will not write” cümlesi Türkçede “write” kök olmak üzere “not” “will” “I” gibi ögeler kökün sonuna eklenerek “yazmayacağım” şeklinde yazılır [17]. Türkçenin dil yapısı kelimelerin doğru bir şekilde köklerine ve eklerine ayrılarak vektörel olarak temsil edilip metinlerin doğru bir şekilde sınıflandırılmasını zorlaştırmaktadır.

Geleneksel makine öğrenmesi yaklaşımları, ön eğitimli derin öğrenme modellerinden farklı olarak metnin dilinden bağımsız olarak sınıflandırma yapar. Örneğin, Gradyan Artırma

(Gradient Boosting) ve DVM gibi makine öğrenmesi yaklaşımları kelimelerle ve dille ilgilenmez [18]. Sadece ön işleme adımlarında kelimelerin köklerini bulma ve gereksiz kelimeleri kaldırma aşamasında dile özgü kurallar önemlidir. Metin ve kelimeler vektörize edildikten sonra dilsel bağımlılık ortadan kalkar.

Türkçe metin sınıflandırma alanında literatürde birçok çalışma bulunmaktadır. Bilinen en eski çalışmalardan biri Amasyalı ve Diri [19] tarafından yapılmıştır. Türkçe bir belgenin yazarının belirlenmesi, belge türüne göre sınıflandırma ve yazarın cinsiyetinin belirlenmesi gibi 3 (üç) farklı sınıflandırma çalışması yapılmıştır. Bu çalışmada, özellik çıkarma için n-gram modeli kullanılmıştır. Sınıflandırma algoritmaları olarak Naive Bayes, DVM, C 4.5 ve Rastgele Ormanlar (RO) Algoritmaları kullanılmıştır. Bu çalışmada bi-gram ile özellik çıkarma işlemi yapılarak ilerletilen çalışma tri-gram'a göre daha iyi sonuçlar vermiştir. Diğer bir çalışma Türkoğlu vd. [20] tarafından Türkçe dokümanların yazarını tespit amaçlı yapılmıştır. Yine n-gramlar kullanılarak 10 farklı özellik vektörü oluşturulmuştur. Naive Bayes, DVM, RO ve Çok Katmanlı Algılayıcı (ÇKA) sınıflandırma yöntemleri ile bu özellik vektörlerinin performansı test edilmiştir.

Çataltepe vd. [21] Türkçe kelimelerdeki kök uzunluğunun Türkçe metin sınıflandırmaya etkilerini incelemişlerdir. Kelimelerin anlamına bakmadan çeşitli teknikler ile daha kısa kökler elde etmişlerdir. TF-IDF yöntemi ile ağırlıklandırılan vektörlerin sınıflandırılma başarımlarını karşılaştırmışlardır. Kısaltılmış köklerle Centroid sınıflayıcılar ile daha iyi sonuçlar verdiğini görmüşlerdir.

Levent ve Diri [22], 3 farklı veri seti ile yapay sinir ağları kullanarak yazar tanıma üzerine çalışmışlardır. Yazarlık özelliği olarak 16 farklı özellik kullanmışlardır. Sistemin başarısı kesinlik (precision) ve duyarlılık (recall) ın harmonik ortalaması alınarak hesaplanan f-ölçüm üzeriden verilmiştir.

Açıkalın ve Beyazıt [23], Türkçe metinlerde ön işlemenin önemini anlatmak için bir çalışma yapmışlardır. Veri seti olarak konferans ve dergi makalelerinin özet kısımlarını kullanmışlardır. 5 uzunluğunda ön ek ve Zemberek ile kelime kökü bulma (stemmimg) yöntemlerini kullanmışlardır.

Ayata vd. [24], Türkçe tweetlerde sınıflandırma yapmışlardır. Word2vec ile vektör temsilleri oluşturulan tweetleri DVM ve RO sınıflandırıcıları ile sınıflandırmışlardır. Doğruluk oranları

Bankacılık sektörüne ilişkin tweetler için %89,97, Futbol tweetleri için %84,02, Telekom tweetleri için %73,86, perakende sektörüne ilişkin tweetler için %63,68 ve hepsi için %74,60 olarak gerçekleşmiştir.

2.3. ÖZELLİK ÇIKARMA

Ham metin verilerinin sınıflandırıcı tarafından tahmin yapmada kullanılabilecek bir özellik kümesine dönüştürülmesi işlemine özellik çıkarma denir. Belge uzunluğu, ortalama cümle uzunluğu ve noktalama işaretlerinin sayısı gibi farklı özellik çıkarma yöntemleri bulunmaktadır ancak bu bölümde özellik çıkarma yöntemlerinden kelime temsil yöntemleri anlatılmaktadır.

Kelime temsil yöntemleri, kelimeleri bir vektör uzayında temsil etmek için kullanılan yöntemlerdir. Bu yöntemlerde her kelime için bir vektör elde edilir ve bu vektörler kelimelerin anlamsal benzerliklerini yansıtır. Kelime temsil yöntemleri doğal dil işleme görevlerinde yaygın olarak kullanılmaktadır. Kelime temsil yöntemleri aşağıdaki şekilde gruplara ayrılabilir:

Frekans Bazlı Kelime Temsil Yöntemleri: One-Hot Encoding, N-gram, Sayma Vektörleştirici (Kelime Çantası), TF-IDF

Tahmine Dayalı Kelime Temsil Yöntemleri: Word2Vec, FastText

Ön Eğitimli Kelime Temsil Yöntemleri: GloVe, Bert, GPT

2.3.1. N-gram

N-gram, metin verilerinde kullanılan bir özellik çıkarma yöntemidir. Bu yöntem, metnin n karakter veya kelime uzunluğundaki ardışık gruplarını (n-gramlarını) oluşturarak metnin yapısını ve özelliklerini temsil eder. N-gramlar doğal dil işlemede gelecek metin veya konuşmayı tahmin etmenin bir yolu olarak kullanılır. Örneğin, “Kitap okumayı severim” şeklinde bir kelime dizisinde “kitap” ve “okumayı” kelimeleri verildiğinde sonraki kelimenin ne olacağını tahmin etmek için kullanılabilir. Ayrıca, dil modellemesi, metin sınıflandırma, yazım denetimi, konuşma tanıma ve makine çevirisi gibi alanlarda da kullanılır. Metin sınıflandırma alanında, n-gramların frekansı sayılarak kelime vektörleri oluşturulur.

N-gramdaki “n”, bir metin veya konuşma örneğinden alınan öğelerin ardışık bir dizisindeki öğe sayısını belirtir. “Ben bugün okula gittim” cümlesinin kelime birimine göre 2-gramları

şunlardır: “Ben bugün”, “bugün okula”, “okula gittim”. 3-gramları ise şunlardır: “Ben bugün okula”, “bugün okula gittim”.

Ekler kısmındaki Ek-1’de N-gram’ın bazı çalışma alanlarında kullanımına dair örnek tablo verilmiştir.

2.3.2. One-Hot Encoding

En basit kelime vektör temsili yöntemidir. Bu yöntem ile X boyutunda bir kelime yığını alınır. Kelime yığınındaki her kelimeye bir sayısal indeks atanır. Vektör boyutu X olarak ayarlanır ve X boyutlu vektörde indekse karşılık gelen sayıya sütuna 1 diğer sütunlara diğer sütunlara 0 atanır [25].

One-Hot Encoding, kategorik değişkenleri sayısal verilere dönüştürmek için kullanılan bir yöntemdir. Bu yöntem, kategorileri birbirinden ayıran sütunlardan oluşan bir matris oluşturur. Her sütun, bir kategoriye karşılık gelir ve o kategorinin var olup olmadığını ifade eden 1 ve 0'lardan oluşur. One-Hot Encoding, metin madenciliği alanında kelime vektörlerini oluşturmak için kullanılabilir. Bu durumda, her kelime bir kategori olarak kabul edilir ve kelimelerin bulundukları sıraya göre bir One-Hot Encoding matrisi oluşturulur.

Örneğin, "cinsiyet" adlı bir kategorik değişkenimiz olduğunu varsayalım. Bu değişkenin iki kategorisi var: "erkek" ve "kadın". Bu durumda, one-hot encoding, "erkek" ve "kadın" için ayrı ayrı iki yeni sütun oluşturur. Veri kümesindeki her bir gözlem için, cinsiyetinin "erkek" veya "kadın" olması durumunda, ilgili sütunun değeri 1, diğer sütunların değeri ise 0 olarak ayarlanır.

Cinsiyet
Kadın
Erkek

Cinsiyet_Erkek	Cinsiyet_Kadın
0	1
1	0

Şekil 2.3 One-Hot Encoding örneği

One-Hot Encoding sayesinde, kelime gibi kategorik değişkenler sayısal değerlere dönüştürülür ve makine öğrenmesi modelleri tarafından işlenebilir hale getirilir. One-hot encoding aynı zamanda "dummy variable encoding" veya "binary encoding" olarak da adlandırılır.

One-hot encoding yöntemini vektör temsili yöntemi olarak kullanmanın dezavantajı; veri kümesindeki kategorik değişkenlerin sayısı çok fazlaysa, veri kümesi boyutunun çok büyük

hale gelmesidir. Bu durum, modelin eğitim süresini uzatabilir ve doğruluğu düşürebilir. Bu nedenle, çok sayıda kategorik değişken içeren veri kümelerinde, one-hot encoding yerine alternatif teknikler kullanılabilir.

2.3.3. Kelime Çantası (Bag-of-words)

Kelime çantası, metin verilerini temsil etmek için doğal dil işlemede yaygın olarak kullanılan bir yöntemdir. Bu yöntemde, bir belgede her kelimenin kaç kez geçtiği sayılır ve belge, her bir öğesi belirli bir kelimenin sayısına karşılık gelen bir vektör olarak temsil edilir. Bu temsilde kelimelerin sırası dikkate alınmaz. Kelime çantası yöntemi, metin sınıflandırma ve duygu analizi gibi çeşitli makine öğrenmesi görevleri için bir özellik çıkarma tekniği olarak kullanılabilir. Kelime çantası yönteminin dilbilimsel bağlamda kullanımına ilk olarak 1954 yılında Harris'in [26] makalesinde rastlanır.

Kelime çantası metodunun kullanımına örnek olarak, "Köpekler çok sadık hayvanlardır. Kediler de köpekler kadar sadık olabilir." cümlelerini ele alırsak:

Bu cümlelerin kelime çantası vektörü {"Köpekler":2, "çok":1, "sadık":2, "hayvanlardır":1, "Kediler":1, "de":1, "kadar":1, "olabilir":1} olur. Bu yöntemde dokümandaki her kelimenin sıklığı belirtilmiştir.

Kelime çantası yöntemi metin sınıflandırma alanında kullanıldığında, her bir doküman için bir kelime sıklık vektörü oluşturulur. Bu vektörler makine öğrenmesi algoritmalarına girdi olarak verilerek bir sınıflandırma modeli eğitilir.

2.3.4. Terim Frekansı-Ters Belge Frekansı (TF-IDF)

TF-IDF vektörleştirici, kelime sıklığı matrislerini oluşturmak için kullanılan bir vektörleştirme aracıdır. TF-IDF (Term Frequency-Inverse Document Frequency) istatistiklere dayanarak belge kümelerinden kelime dağılımlarını vektörize eder.

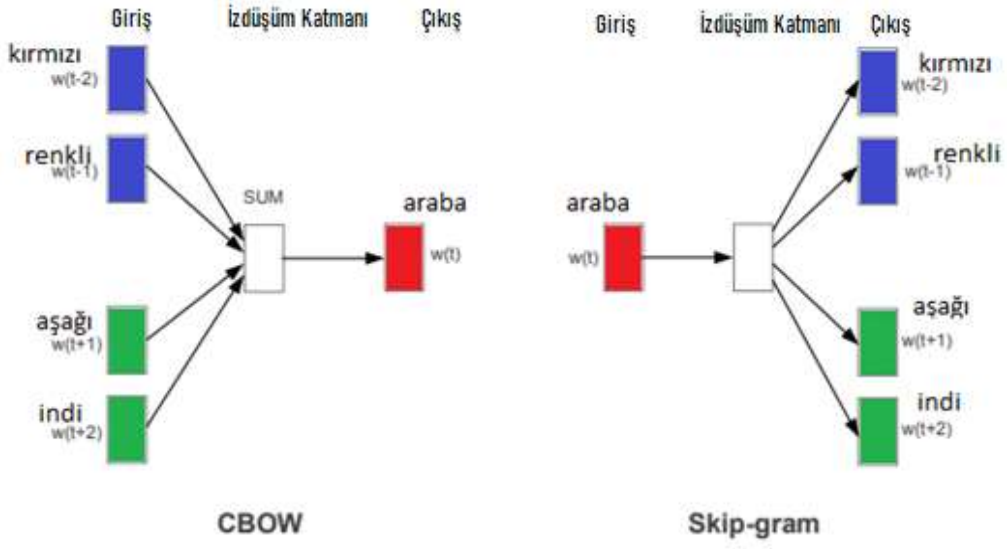
TF-IDF, bir belgedeki bir kelimenin önemini hesaplamak için kullanılan bir ölçüttür. TF, kelimenin belgedeki sıklığını belirlerken, IDF, kelimenin belgenin tamamındaki sıklığını veya nadirliğini belirler. Yüksek TF-IDF değeri, belgedeki kelimenin önemli olduğunu ve diğer belgelerdeki kelime dağılımıyla karşılaştırıldığında özgünlüğünü artırdığını gösterir. TF-IDF yönteminin ayrıntılı anlatımı 3. bölümdeki Yöntem başlığı altında yer almaktadır.

2.3.5. Word2Vec

Word2Vec, doğal dil işleme alanında kelime vektörleri oluşturmak için kullanılan yöntemlerden biridir. Bu yöntem, 2013 yılında Mikolov vd. [27] tarafından geliştirilmiştir. Kelimelerin anlamsal benzerliklerini ölçmek ve kelime ilişkilerini yakalamak amacıyla kullanılan Word2Vec, derin öğrenme ve yapay sinir ağları modellerine dayanmaktadır.

Word2Vec, Continuous Bag of Words (CBOW) ve Skip-Gram gibi iki farklı yöntemden oluşur. CBOW, bir cümledeki tüm kelimelerin bağlantılarından yararlanarak, bir kelimenin bağlamını tahmin etmeye çalışır. Örneğin, "araba" kelimesi verildiğinde, CBOW modeli, "araba" kelimesinin etrafındaki diğer kelimeleri kullanarak, cümle içindeki kelimenin bağlamını tahmin eder.

Skip-gram yöntemi ise; bir kelime verildiğinde, o kelimenin etrafındaki diğer kelimelerin olasılığını tahmin etmeye çalışır. Kelimenin çevresinde hangi kelimelerin kullanılabileceğini tahmin etmeye çalışır. Örneğin, "araba" kelimesi verildiğinde, Skip-gram modeli, "araba" kelimesinin etrafındaki diğer kelimelerin olasılığını tahmin eder.



Şekil 2.4 CBOW ve Skip-gram. CBOW bağlama göre kelimeyi tahmin eder. Skip-gram kelimenin çevresindeki kelimeleri tahmin eder. [27]

Word2Vec modeli, büyük veri kümeleri üzerinde eğitilir. Eğitim sırasında, model, veri kümelerindeki kelimeler arasındaki ilişkileri ve örüntüleri öğrenir. Modelin eğitimi tamamlandıktan sonra, her bir kelime için bir vektör oluşturulur. Bu vektör, kelimenin anlamsal özelliklerini ifade eder ve kelimenin bağlamsal benzerliklerini ölçmek için kullanılır.

Word2Vec, birçok doğal dil işleme görevinde kullanılabilir. Örneğin, kelime sınıflandırması, cümle benzerliği ölçümü, kelime önerisi gibi görevlerde başarılı sonuçlar verir. Ayrıca, kelime vektörlerini görselleştirerek, kelime ilişkilerinin anlaşılmasına yardımcı olur.

2.3.6. FastText

FastText, kelime temsili (embedding) yöntemlerinden biridir ve Facebook tarafından geliştirilmiştir. Temel olarak, kelime vektörleri oluştururken o kelimenin karakter dizisini de dikkate alır. Bu sayede, kelimenin köküne ek olarak içerdiği önemli karakterlerin de vektörleştirilmesi sağlanır. Bu özellik özellikle morfolojik olarak zengin diller için önemlidir, çünkü bu dillerde bir kelimenin farklı eklerle kullanımı ve anlamı değişebilir.

FastText, CBOW (Continuous Bag-of-Words) ve Skip-gram modellerinin temel prensiplerini kullanır. CBOW, bir kelimenin bağlamını verilerek o kelimenin tahmin edilmesini sağlar. Skip-gram modeli ise, bir kelime verildiğinde o kelimenin çevresindeki kelimelerin tahmin edilmesini sağlar. Ancak FastText, CBOW ve Skip-gram modellerinden farklı olarak, her kelimenin karakter n-gram'larını da modellemeyi içerir. Örneğin, "FastText" kelimesi "Fast", "ast", "stT", "Tex", "ext" gibi karakter n-gram'larına ayrılabilir ve bu n-gram'lar da kelime vektörlerinin oluşturulmasında kullanılabilir.

FastText'in avantajları arasında, morfolojik olarak zengin dillere uyarlanabilir olması, nadir veya hiç görülmemiş kelimelerin bile anlamlı vektörlere dönüştürülebilmesi, hızlı eğitim süreci ve düşük bellek gereksinimi sayılabilir.

2.3.7. GloVe

GloVe, Global Vektörler anlamına gelir. Stanford Üniversitesi [28] tarafından geliştirilen, kelime vektörlerini oluşturmak için bir kelimenin diğer kelimelerle olan dağılımını modelleyen matrisleri hesaplayarak çalışan bir denetimsiz öğrenme algoritmasıdır. Bu matrisler, kelimenin metindeki diğer kelimelerle olan ilişkisini temsil eder ve kelime vektörünün oluşumunda kullanılır. Bu sayısal vektörler, kelimenin anlamını ve kullanımını yansıtan bilgileri içerir. Glove vektör temsil yöntemi sinir ağı mimarisini kullanmaz [29]. Kelimelerin eş anlamlılıklarını ve birlikte bulunma istatistiklerini göz önüne alır. Temel mantık şudur: İki kelime çok kez bir arada kullanılıyorsa, bu iki kelime arasında bazı dilsel ve anlamsal benzerlikler vardır. GloVe yönteminin işleyiş mekanizmasını anlamak için örnek olarak, bir metinde "araba" kelimesinin geçtiğini varsayalım. GloVe, "araba" kelimesinin kullanımıyla ilgili matrisleri hesaplar. Bu matrisler, "araba" kelimesinin diğer kelimelerle olan bağlantısını

temsil eder. Matrislerdeki değerler, kelime vektörlerinin belirlenmesinde kullanılır ve sonuç olarak bir kelime vektörü elde edilir.

Tablo 2.1 GloVe Matrisi örneği. Örnek metinde kelimelerin bir arada kullanılma sıklığına göre Araba, benzin, yol ve hız arasındaki ilişkiyi verir.

	Araba	Benzin	Yol	Hız
Araba	0	0.3	0.2	0.1
Benzin	0.3	0	0.4	0.2
Yol	0.2	0.4	0	0.3
Hız	0.1	0.2	0.3	0

GloVe, kendinden önce geliştirilen kelime temsili yöntemlerine göre daha iyi performans gösterir ve özellikle büyük veri setleri için uygundur. GloVe, bir kelimenin bir belgedeki frekansı ve diğer kelimelerle olan ortak kullanımını bir araya getirerek kelime vektörlerini oluşturur. Bu sayede, kelime vektörleri, kelimenin anlamına uygun olarak benzer kelimelerle ilişkilendirilebilir. Oluşturulan kelime vektörleri, doğal dil işleme ve metin sınıflandırma gibi uygulamalarda kullanılabilir.

2.3.8. BERT

BERT (Bidirectional Encoder Representations from Transformers) derin öğrenme tabanlı bir doğal dil işleme modelidir. Google tarafından [30] 2018 yılında kullanıma sunulmuştur. BERT modeli, belgelerdeki kelimelerin birbirine olan ilişkilerini hem geçmiş hem de gelecek bağlamına göre dikkate alarak öğrenir. Bu sayede daha doğal ve anlamlı kelime vektörleri elde edilebilir.

BERT modeli, Transformer adı verilen bir sinir ağı mimarisini kullanır. Transformer mimarisi, dikkat mekanizmaları (attention mechanisms) adı verilen bir yapıya sahiptir. Bu yapı sayesinde model, belgelerdeki kelimelerin birbirine olan ilişkilerini daha iyi anlayabilir.

2.3.9. Üretici Ön Eğitimli Transformer

Üretici ön eğitimli transformer (Generative Pretrained Transformer – GPT), OpenAI tarafından geliştirilen bir dil modelidir. GPT, doğal dil anlamaya ve üretmeye yönelik bir yapay zekâ modelidir ve büyük miktarda metin verisi üzerinde eğitilmiştir. GPT, dil işleme görevlerinde yüksek performans göstermektedir ve metin üretme, metin tamamlama, makine çevirisi gibi görevlerde kullanılmaktadır.

GPT'nin mimarisi, dönüşümlü sinir ağı (transformer) mimarisine dayanmaktadır. Transformer mimarisi, dikkat mekanizmasını kullanarak girdi dizisindeki her bir öğenin diğer öğelerle ilişkisini modellemeyi amaçlar. Bu sayede, model uzun mesafeli bağımlılıkları daha iyi öğrenebilir ve daha doğru tahminler yapabilir. Dikkat mekanizmasına dayalı transformer mimarisini kullanan bir model çeviri yaparken her cümleyi tek tek çevirmez. Bunun yerine cümledeki her kelimenin diğer kelimelerle ilişkilerini dikkate alarak çeviri yapar. Örneğin; "Benim kedim var" cümlesi İngilizce'ye çevrilirken "Benim" kelimesini "I" olarak çevirebilir. Ancak "kedim" kelimesini "cat" olarak çevirmek yerine, "my cat" olarak çevirir. Çünkü model, "Benim" kelimesinin zaten "I" olarak çevrildiğini ve "kedim" kelimesinin aslında "my cat" anlamına geldiğini dikkat mekanizması sayesinde anlar.

GPT'nin eğitiminde büyük miktarda veri kullanılmıştır ve bu sayede model geniş bir dil bilgisi edinmiştir. Ayrıca, GPT sıfırdan eğitim yeteneğine sahiptir. Bu, modelin daha önce hiç görmediği bir görevi bile doğru bir şekilde yapabilmesini sağlar.

2.4. Metin Sınıflandırma Modelleri

2.4.1. Kural Tabanlı Yaklaşım Modeli

Metin sınıflandırmada kullanılan en eski metotlardan biri kural tabanlı sınıflandırmalardır. Kural tabanlı metin sınıflandırma, dil bilimsel kurallara göre elle oluşturulan metin sınıflandırma tekniğidir. Metnin yazıldığı dile özgü bir takım kurallar ve anahtar sözcükler oluşturulur. Bu anahtar sözcüklere ve kurallara göre metin kategorilere ayrılır. Bu yöntemin popülaritesi, şeffaflığından, sınıflandırmaların açıklanabilmesindeki kolaylıktan ve göreceli basitliğinden (destek vektör makineleri veya sinir ağları ile karşılaştırıldığında) kaynaklanmaktadır [31]. Bu yöntemin dezavantajı, dildeki karmaşıklık, dilde eş sesli kelimelerin yer alması ve kuralların sınırlı olması dolayısıyla yanlış sınıflandırma ihtimalinin yüksek olmasıdır.

2.4.2. Makine Öğrenmesi Yaklaşımı

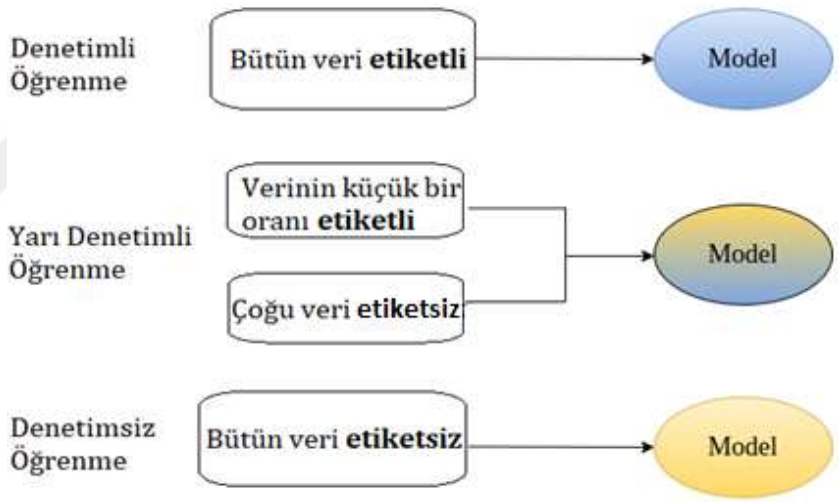
Makine öğrenmesi veri madenciliği alanında istatistiğin bir kolu olarak düşünülür [32]. Metin sınıflandırma alanında en çok kullanılan geleneksel makine öğrenmesi yöntemleri Naive Bayes, K-en Yakın Komşuluk, DVM gibi istatistiksel tabanlı yöntemlerdir [33]. Makine öğrenmesi ile metin sınıflandırma yapılırken çeşitli matematiksel ve istatistiksel hesaplardan ve algoritmalardan yararlanılır.

Makine öğrenmesi, denetimli (danışmanlı), denetimsiz (danışmansız) ve yarı denetimli öğrenme olarak üçe ayrılır.

Denetimli öğrenme; etiketli (kategorize edilmiş) verilerden bir model oluşturup, bu modele göre yeni verinin hangi sınıfa ait olduğunun tahminidir.

Denetimsiz öğrenmede, verilerde etiket bilgisi yoktur. Verilerin hangi sınıfa ait olduğuna makine öğrenmesi modeli karar verir. Faaliyet alanı, genellikle etiketsiz verilerin kümelenmesine yöneliktir.

Yarı denetimli öğrenme, hem etiketli hem de etiketsiz verilerin kullanıldığı bir makine öğrenmesi yöntemidir. Bu yöntem, genellikle etiketli verinin az olduğu durumlarda kullanılır. Yarı denetimli öğrenme algoritmaları, etiketsiz verileri de kullanarak modelin performansını artırmayı amaçlar.



Şekil 2.5 Denetimli, denetimsiz ve yarı denetimli öğrenme [34]

Denetimli öğrenme genellikle sınıflandırma ve regresyon analizi problemleri için kullanılırken denetimsiz öğrenme ise kümeleme problemleri için kullanılır.

Makine öğrenmesinde farklı sınıflandırma yöntemleri vardır: İkili sınıflandırma, Çok sınıflı sınıflandırma ve hiyerarşik sınıflandırma.

İkili sınıflandırmada veri örnekleri iki sınıfa ayrılır. Örneğin, bir e-postanın ‘spam’ veya ‘spam değil’ olarak sınıflandırılması ikili sınıflandırma örneğidir.

Çok sınıflı sınıflandırma, veri örnekleri birden fazla sınıfa ayrılır. Örneğin, bir hayvanın "kedi", "köpek" veya "kuş" olarak sınıflandırılması çok sınıflı sınıflandırmadır.

Hiyerarşik sınıflandırma, bu sınıflandırmada sınıflar bir hiyerarşi içinde düzenlenir ve veri örnekleri bu hiyerarşiye göre sınıflandırılır. Örneğin, bir hayvanın önce "memeli" veya "memeli değil" olarak sınıflandırılması, daha sonra "memeli" sınıfı içinde "kedi", "köpek" veya "inek" olarak alt sınıflandırılması hiyerarşik sınıflandırmaya örnektir.

Makine öğrenmesi yöntemleri, çok sınıflı metin sınıflandırmasında iyi performans gösterir. Çok sınıflı ve çarpık etiketli (dengesiz etiket dağılımı) veri setlerindeki sınıflandırmada makine öğrenmesi gibi istatistiksel yaklaşımların kullanılması geçmişte başarılı sonuçlar vermiştir [35]

2.4.2.1. *Naive Bayes Sınıflandırıcı*

Naive Bayes sınıflandırıcısı, Bayes teoremine dayalı bir makine öğrenmesi algoritmasıdır. Bu algoritma, verilen bir veri kümesindeki özelliklerin birbirinden bağımsız olduğu varsayımına dayanır [36]. Bu nedenle "naive" (naif) olarak adlandırılır.

Naive Bayes sınıflandırıcısı, olasılık teorisi kullanarak sınıflandırma yapar. Verilen bir veri noktasının her bir sınıfa ait olma olasılığını hesaplar ve en yüksek olasılığa sahip sınıfı tahmin olarak verir. Bu algoritma, özellikle metin sınıflandırma gibi doğal dil işleme uygulamalarında yaygın olarak kullanılmaktadır. Spam filtreleme, duygu analizi ve konu etiketleme gibi uygulamalar örnek olarak gösterilebilir.

Naive Bayes'in temelini oluşturan Bayes teoremi, iki olayın koşullu olasılıklarını ilişkilendiren bir teoremdir. Bayes teoreminin formülü aşağıdaki gibidir:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (2.1)$$

Burada P(A|B), B olayı gerçekleştiğinde A olayının gerçekleşme olasılığını; P(B|A), A olayı gerçekleştiğinde B olayının gerçekleşme olasılığını; P(A) ve P(B), A ve B olaylarının ayrı ayrı gerçekleşme olasılıklarını ifade eder.

2.4.3. Derin Öğrenme Yaklaşımı

Derin öğrenme, yapay sinir ağlarına dayalı bir makine öğrenmesi yöntemidir. Bu yöntem, büyük ve karmaşık veri kümelerini işleyerek anlamlı sonuçlar elde etmeyi amaçlar. Derin öğrenme yöntemleri, çok katmanlı yapay sinir ağlarından oluşur. Bu ağlar, girdi verilerini

işleyerek çıktı verilerini üretir. Yapay sinir ağlarında; her bir katman, girdi verilerini işleyerek daha yüksek seviyeli bir temsil elde eder. Bu sayede, derin öğrenme yöntemleri karmaşık veri yapılarını anlayabilir ve bu veriler üzerinde tahminler yapabilir. Derin öğrenme yöntemleri, özellikle görüntü ve ses işleme gibi alanlarda yaygın olarak kullanılmaktadır. Örnek olarak, görüntü sınıflandırma, nesne tanıma ve konuşma tanıma gibi uygulamalar gösterilebilir.

Derin öğrenme yöntemlerinin yapay zekâ alanında kullanılmasının en büyük avantajı, büyük ve karmaşık veri kümelerini işleyebilmesidir. Derin öğrenme yöntemleri, insanların bile anlamakta zorlandığı karmaşık veri yapılarını anlayabilir ve bu veriler üzerinde doğru tahminler yapabilir. Ayrıca, çok hızlı bir şekilde veri işleyebilir. Ancak derin öğrenme yöntemlerinin de bazı dezavantajları vardır. Özellikle, bu yöntemler çok sayıda parametreye sahip olduğu için aşırı uyum (overfitting) problemine karşı hassastır. Ayrıca, derin öğrenme yöntemlerinin eğitilmesi uzun zaman alabilir ve yüksek performanslı bilgisayar donanımları gerektirebilir.

2.4.3.1. *Evrişimli Sinir Ağları*

Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN), derin öğrenme alanında kullanılan bir yapay sinir ağı türüdür. Bu ağlar, özellikle görüntü işleme gibi alanlarda yaygın olarak kullanılmaktadır. Hayvanların görme yapısından ilham alınarak oluşturulmuştur.

Evrişimli sinir ağları (ESA), çeşitli alanlarda kullanılır. Bu alanlardan bazıları doğal dil işleme, biyomedikal, veri sınıflandırma, resim ve video işlemedir. Özellikle resim işleme konusunda ESA, iyi sonuçlar ortaya çıkarmıştır. Örneğin, Cireşan vd. tarafından yapılan bir çalışmada MNIST veri seti üzerinde ESA kullanılarak hata oranı %2'ye kadar düşürülmüştür. Ayrıca, 2014 yılında düzenlenen ImageNet yarışmasında milyonlarca görüntünün sınıflandırılmasında ESA algoritması kullanılarak büyük bir başarı elde edilmiştir [37] [38]. Google mühendisleri tarafından 12 katmanlı ön eğitimli ESA modeli ile tasarlanan AlphaGo adlı bir bilgisayar programı, profesyonel bir insan Go oyuncusunu yenen ilk bilgisayar programıdır. [39].

Evrişimli sinir ağları, girdi verilerini işleyerek çıktı üretir. Bu işlem sırasında, girdi verileri üzerinde evrişim (convolution) işlemleri uygulanır. Evrişim işlemi, girdi verilerinin yerel özelliklerini yakalamaya yardımcı olur. Örnek olarak, bir görüntüdeki kenarlar ve köşeler gibi yerel özellikler evrişim işlemi sayesinde tespit edilebilir. Evrişimli sinir ağları mimarisi, giriş katmanı, bir veya daha fazla evrişim katmanına sahip olan gizli katmanlar ve çıkış katmanı olmak üzere üç katmandan oluşur [40]. En az bir katman evrişim işlemini yapar. ESA'nın diğer

tipik katmanları: Evrişim katmanı, Aktivasyon katmanı, Havuzlama katmanı ve Tam Bağlantı katmanıdır.

2.4.3.2. *Tekrarlayan Sinir Ağları*

Tekrarlayan sinir ağları (Recurrent Neural Network - RNN), doğal dil işleme ve konuşma tanıma gibi sıralı verilerle çalışırken kullanılan yapay sinir ağı türüdür. RNN'ler, girdi verilerinin sırasını dikkate alarak önceki girdilerin bilgilerini saklar ve gelecekteki tahminlerde kullanır. Bu özellikleri sayesinde RNN'ler, zaman serisi verileri, metin verileri ve konuşma verileri gibi sıralı verilerle çalışırken oldukça etkilidir.

RNN'lerin temel yapısı, gizli katmanlardaki nöronların kendilerine geri beslemesi ile oluşur. Bu geri besleme sayesinde RNN'ler, önceki girdilerin bilgilerini saklayabilir ve gelecekteki tahminlerde kullanabilir. RNN'lerin en yaygın kullanım alanlarından biri doğal dil işleme alanıdır. Bir cümleyi anlamlandırmak için RNN'ler cümlenin kelime sırasını dikkate alarak her bir kelimenin anlamını ve cümlenin genel anlamını tahmin edebilir.

RNN'ler oldukça güçlü yapay sinir ağı türlerindendir ancak eğitimleri zor olabilir. Uzun süreli bağımlılıkları öğrenmekte zorlanabilirler ve bu nedenle UKSB(Uzun-Kısa Süreli Bellek) ve GRU (Gated Recurrent Unit) gibi daha gelişmiş RNN türleri geliştirilmiştir.

2.4.3.3. *Uzun-Kısa Süreli Bellek*

Uzun-Kısa Süreli Bellek, yapay sinir ağı türüdür ve RNN'lerin (Tekrarlayan Sinir Ağları) bir türüdür. UKSB'ler, RNN'lerin uzun süreli bağımlılıkları öğrenmekte zorlanmasını çözmek için geliştirilmiştir.

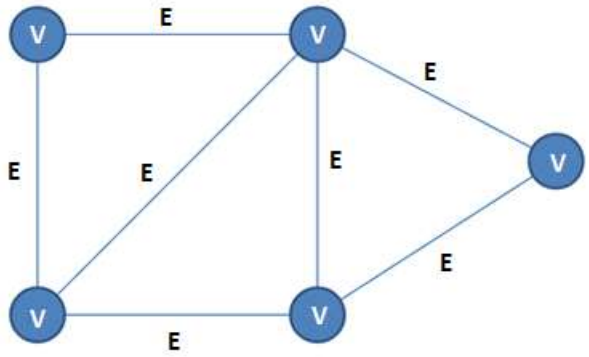
UKSB'lerin temel RNN'lerden farkı, hücre durumu adı verilen bir bileşene sahip olmalarıdır. Hücre durumu, UKSB ağının içinde bilgiyi taşıyan bir yapıdır ve ağın geçmiş bilgilerini saklar. UKSB hücrelerinde bulunan kapılar (gates) sayesinde hücre durumundaki bilgilerin ne kadarının saklanacağı veya unutulacağı kontrol edilir. Bu sayede UKSB'ler, uzun süreli bağımlılıkları öğrenebilir ve daha iyi tahminler yapabilir.

UKSB'ler, doğal dil işleme, konuşma tanıma ve zaman serisi tahmini gibi alanlarda oldukça etkilidir. Özellikle metin verileri ile çalışılırken UKSB kullanımı yaygındır.

2.4.3.4. *Çizge Sinir Ağları*

Çizge (Graf), düğüm ve kenarlardan meydana gelen bir veri yapısıdır. Nesneler arasındaki ilişkiyi modellemek için kullanılır. Bir çizge G=(V,E) şeklinde tanımlanabilir. Burada, V

düğümler, E kenarlardır. Düğümler kenarlar aracılığıyla bağlanır. Düğümler nesneleri temsil eder. Kenarlar da nesneler arasındaki ilişkilerdir. Yol haritaları, bilgisayar ağları, web sayfaları ve sosyal ağlar çizge şeklinde bulunabilir.



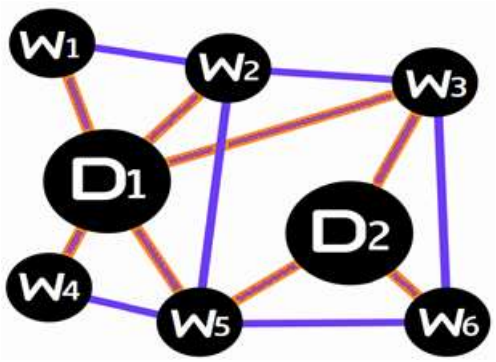
Şekil 2.6 Çizge veri yapısı. V: Düğümler, E: Kenarlardır [41].

Çizge veri yapısında tutulan verilere, Öklid dışı veriler denir. Öklid dışı veriler birçok alanda kullanılmaktadır [42]. Haberleşmede kullanılan sensör ağları, sosyal bilimlerde kullanılan sosyal ağlar ve ilaç keşfi için kullanılan protein etkileşim ağları Öklid dışı verilerin kullanım alanlarından bazılarıdır.

Metin verileri karakter veya kelime dizilerinden oluştuğu için bu veriler de Öklid dışı (Non-Euclidean) veriler olarak nitelendirilebilir. Bu verilerin Öklid dışı olmasının nedeni; genelde kelime çantası, kelime temsilleri veya metnin anlamsal anlamını yakalayan diğer yöntemlerle temsil edilmesi nedeniyle veri noktaları arasındaki Öklid mesafelerinin korunamamasıdır. Ancak metin verileri Öklid dışı veri olmasına rağmen sıralı verilerdir. Bu nedenle, metin gibi sıralı verileri modellemek için kelime temsil yöntemlerinin gelişmesi ile birlikte RNN ve ESA gibi derin öğrenme yöntemleri sıklıkla kullanılabilir. Bu yöntemlerin metin sınıflandırma konusunda başarıları ispatlanmıştır. Sıralı verilerin işlenmesine olanak tanıyan yöntemler ile başarılı sonuçlar alınmasına rağmen kelimeler arası bağlantılar ve cümle yapısındaki bilgiler çizge veri yapısı ile daha iyi bir şekilde ifade edilebilir. Metin verilerini çizge şeklinde temsil etmek kelimeler arasındaki ilişkileri anlamak için iyi bir yöntemdir ancak Öklid verileri ve sıralı veriler için kullanılan klasik derin öğrenme teknikleri, çizge verileri işleyemez. Bunun nedeni çizge yapısının düzensiz ve değişen komşuluklara sahip olmasıdır. Öklid dışı verilerin doğrudan yapay sinir ağları tarafından işlenerek, verideki özniteliklerin çıkarılması ihtiyacı Çizge Sinir Ağları'nın (Graph Neural Networks - GNN) ortaya çıkmasına neden olmuştur. Öklid dışı (çizge) halinde verilerden öznitelik çıkarmak çizge sinir ağları ve bu sinir ağlarının

bir türü olan çizge evrişimli sinir ağları kullanılmaktadır [43]. Örneğin, bir sosyal ağ grafındaki düğümlerin benzerliklerini ölçmek veya bir moleküler yapının özelliklerini tahmin etmek için GNN kullanılabilir.

Çizge sinir ağlarının bir üyesi olan GCN, Kipf ve Welling [44] tarafından 2017 yılında tanıtılmıştır. Kipf ve Welling, bilimsel yayınlar için bir atıf ağı olan Cora veri setini kullanarak yarı denetimli düğüm sınıflandırma işlemini gerçekleştirmiştir. Yao vd. [1], Kipf ve Welling [44] tarafından tasarlanan ve düğüm sınıflandırmayı esas alan GCN modelini metin sınıflandırma problemine uyarlamışlardır. Tasarladıkları metin tabanlı GCN modelini ve diğer derin öğrenme modellerini çeşitli metin veri setleri ile eğitmişlerdir. Bu modeller ile %68-97 arası test doğruluğu elde etmişlerdir. Bu başarım oranı deneylerinde karşılaştırdıkları ve aynı veri seti ile eğittikleri diğer derin öğrenme modellerinin başarımlarının üstündedir. Kullanılan bu yöntem metin sınıflandırma alanında başarılı olmuştur. Ancak bütün belgeleri (korpus) tek bir çizge olarak ifade ederek sınıflandırma yapmak bazı sorunlara neden olmaktadır. Bu sorunlardan bazıları, çizgenin çok sayıda kenar içermesi nedeniyle çizgenin oluşturulması aşamasında yüksek bellek tüketimi gerektirmesi ve çevrimiçi test yapmanın bu modelde zor olmasıdır. Çünkü oluşan çizge yapısı korpusa bağlıdır ve eğitim sonrası çizge yapısı değiştirilemez [45].



Şekil 2.7 Korpustan oluşturulan çizge (graf). Düğümler, kelime ve dokümanlardan oluşmaktadır. D1,D2:dokümanlar, W1,W2,W3,W4,W5,W6 kelimelerdir [46].

[1]'de bahsedilen modelde bütün kelimeler de grafta bir düğüm olarak gösterildiği için çizge oluşturulurken kelime-kelime arası ağırlıkları hesaplamak için yüksek miktarda bellek gerekir. Huang vd. [47] çizge oluşumu sırasında ortaya çıkan bellek tüketim sorununun üstesinden gelebilmek için metin seviyesinde graflar oluşturmuşlardır. Lin vd. [48] ise BERT modeli ile temsil ettikleri dokümanlar ile graf oluşturup BERTGCN adını verdikleri model ile bu graftan örüntü çıkarımı ile yüksek metin sınıflandırma başarımı elde etmişlerdir.

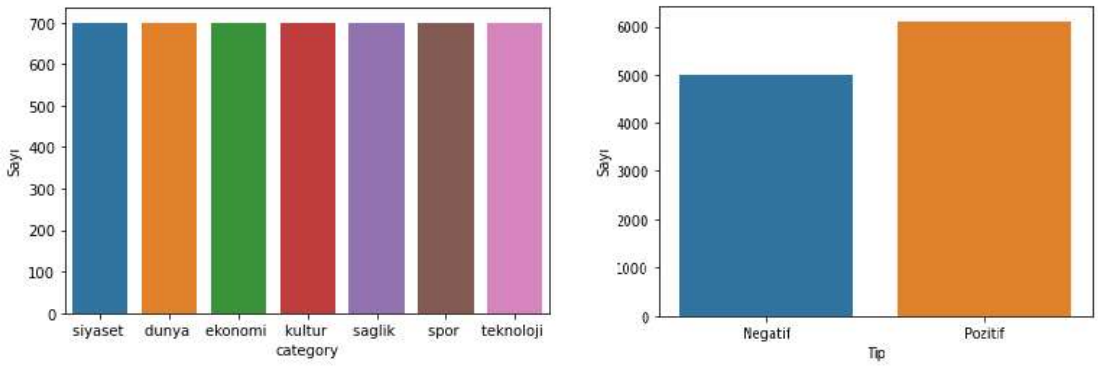
3. YÖNTEM

Tezde kullanılan yöntemler Şekil 3.1'de verilmiştir.

Şekil 3.1 Tez çalışmasında kullanılan yöntemler

3.1. Veri Seti

Veri seti olarak web üzerinde bulunan hazır veri setleri kullanılmıştır. Bu veri setlerinin her sınıftan %10'u test, %10'u doğrulama için ayrılmıştır. Veri setindeki her sınıftan verinin %80'i ile modeller eğitilmiş ve elde edilen bulgular Bölüm 4'te bulunan Bulgular başlığı altında paylaşılmıştır.



Şekil 3.2 Kullanılan TTC4900 (soldaki) ve Türkçe Sosyal Medya Verileri(sağdaki) veri setlerinin bar grafiği

3.1.1. TTC4900 Veri Seti

TTC4900, 4900 metin ve 7 sınıftan oluşan veri setidir. Yıldırım S. ve Yıldız T. [49], tarafından Yıldız Teknik Üniversitesi Kemik Grubu [50] web sayfasından alınarak TTC4900 olarak isimlendirilmiştir. Türkçe haber metinlerini içerir.

Tablo 3.1 TTC4900 veri setinde bulunan kategoriler ve kategorilerdeki verilerin sayısı

Kategori	Sayı
siyaset	700
dunya	700
ekonomi	700
kultur	700
saglik	700
spor	700
teknoloji	700
Toplam	4900

TTC4900 veri setindeki teknoloji ve spor kategorisinden örnekler aşağıda görülmektedir.

```
In [13]: data['text'][4500]
Out[13]: ' galaxy s4 ortaya çıktı samsung un galaxy s3 ten sonra beklenen yeni akıllı
cebinin ilk fotoğrafı internete sızdı sammobile isimli sitede yayınlanan fotoğrafta yıl
sonuna doğru satışa sunulması beklenen galaxy s4 ilk kez karşımıza çıkıyor şubat ayı
sonunda barselona da gerçekleştirilecek dünya_mobil_kongresi nde kamuoyuna tanıtılacak
olan cihazın tasarımı galaxy s3 e benziyor android işletim sistemini kullanan cihazın
ekranı ise siii e göre daha büyük bir yapıda bulunuyor s4 ün donanımsal özellikleriyle
ilgili henüz bir bilgi yok galaxy s3 \x92 ün bilinmeyenleri'
```

```
In [20]: data['text'][3804]
Out[20]: ' elmander takımdan ayrı çalıştı spor_toto_süper_lig in 10 haftasında
istanbul_bb ile karşılaşacak olan galatasaray da karşılaşmanın hazırlıklarına devam
edildi galatasaray spor_toto_süper_lig in 10 haftasında 2 kasım_cuma günü
istanbul_büyükşehir_belediyespor ile yapacağı maçın hazırlıklarını sürdürdü
florya_metin_oktay_tesisleri nde basına kapalı gerçekleştirilen antrenmanda sarı
kırmızılıların el topu ve dar alanda çift kale maç oynayıp taktik çalışma yaptığı
bildirildi sakatlığı bulunan johan_elmander in tedavisinin ardından sahada takımdan ayrı
çalışma yaptığı kaydedildi galatasaray istanbul_büyükşehir_belediyespor maçının
hazırlıklarını yarın akşam yapacağı antrenmanla tamamlayarak tesislerinde kampa girecek'
```

Şekil 3.3 TTC4900 teknoloji ve spor kategorisinden örnek kayıtlar. Sırayla teknoloji ve spor.

3.1.2. Türkçe Sosyal Medya Verileri

Türkçe Sosyal Medya verilerinden oluşan veri seti [51], Kaggle web sitesinden alınmıştır. Pozitif ve Negatif şeklinde etiketlenmiş 11117 metin içerir. Sosyal medya üzerinde sıklıkla rastlanan siber zorbalık tespiti için hazırlanmıştır.

Tablo 3.2 Türkçe Sosyal Medya veri setindeki kategoriler ve sayıları

Kategori	Sayı
Pozitif	6113
Negatif	5004
Toplam	11117

Sosyal Medya Verileri veri setinden pozitif ve negatif kategorisinden örnekler aşağıda görülmektedir.

```
In [31]: data['Paylaşım'][101]
Out[31]: 'adamda tip desen yok fizik desen yok adamın neyini sevdin '
```

Şekil 3.4 Sosyal Medya Verileri veri seti negatif kategoriye örnek bir kayıt

```
In [41]: data['Paylaşım'][355]
Out[41]: 'survivor parkurundan sonra exatlon parkuru gözümüzü gönlümüzü açtı '
```

Şekil 3.5 Sosyal Medya Verileri veri seti pozitif kategoriye örnek bir kayıt

3.2. Metin Ön İşleme

3.2.1. Verilerin Hazırlanması

Metin verilerin hazırlanması için metinlerin eklerine köklerine ayrılmadan önce birkaç ön işlemden geçmesi gerekmektedir. Bunlar:

- Veriler, veri çerçevesi olarak yüklendikten sonra boş olan satırlar çıkarılır.
- Metindeki tüm harfler küçük harfe dönüştürülür. Böylece büyük harf ve küçük harf için ayrı tokenizasyon yapılması önlenir.

```
# Küçük harfe çevirme
data['text'] = data['text'].str.lower()
```

Şekil 3.6 Küçük harfe çevirme ile ilgili Python dilinde yazılan kod parçası

- Metinden noktalama işaretleri ve gereksiz kelimeler(stopwords) çıkarılır.

```
# Noktalama işaretlerinin, stopword'lerin ve sayıların çıkarılması
nltk.download('stopwords')
stopwords = set(nltk.corpus.stopwords.words('turkish'))
data['text'] = data['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in stopwords]
data['text'] = data['text'].apply(lambda x: re.sub('\d+', '', x))
data['text'] = data['text'].apply(lambda x: x.translate(str.maketrans('', '', string.punctuation)))
```

Şekil 3.7 Metinden noktalama işaretlerinin ve gereksiz kelimelerin çıkarılması ile ilgili kod

Doğal dil işleme araç setindeki (NLTK - Natural Language Toolkit) Türkçe dili için gereksiz kelimeler Tablo 3.3'teki gibidir:

Tablo 3.3 Doğal Dil İşleme Araç Setinde bulunan gereksiz kelimeler

acaba	ama	aslında	az	bazı	belki	biri
birkaç	birşey	biz	bu	da	daha	de
defa	diye	en	eğer	gibi	hem	hep
hepsi	her	hiç	ile	ise	için	kez
ki	kim	mu	mü	mı	nasıl	ne
neden	nerde	nerede	nereye	niye	niçin	o
sanki	siz	tüm	ve	veya	ya	yani
çok	çünkü	şey	şu			

- Metinden rakamsal karakterler çıkarılır. Genel olarak metin sınıflandırmada rakamsal karakterlerin etkisi yoktur.

- Kelimelerin kökleri bulunur. Bu tezde kelimelerin köklerinin bulunması için Zemberek [52] Türkçe morfoloji kütüphanesi kullanılmıştır.

Ön işlemden geçmiş olan ve bu tezde kullanılan veri setlerinden metin örnekleri Şekil 3.8 ve Şekil 3.9'da görülmektedir.

```
In [27]: data['text'][4500]
Out[27]: 'galaxy s orta çık samsung un galaxy s ten sonra bekle yeni akıl cep ilk
fotoğraf internet sız sammobile isim site yayınla fotoğraf yıl son doğru sat sun bekle
galaxy s ilk karşı çık şubat ay sonunda barselona gerçek UNK n kamuoyu tanı ol cihaz
tasarım galaxy s e benze android işletim sistem kullan cihaz ekran UNK e göre büyük bir
yapı bulun s ün donanım özellik ilgili henüz bir bilgi yok galaxy s UNK ün bil'
```

Şekil 3.8 TTC4900 veri setinde bir kaydın ön işlemden geçmiş hali

```
In [5]: data['Paylaşım'][101]
Out[5]: 'adam tip desen yok fizik desen yok adam ney sev'
```

Şekil 3.9 Sosyal Medya Verileri veri setinde bir kaydın ön işlemden geçmiş hali

3.2.2. Türkçe Kök Bulma

Türkçe gibi eklemeli dillerde, kelimelere yapım ve çekim ekleri gelebilir. Kelimeye gelen ekler kökle kaynaşarak farklı anlamları ortaya çıkarabilir. Bu durum, Türkçede kök bulma konusunu diğer dünya dillerine göre zorlaştırmaktadır.

Türkçe kök bulma konusunda Zemberek kütüphanesi, Zemberek kütüphanesinin Python modülü (Zeyrek) ve Snowball algoritması kullanılmaktadır. Türkçe kök bulma konusunda başka yöntemler de geliştirilmiştir. Bu yöntemlerden biri, Türkçe doğal dil işleme alanında bir başka çalışma ise Doğuç'un [53] 2020 yılında Python ile geliştirdiği Lemmatizer sistemidir. Lemmatizer sistemi Türkçe'de en sık kullanılan 130'dan fazla eki ve TDK sözlüğünü baz almaktadır. Kalbur [54] isimli Türkçe ek ve kök veri tabanı kullanılarak alınan geri beslemelerle doğruluk oranı sürekli artmıştır.

3.2.2.1. *Zemberek*

Zemberek, Türkçe doğal dil işleme için bir araçtır. Akın ve Akın [55] tarafından 2007 yılında geliştirilmiştir. Zemberek Java dilinde yazılmıştır. Türkçe için ilk doğal dil işleme çözümlerindendir. Türkçe metinlerin işlenmesi ve anlaşılması için bir dizi işlev sunar. Örneğin, kelime köklerini bulma, yazım denetimi, kelime türetme ve cümle ayrıştırma gibi işlevler yer alır.

Zemberek ayrıca, Türkçe metinlerin dilbilgisi kurallarına uygunluğunu kontrol etmek için de kullanılabilir. Dilbilgisi denetimi işlevi, metindeki cümleleri ayrıştırarak her bir cümlenin dilbilgisi kurallarına uygun olup olmadığını kontrol eder. Zemberek kütüphanesinin Python programlama diline kısmi olarak (sadece kök bulma amacıyla) aktarılması Zeyrek [56] isimli

bir Python modülü ile mümkün hale gelmiştir. Bu tez çalışmasında metinlerin köklerini bulma işlemi için Zemberek kütüphanesi kullanılmıştır.

```
# Zemberek Jar dosyasının yolu
zemberek_jar_path = r"C:/users/rozdemir/Downloads/Zemberek_kaggle/zemberek-full.jar"
# JVM başlatma işlemi
jpype.startJVM(jpype.getDefaultJVMPath(), "-ea", "-Djava.class.path=%s" % zemberek_jar_path)
TurkishMorphology = JClass('zemberek.morphology.TurkishMorphology')
morphology = TurkishMorphology.createWithDefaults()

for kelimeler in data['text']:
    analysis: java.util.ArrayList = (
        morphology.analyzeAndDisambiguate(kelimeler).bestAnalysis()
        )
    pos: List[str] = []
    for i, analysis in enumerate(analysis, start=1):

        f'\nAnalysis {i}: {analysis}',
        f'\nPrimary POS {i}: {analysis.getPos()}'
        f'\nPrimary POS (Short Form) {i}: {analysis.getPos().shortForm}'

        pos.append(
            f'{str(analysis.getLemmas()[0])}'
            )
    stemming_liste.append(" ".join(pos))
# JVM'i kapatın
jpype.shutdownJVM()
```

Şekil 3.10 Zemberek ile kök bulma işlemi. İşlem yapıldıktan sonra stemming_liste, data[‘text’]’e tekrar aktarılır ve liste seriye dönüştürülür.

3.2.2.2. *Snowball*

Snowball bir kök bulma algoritmasıdır. Bu algoritma, Porter [57] tarafından 2001 yılında tasarlanmış ve geliştirilmiştir. Porter’ın ilk geliştirdiği Porter algoritmasının daha gelişmiş hali olduğu için Porter2 olarak da bilinir.

Snowball algoritması, bir kelimenin son eklerini çıkarmak için koşullu olarak uygulanan bir dizi kural içerir. Python’da Snowballstemmer modülünün TurkishStemmer sınıfı bulunmaktadır.

Tablo 3.4 İngilizce dili için, Snowball algoritmasında yaygın kullanılan bazı kuralların gösterimi [58]

Kural	Kelime	Kök
ILY -----> ILI	easily	easili
LY -----> (boş)	fairly	fair
S -----> (boş)	sings	sing
ED -----> E,(boş)	cared	care

3.3. Kelime ve Doküman Temsilleri Oluşturma

Bu tez çalışmasında, Naive Bayes, ESA ve UKSB sınıflandırıcıları için metnin sayısal temsilinde Naive Bayes için sayma vektörleştirici (kelime çantası), ESA ve UKSB için Keras

Tokenizer ve GCN modelinde korpustan çizge oluşturmak için TF-IDF vektörleştirici ve PMI kullanılmıştır.

3.3.1. Sayma Vektörleştirici

Sayma vektörleştirici (CountVectorizer), metin verilerini sayısal özniteliklere dönüştürmek için kullanılan bir yöntemdir. Bu yöntem, bir dokümandaki her kelimenin frekansını sayar ve her kelimeye bir sütun atar. Kelimelerin bir öznitelik olarak temsil edildiği bir kelime çantası modeli oluşturur. Sayma vektörleştirici ile metin sayısal olarak ifade edilerek makine öğrenmesi modellerinde kullanılabilir.

3.3.2. TF-IDF Vektörleştirici

TF-IDF (Term Frequency-Inverse Document Frequency), metin verilerini sayısal olarak temsil etmek için kullanılan bir yöntemdir. Bu yöntemde; her kelimenin belgedeki önemini belirlemek için, kelimenin belgedeki sıklığı (Term Frequency - TF) ile tüm belgelerdeki nadirliği (Inverse Document Frequency - IDF) çarpılır. Böylece her kelime için bir ağırlık değeri elde edilir ve bu ağırlıklar bir vektör olarak kaydedilir. Bu vektörler, belgelerin TF-IDF temsilleridir. TF-IDF belgeler arası kelime ilişkilerine baktığı için belgeler arası bağlamı anlar ve buna göre vektörizasyon işlemini gerçekleştirir. TF-IDF yönteminde her kelimenin ağırlığı TF*IDF formülü ile hesaplanır. Burada TF değeri bir kelimenin belgedeki sıklığını, IDF değeri ise bir kelimenin tüm belgelerdeki nadirliğini gösterir.

TF değeri genellikle aşağıdaki formül ile hesaplanır:

$$TF(t, d) = \frac{f(t,d)}{max\{f(w,d): w\ in\ d\}} \tag{3.1}$$

Formül 3.1'de t kelimesi, d belgesi, f(t, d) t kelimesinin d belgesindeki sayısı ve max{f(w, d): w in d} d belgesindeki en yüksek (toplam) kelime sayısını gösterir.

IDF değeri ise genellikle aşağıdaki formül ile hesaplanır:

$$IDF(t, D) = log(\frac{N}{|\{d\ in\ D: t\ in\ d\}|}) \tag{3.2}$$

Formül 3.2'de t kelimesi, D belge kümesi (korpus), N toplam belge sayısı ve |{d in D: t in d}| t kelimesinin geçtiği belge sayısını gösterir.

TF-IDF değeri ise bu iki değerin çarpımıdır:

$$TF\text{-}IDF(t, d, D) = TF(t, d) * IDF(t, D) \quad (3.3)$$

Burada t kelimesi, d belgesi ve D belge kümesini (korpus) gösterir.

TF değeri, bir kelimenin belgedeki sıklığını gösterir. Genellikle bir kelimenin belgedeki sayısının belgedeki toplam kelime sayısına bölünmesi ile hesaplanır. Örnek olarak, "Bu elma çok güzel" cümlesinde "elma" kelimesinin TF değeri 1/4 = 0.25'tir. IDF değeri ise bir kelimenin tüm belgelerdeki nadirliğini gösterir. Genellikle toplam belge sayısının, bir kelimenin geçtiği belge sayısına bölünmesi ve sonucun logaritmasının alınması ile hesaplanır. Örnek olarak, "Bu elma çok güzel" ve "O elma da güzel" cümlelerini ele alalım. Bu iki cümle ayrı bir belge olarak kabul edilir. Elimizde 2 belge var ve iki cümlede de "elma" kelimesi geçiyor. Dolayısıyla "elma" kelimesinin IDF değeri log(2/2) = 0'dır. TF-IDF değeri ise bu iki değerin çarpımıdır: TF * IDF. Örnekteki; "Bu elma çok güzel" cümlesinde "elma" kelimesinin TF-IDF değeri 0.25*0 = 0'dır.

Tablo 3.5 TF-IDF ile metin verilerinin sayısal gösterimi. 1. Cümle="Bu elma çok güzel"; 2. Cümle="O elma da güzel" [59]

	TF			TF*IDF	
Kelime	1. Cümle	2. Cümle	IDF	1. Cümle	2. Cümle
Bu	1/4	0	log(2/1)=0.3	0.075	0
Elma	1/4	1/4	log(2/2)=0	0	0
Çok	1/4	0	log(2/1)=0.3	0.075	0
Güzel	1/4	1/4	log(2/2)=0	0	0
O	0	1/4	log(2/1)=0.3	0	0.075
da	0	1/4	log(2/1)=0.3	0	0.075

TF-IDF yöntemi ile bir kelimenin önemi hem kendi belgesindeki sıklığına hem de diğer belgelerdeki nadirliğine göre belirlenir. Böylece sıkça geçen ancak anlamsız kelimelerin ("ve", "ama", "bu" vb.) ağırlıkları düşük olurken, az geçen ancak anlamlı kelimelerin ağırlıkları yüksek olur.

3.4. PMI

PMI (Pointwise Mutual Information - Noktasal Karşılıklı Bilgi), iki kelimenin birlikte kullanılma olasılığının kelimelerin bağımsız olarak kullanılmaları olasılığına göre ne kadar yüksek olduğunu ölçmek için kullanılan bir ölçüttür. Doğal dil işleme alanında sıklıkla, kelimeler arasındaki ilişkiyi ölçmek için kullanılmaktadır. PMI, kelimeler arasındaki anlamsal

ilişkileri belirlemek ve kelime çiftlerini değerlendirmek için kullanılan bir ölçüttür. PMI, belirli bir kelimenin bir belge veya metinde görünme sıklığının, bir diğer kelimenin görünme sıklığından ne kadar farklı olduğunu ölçer.

Bu tez çalışmasında PMI ve TF-IDF metinden çizge oluşturmak için kullanılmıştır. PMI ile metin çizgesi oluşturulurken, çizgedeki iki düğüm arası (kelime-kelime) ve TF-IDF ile kelime-doküman arası ilişki aşağıdaki şekilde formüle edilebilir [1].

$$A_{xy} = \begin{cases} PMI(x,y) & x \; ve \; y \; kelime \; ve \; PMI(x,y) > 0 \; ise \\ TF - IDF_{xy} & x \; bir \; belge \; y \; bir \; kelime \; ise \\ 1 & x = y \; ise \\ 0 & diğer \; durumlarda \end{cases} \quad (3.4)$$

PMI değeri aşağıdaki şekilde hesaplanır:

$$PMI(x,y) = log \frac{p(x,y)}{p(x)p(y)} \quad (3.5)$$

Burada p(x, y) x ve y kelimelerinin birlikte bulunma olasılığını temsil eder. Aşağıdaki formülle hesaplanır:

$$p(x,y) = \frac{\#W(x,y)}{\#W} \quad (3.6)$$

Formüllerdeki p(x) ve p(y) ise x ve y kelimelerinin bağımsız olarak bulunma olasılıklarını temsil eder ve aşağıdaki formülle hesaplanır

$$p(x) = \frac{\#W(x)}{\#W} \quad (3.7)$$

Burada #W(x, y), x ve y kelimelerinin birlikte olduğu kelime sayısını, #W(x) x kelimesinin olduğu kelime sayısını ve #W toplam kelime sayısını temsil eder.

PMI hesaplamasına örnek olarak, “İstanbul” ve “Ankara” kelimelerini ele alalım. Bu iki kelimenin bir metin derlemesinde birlikte bulunma olasılığı, “İstanbul” ve “Ankara” kelimelerinin bağımsız olarak bulunma olasılıklarına göre yüksekse, bu kelime çiftinin PMI değeri pozitif olacaktır. Bu durumda, “İstanbul Ankara” kelime çiftinin bir arada bulunmasının anlamlı olduğu sonucuna varabiliriz. Bir metin derlemesinde “İstanbul” kelimesinin bulunma olasılığının 0.2, “Ankara” kelimesinin bulunma olasılığının 0.1 ve “İstanbul Ankara” kelime

çiftinin birlikte bulunma olasılığının 0.05 olduğunu varsayalım. Bu durumda "İstanbul Ankara" kelime çiftinin PMI değerini aşağıdaki gibi hesaplayabiliriz:

PMI("İstanbul", "Ankara") = log(0.05 / (0.2 * 0.1)) = 0.398

3.5. Sınıflandırma Algoritmaları

Bu tez çalışmasında; metin sınıflandırma modellerinin eğitiminde TTC4900 [49] ve Türkçe Sosyal Medya Verileri veri setleri kullanılmıştır. Sınıflandırma algoritmaları olarak Naive Bayes gibi geleneksel makine öğrenmesi algoritması, Evrişimli Sinir Ağları, Tekrarlayan Sinir Ağları ve Çizge Evrişimli Sinir Ağları gibi derin öğrenme algoritmaları kullanılmıştır. Bu algoritmalar ile oluşturulan modellerin test verileri karşısındaki F1 Skoru ve doğruluk gibi başarım metrikleri bu tez çalışmasındaki 4. Bölümde bulunan Bulgular kısmında değerlendirilmektedir.

3.5.1. Naive Bayes Sınıflandırıcı

Naive Bayes sınıflandırıcıları, Bayes Teoremi'ne dayanan bir sınıflandırma algoritması koleksiyonudur. Naive Bayes sınıflandırıcıları koleksiyonu, özellik değerlerinin dağılımlarına göre farklılık gösterir. Naive Bayes sınıflandırma algoritmalarının bazıları şunlardır:

Gaussian Naive Bayes: Sürekli değişkenlerle ve normal dağılımlarla kullanılır.

Multinomial Naive Bayes: Çok değişkenli dağılımlarla ve sayma verileri ile kullanılır.

Bernoulli Naive Bayes: İkili değişkenlerle ve çok değişkenli Bernoulli dağılımları ile kullanılır.

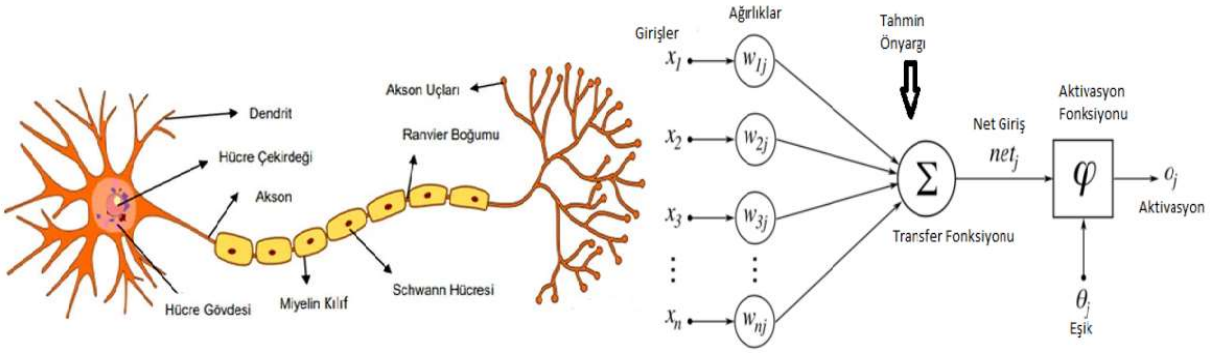
Bu sınıflandırıcıların her biri, farklı veri türlerinde en iyi sonuçları elde etmek için kullanılabilir. Bunların hepsi, sınıflandırılan her özellik çiftinin birbirinden bağımsız olduğu ortak bir ilkeyi paylaşır. Örneğin; bir meyve kırmızı, yuvarlak ve yaklaşık 10 cm çapında ise elma olarak kabul edilebilir. Naive Bayes sınıflandırıcısı, bu özelliklerin her birinin bu meyvenin bir elma olma olasılığına bağımsız olarak katkıda bulunduğunu düşünür [60]. Naive Bayes sınıflandırıcılarının temel varsayımı, her özelliğin sonuca bağımsız ve eşit katkıda bulunmasıdır.

Bu tez çalışmasında metin sınıflandırma performansının karşılaştırıldığı sınıflandırma algoritmalardan biri Multinomial Naive Bayes sınıflandırıcısıdır. Vektör temsili için kelime

çantası yöntemine dayanan sayma vektörleştirici kullanılmıştır. Sayma vektörleştirici gibi metinlerdeki kelime frekans sayımları gibi ayrık özniteliklerle yapılan sınıflandırmalarda Multinomial Naive Bayes yöntemi oldukça kullanışlıdır [61].

3.5.2. Yapay Sinir Ağları

Son yıllarda, yapay sinir ağları görüntü tanıma, ses tanıma ve makine çevirisi gibi alanlarda popülerlik kazanan bir yapay zekâ yöntemidir. Bu yöntem, insan beyninin temelini oluşturan sinir ağlarından ilham alınarak oluşturulmuştur. Yapay sinir ağları, insan beynindeki biyolojik nöronlara benzer bir şekilde çalışır ve programlamaya ihtiyaç duymadan deneyimlerden çıkarımlar yaparak öğrenir. Yapay sinir ağları, sinir ağı katmanlarından oluşur. Bu katmanlarda, algılayıcı (perceptron) olarak adlandırılan nöronlar bulunur. İnsan beynindeki nöronların birbirleriyle iletişim kurduğu gibi, yapay sinir ağlarındaki nöronlar da her katmanda birbirleriyle bağlantı kurar.



Şekil 3.11 Sinir hücresinin biyolojik ve matematiksel yapısı

Biyolojik nöronların çalışma şekli şu şekildedir: Nöronlar, dendritleri aracılığıyla diğer nöronların sinapslarından bilgileri alır. Alınan bilgilerin toplamı eşik değerini aşarsa, aksonlar aracılığıyla elektriksel sinyaller üreterek dendritler aracılığıyla diğer nöronlara bilgi iletilir. Yapay sinir ağlarında ise nöronlar işlem elemanlarına, dendrit toplama fonksiyonuna, hücre gövdesi aktivasyon fonksiyonuna, akson işlem elemanlarının çıkışına ve sinaps ağırlıklara karşılık gelir.

Yapay sinir ağları, bilgisayar sistemlerinin karmaşık problemleri çözmek, desenleri tanımak, tahminler yapmak veya kararlar vermek gibi görevleri gerçekleştirebilmesi için kullanılan bir makine öğrenimi yöntemidir. Veriye dayalı örüntüleri keşfedebilme ve bu örüntüleri kullanarak yeni veriler üzerinde tahminler yapabilme yeteneğine sahiptir. Bu nedenle, yapay sinir ağları geniş bir uygulama alanına sahiptir. Görüntü ve nesne tanıma, doğal dil işleme, öneri sistemleri

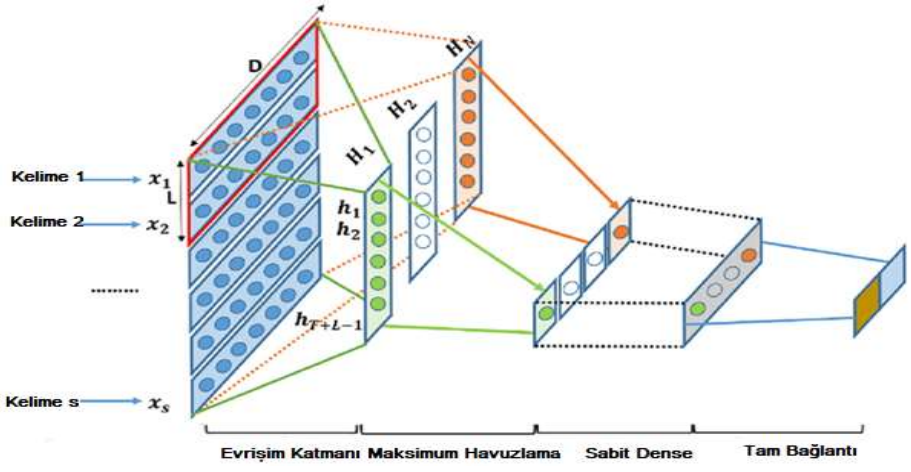
ve finansal analiz gibi alanlarda kullanılabilir. İleri beslemeli sinir ağları, evrişimli sinir ağları, tekrarlayan sinir ağları ve yinelemeli sinir ağları gibi türleri vardır. Derin öğrenme, yapay sinir ağlarının bir türüdür. Daha karmaşık problemleri çözmek için geliştirildiği için klasik sinir ağlarından farkı; girdi ve çıktı katmanı dâhil olmak üzere üç katmandan daha fazla katman içermesidir. Derin öğrenme, daha yüksek seviyede soyutlamalar ve temsiller yaparak verilerdeki karmaşıklığı anlama ve yüksek düzeyde performans gösterme potansiyeline sahiptir.

3.5.3. Evrişimli Sinir Ağları

Evrişimli Sinir Ağları, derin öğrenme alanında popüler bir yöntemdir. Genellikle görüntü sınıflandırma gibi görsel verilerle ilgili problemlerde kullanılır. Ancak, metin verileri için de kullanılabilir. Evrişimli sinir ağları, bir yapay sinir ağı çeşididir. Girdi ve çıktı katmanları ve aynı zamanda gizli katmanlara sahiptir. Evrişimli sinir ağlarında bulunan gizli katmanlar, evrişim adı verilen belirli matematiksel işlevleri kullanarak özetleme veya filtreleme gibi işlemleri gerçekleştirir. Evrişimli sinir ağlarında evrişim işleminin temel matematiksel formülü şu şekildedir:

$$C_{a,b} = \sum_m \sum_n I_{a+m,b+n} * F_{m,n} \tag{3.8}$$

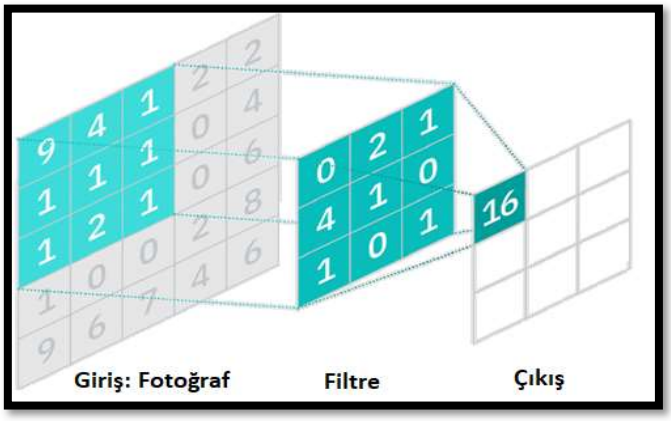
Formül 3.8'de, $C_{a,b}$ çıktı matrisinin a'ncı satır ve b'nci sütunundaki elemanı, $I_{a+m,b+n}$ giriş matrisinin a+m'nci satır ve b+n'nci sütunundaki elemanı ve $F_{m,n}$ filtre matrisinin m'nci satır ve n'nci sütunundaki elemanıdır. Evrişimli sinir ağlarının tipik katmanları Şekil 3.12'de gösterilmiştir.



Şekil 3.12 Metin sınıflandırma için standart ESA [62]

Evrişim işlemi, evrişim katmanında gerçekleştirilir. Bu katmanda birden fazla filtre kullanılabilir ve her filtre için ayrı bir evrişim işlemi gerçekleştirilir. Her bir evrişim işlemi sonucu elde edilen matrisler özellik (öznitelik) haritaları olarak adlandırılır.

Evrişim katmanı: Bu katmanda girdi verileri üzerinde evrişim işlemleri uygulanır. Evrişim işlemi, bir filtre matrisinin girdi verileri üzerinde gezdirilmesi ile gerçekleştirilir. Her bir filtre matrisi farklı bir yerel özelliği tespit etmeye yardımcı olur. Şekil 3.13'te görüldüğü üzere evrişim işlemi sırasında filtre, giriş matrisi ya da vektörü üstünde gezdirilerek üst üste gelen hücreler çarpılır. Sonunda çarpım sonuçları toplanır ve çıkış matrisine sırayla yazılır. Evrişimin bu şekilde uygulanması giriş matrisinin boyutunda bir indirgemeye sebep olur. Boyut olarak giriş matrisinin değerlerinin aynı kalması isteniyorsa matris kenarlarına dolgu (padding) işlemi yapılabilir.



Şekil 3.13 Evrişim işlemi. Giriş verisi resim olduğundan iki boyutludur. Bu nedenle matris şeklinde gösterilmiştir [63].

Evrişim işlemi sonucu oluşacak boyut aşağıdaki şekilde hesaplanır:

$$n_{out}\,\frac{n+2p-f}{s}+1 \tag{3.9}$$

p: dolgu miktarı; f: filtre boyutu; n: matrisin ilk boyutu (satır veya sütun); s: adım kaydırma

Aktivasyon katmanı: Bu katmanda evrişim katmanının çıktısı üzerinde bir aktivasyon fonksiyonu uygulanır. ESA'da aktivasyon fonksiyonu, sinir ağının doğrusal olmamasını sağlamak için kullanılır. Sinir ağlarının doğrusallığının bozulması, sinir ağının daha karmaşık ilişkileri öğrenmesine ve daha iyi bir performans göstermesine yardımcı olur. Aktivasyon fonksiyonları, sinir ağının her bir katmanındaki nöronların çıktılarını hesaplamak için kullanılır.

Sinir ağlarında aktivasyon fonksiyonu olarak genellikle Sigmoid, Tanh, ReLU ve Leaky ReLU fonksiyonları kullanılır.

Havuzlama (pooling) katmanı: Havuzlama katmanının amacı, giriş verisinin boyutunu azaltmak ve böylece hesaplama maliyetini düşürmektir. Havuzlama işlemi sayesinde modelin hesaplama yükü azaltılacağı gibi aşırı uyum (overfitting) problemine karşı dayanıklılık arttırılır ve modelin genelleştirme yeteneği arttırılır. Havuzlama işlemi sırasında, giriş verisi belirli bir boyuttaki pencere ile taranır ve her pencere için belirli bir işlem gerçekleştirilir. Örneğin, maksimum havuzlama işleminde her pencerenin maksimum değeri alınırken, ortalama havuzlama işleminde her pencerenin ortalaması alınır.

Tam Bağlantı Katmanı (Fully Connected Layer): Tam bağlantı katmanı, giriş verisinin boyutunu azaltarak daha yüksek seviyeli özelliklerin öğrenilmesine yardımcı olur. Genellikle ESA mimarisinin sonuna doğru bulunur. Tam bağlantı katmanı, her bir girişin tüm nöronlara bağlı olduğu bir katmandır. Bu katman, giriş verisini alarak sınıf skorları gibi hedefleri optimize etmek için kullanılır. Bu katman sayesinde modelin çıktısı, sınıflandırma problemleri için olasılık dağılımına dönüştürülebilir.

Bu tezde, ESA ile metin sınıflandırma yapmak için 1B(Conv1D) ESA modeli kullanılmıştır. Metin verilerinin Keras Vektorizer ile oluşturulan sayısal temsilleri modele girdi olarak verilmiş ve model eğitilmiştir.

3.5.3.1. *Dolgulama İşlemi*

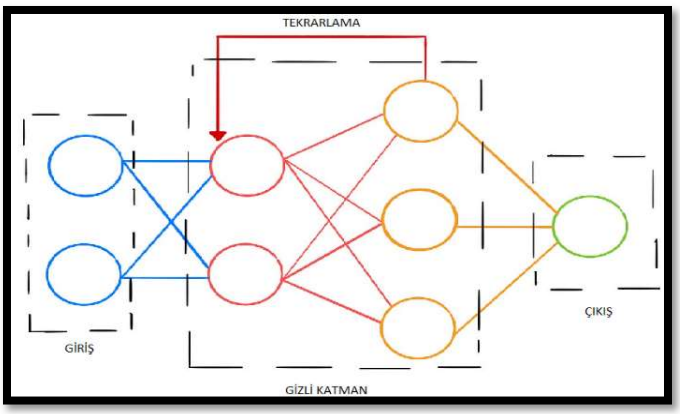
Dolgulama (Padding), genellikle doğal dil işleme veya zaman serisi analizi gibi alanlarda kullanılan bir ön işleme tekniğidir. Bu teknik, veri örneklerinin boyutlarını eşitlemek için kullanılır.

Şekil 3.14 Cümle veya kelime dizilerinde dolgulama işlemi [64]

Bir metin sınıflandırma problemi için bir veri setindeki cümlelerin uzunlukları farklı olabilir. Ancak, geleneksel sinir ağları modelleri için girdi boyutlarının sabit olması gerekir. Bu durumda, dolgulama tekniği kullanılarak tüm cümleler aynı uzunluğa getirilir. Kısa cümlelerin sonuna belirli bir karakter (genellikle 0) eklenerek bu işlem gerçekleştirilir. Tipik bir RNN farklı uzunluktaki dizilerle eğitilebilir. Ancak tekrarlayan sinir ağı mimarilerinde bir RNN katmanına girdi olarak verilen verinin bir şekli olmalıdır. Bunun için girdi olarak verilen tüm eğitim örneklerini sabit bir dizi uzunluğuna getirmek için dolgulama işlemi yapılır. Daha sonra model bir maskeleme katmanı ile başlatılarak eğitim verilerindeki dolgulu değerler gizlenebilir.

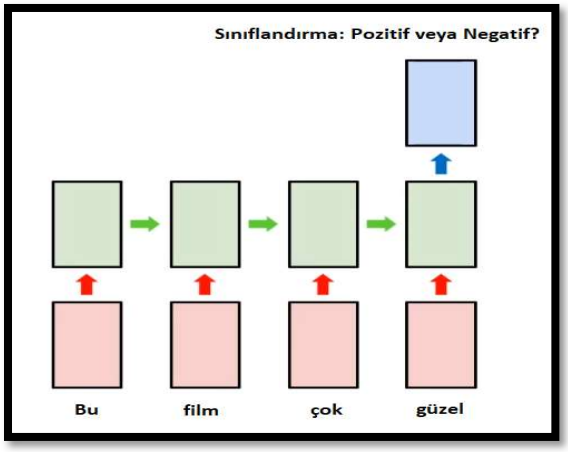
3.5.4. Tekrarlayan Sinir Ağları

Geleneksel sinir ağlarında giriş verilerinin birbirinden bağımsız olduğu varsayılır. Fakat bu varsayım, bazı problemler için geçerli değildir. Örneğin, Türkçeden İngilizceye çeviri uygulamalarında "Kalemle yazı yaz" ve "Bu yaz çok güzel olacak" metinlerini İngilizce diline çevirebilmek için yaz kelimesinin önündeki kelimeleri bilmek önemlidir. Bu nedenle bir cümledeki elemanlar bağımsız olarak ele alınamaz. Tekrarlayan sinir ağlarına (Recurrent Neural Networks-RNN) bir girdi dizisi olarak verilen tüm elemanlar çıktılarıyla birlikte bir sonraki katmanın girdisi için hesaplamalara tabi tutulur.



Şekil 3.15 RNN mimarisinde tekrarlayan gizli katmanlar [65]

Geleneksel sinir ağları; girdi olarak sabit uzunluklu vektör alarak, çıktı olarak yine sabit uzunluklu vektör üretir [66]. Tekrarlayan sinir ağlarında girdi ve çıktılar arasında bire-bir ilişki olmak zorunda değildir. Birden-çoğa, çoktan-bire ve çoktan-çoğa ilişkiler kurulabilir. Şekil 3.16'daki gibi çok sayıda kelime veya cümle vektörü modele girdi olarak verilerek paralel şekilde eğitilir ve bir çıktı üretebilir.



Şekil 3.16 RNN ile çoktan-bire sınıflandırma [67]

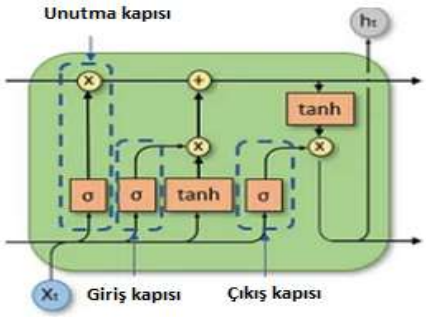
Tekrarlayan Sinir Ağları, zaman serisi verilerini işlemek için tasarlanmıştır ve metin verileri gibi sıralı verilerle ilgili problemlerde oldukça başarılıdır. RNN'ler, metin verilerini işlemek için kelime vektörlerinin üzerinde eğitilir. Bu vektörler, metin verilerini sayısal bir forma dönüştürür ve RNN tarafından işlenebilir hale getirir. RNN'ler, metin verilerini işlerken her bir kelimeyi sırayla işler ve her bir kelimenin işlenmesi sırasında önceki kelimelerden elde edilen bilgileri kullanır.

3.5.5. Uzun-Kısa Süreli Bellek

Uzun-kısa süreli bellek (UKSB) yapay sinir ağları, yapay zekâ ve derin öğrenme alanlarında kullanılan bir yapay sinir ağı türüdür. Standart ileri beslemeli (feedforward) sinir ağlarından farklı olarak, bir RNN türü olduğu için geri beslemeli bağlantılara sahiptir. Klasik sinir ağları gibi sadece tek veri noktasını (resimler gibi) değil, aynı zamanda veri dizilerinin tamamını (konuşma veya video gibi) işleyebilir. Bu özellik, UKSB ağlarını veri işleme ve tahmin etme için ideal hale getirir. UKSB; bağlantılı el yazısı tanıma, konuşma tanıma, makine çevirisi, konuşma aktivitesi tespiti, robot kontrolü, video oyunları ve sağlık hizmetleri gibi görevler için uygulanabilir.

UKSB, hem "uzun süreli hafızaya" hem de "kısa süreli hafızaya" sahiptir. Bu, UKSB'nin hem uzun vadeli bağımlılıkları hem de kısa vadeli bağımlılıkları öğrenme yeteneğine sahip olduğu anlamına gelir. Temel RNN'nin kısa süreli bellek problemini ortadan kaldırmak için tasarlanmıştır. UKSB uzun dizilerde ve zaman serilerinde temel RNN'leri etkileyen kaybolan gradyan probleminden kaçınır. Kaybolan gradyan problemi, RNN'lerin uzun süreli

bağımlılıkları öğrenmesini zorlaştıran bir problemdir. Bu problem nedeniyle, RNN'ler gecikmelerle gözlemlenen olayların etkilerini öğrenmekte zorlanabilirler.

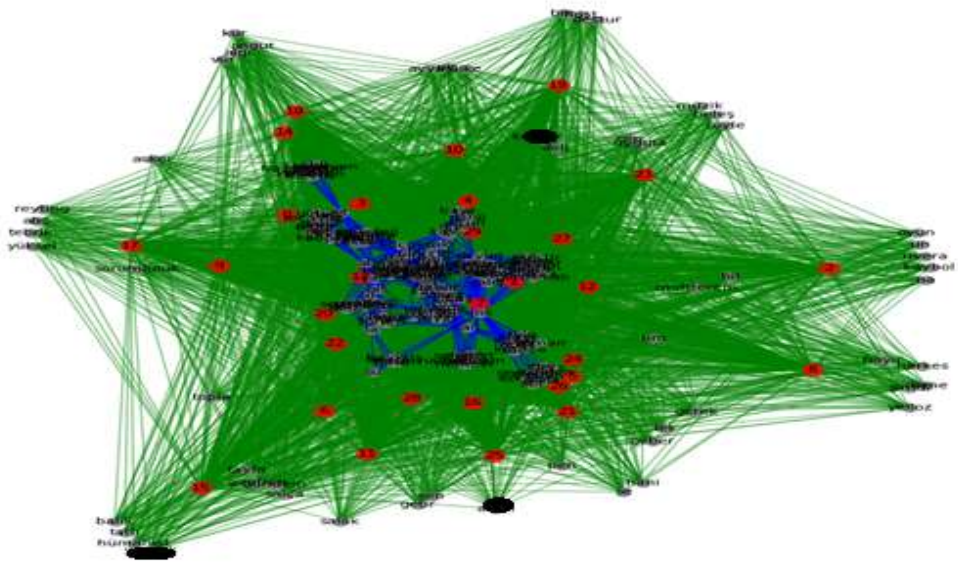


Şekil 3.17 UKSB hücresi [68]

UKSB ağı çok sayıda hücreden oluşur. Her bir hücre veriye ait farklı özellikleri öğrenir. Hücreler yapay sinir ağlarındaki nöronlara benzer şekilde çalışır. Her bir hücre girdi verisini işler, çıktı üretir ve kendi hücre durumunu günceller. Hücreler, ağın uzun süreli bilgiyi saklamasına ve taşımasına izin verir. Hücre durumu her bir zaman adımında güncellenir. Örneğin; metin verilerini işleyen bir UKSB ağında her bir kelimenin işlendiği adımda hücre durumu güncellenir. Bu nedenle ağ uzun süreli bağımlılıkları öğrenebilir. Hücre durumunu kontrol etmek için UKSB ağları, bir dizi "kapı" kullanır ve bu kapılar, veri dizisindeki bilgilerin ağa nasıl girdiğini, ağda nasıl saklandığını ve ağdan nasıl çıktığını kontrol eder. Tipik bir UKSB'de üç kapı vardır: unutma kapısı (forget gate), giriş kapısı (input gate) ve çıkış kapısı (output gate). Bu kapılar filtreler olarak düşünülebilir ve her biri sinir ağıdır. Unutma kapısı, önceki durumdan hangi bilgilerin hatırlanacağına veya önemsiz olduğu için unutulacağına karar verir. Giriş kapısı, hücrenin mevcut durumundan hangi yeni bilgilerin saklanacağına karar verir. Çıkış kapısı ise mevcut durumdaki bilgilerin hangilerinin çıkış yapacağını kontrol eder.

3.5.6. Çizge Evrişimli Sinir Ağları

Çizgeler, noktalar ile noktaları bağlayan kenarlardan oluşan bir yapıdır. Bir çizgede, genellikle noktalar durumları, kenarlar ise bu durumlar arasındaki ilişkileri gösterir. Gerçek hayattaki ya da bilişim dünyasındaki ilişkiler çizge olarak temsil edilebilir. Bilgisayar ağları, karayolları haritaları çizge olarak temsil edilebileceği gibi metinler de çizge olarak temsil edilebilir. Çizgede metindeki kelimeler düğümleri, kelimeler arasındaki ilişkiler kenarları temsil eder. Çizge şeklindeki bu temsil, metinlerin yapısını ve içeriğini daha iyi anlamaya yardımcı olabilir.



Şekil 3.18 Sosyal Medya Verileri veri setinden alınan bir örnek verilerin çizge yapısı. Çizgedeki yeşil kenarlar doküman ve kelimeler arasındaki kenarlar, mavi kenarlar ise kelime-kelime arası kenarlardır. Kırmızı noktalar ise dokümanların TF-IDF indisleridir. (Düğümlerdeki bazı kelimeler sansürlenmiştir.)

Çizge Evrişimli Sinir Ağları, çizgeler üzerinde işlem yapabilen bir derin öğrenme modelidir. Çizge yapıdaki veriler fotoğraf verisi gibi düzenli ve ızgara şeklinde olmadığından ESA gibi geleneksel derin öğrenme algoritmalarını çizgelere uygulamak zordur. Evrişim işlemini genişletmek ve çizge yapılara uygulamak için uzamsal yaklaşımlar ve spektral yaklaşımlar geliştirilmiştir.

Uzamsal (Spatial) yaklaşımlar: Uzamsal alandaki evrişim için, veriyi önce çizge düğümleri ve bunların ilişkileri şeklinde temsil etmek gerekir. Çizge oluşturulduktan sonra evrişim işlemi çizge düğümleri ve kenarları üzerinde geleneksel evrişim işlemine benzer bazı kurallara göre yapılır. Örneğin; GraphSAGE, Hamilton vd. [69] tarafından önerilen çizgelerin evrişimi için bir uzamsal yaklaşımdır. Düğüm sayısı sabit olmayan büyük ölçekli çizgelerde düğüm temsilleri oluşturmak için tasarlanmıştır. GraphSAGE, bir düğümün yerel komşuluğundan özellikleri örnekleyerek ve toplayarak düğüm yerleştirmelerini oluşturmak için bir toplama işlevi öğrenir. Uzamsal yaklaşımların en büyük zorluğu evrişim işlemini farklı uzaklıktaki komşular ile yapmaktır. Duvenaud vd. [70] Neural FP adlı yöntemle bu zorluğun üstesinde gelmek için farklı derecelerdeki düğümler için farklı komşuluk matrisi kullanmıştır. Böylece bu yöntemle modelin yerelliği korunurken farklı boyuttaki komşulukların işlenmesi sağlanır.

Spektral yaklaşımlar: Spektral yaklaşımlarda, öncelikle çizgeler üzerinde Fourier dönüşümü yapılarak önemli öznitelikler çıkarılır. Spektral yaklaşımda öncelikle graf sinyali x, Fourier dönüşümü F ile spektral domaine dönüştürülür. Daha sonra evrişim işlemi uygulanır. Evrişim işleminden sonra elde edilen sinyal ters Fourier dönüşümüne tabi tutulur [71].

$$F(x) = U^T x \tag{3.10}$$

$$F^{-1}(x) = Ux \tag{3.11}$$

Burada U, normalize edilmiş çizge Laplace özvektörler matrisidir. x çizge sinyali, T zaman değişkenidir.

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{3.12}$$

$$L = U \Lambda U^T \tag{3.13}$$

L özvektörler matrisi, D derece matrisi ve A grafik komşuluk matrisidir Λ(lambda) özdeğerlerin köşegen matrisidir.

Mallat'ın [72] evrişim teoremine göre evrişim işlemi:

$$g \star x = F^{-1}\big(F(g) \odot F(x)\big) = U(U^T g \odot U^T x) \tag{3.14}$$

($\mathrm{U}^{\mathrm{T}} g$ spektral domainin filtresidir.) Filtreyi köşegen matris $\mathrm{g_w}$ ile basitleştirirsek, spektral yöntemlerin temel fonksiyonu aşağıdaki gibi olur.

$$\mathrm{g_w} \star \mathrm{x} = \mathrm{U g_w U^T} x \tag{3.15}$$

3.5.6.1. *Çizge Evrişimli Sinir Ağları Literatür Taraması*

Literatür incelendiğinde, çizgelere evrişim işleminin uygulanabilmesi için birçok farklı yöntem denendiği görülmektedir. 2014'te LeCun vd. ESA'nın çizgelere uygulanabilmesi için iki ayrı yöntem denediler. Yöntemlerden biri çizge alanının hiyerarşik kümelenmesine, diğeri Laplace spektrumuna dayanır. Düşük boyutlu çizgeler ile yaptıkları deneyde, girdi boyutundan bağımsız bir miktar parametreyle evrişim katmanlarının öğrenilebileceğini gösterdiler [73].

Defferrard vd. [74] 2016 yılında yaptıkları çalışmada Chebyshev polinomlarını kullanarak spektral filtreler elde ederek filtre yerelleştirme sorununu çözdüler. Bu filtreler ile MNIST ve 20NEWS veri setlerinde deneyler yaptılar. Kipf ve Welling [44] çizgeler üzerinde evrişim

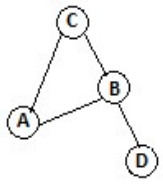
işlemini Weisfeiler-Lehman algoritması ile basitleştirerek Çizge Evrişimli Sinir Ağları fikrini ortaya atmışlardır [75]. Weisfeiler-Lehman algoritması, çizgelerin yapısını basitleştirmek için birleşik ve tekrarlayan bir işlem uygular. Algoritma, her bir düğümün etrafındaki komşuluk yapısını analiz ederek düğümlere etiketler atar. Bu işlem, her adımda düğümlerin etiketlerini günceller ve daha sonra etiketleri kullanarak düğümleri gruplar. Bu şekilde, çizgenin yapısı basitleştirilir ve graf (çizge) üzerindeki örüntüler daha kolay yakalanabilir hale gelir.

Yao vd. [1], [44]'deki çizge evrişimli sinir ağları modelini metin sınıflandırma problemine uyarlayarak metin tabanlı GCN (textGCN) ismini verdikleri bir yöntem geliştirmişlerdir. Bu yöntem bütün derlemin (korpus) tek bir çizge haline getirilmesine dayanır. Metin tabanlı GCN modelinde oluşturulan bu çizgeye iki katmanlı evrişim işlemi uygulanır. Bu yöntem ile metin sınıflandırma problemi düğüm sınıflandırma problemine indirgenmiştir.

3.5.6.2. *Çizge Evrişimli Sinir Ağları Temel Kavramlar*

Çizge evrişimli sinir ağlarının temel amacı düğümlerin özelliklerini ve komşuluk ilişkilerini kullanarak veri üzerinde işlemler gerçekleştirmektir. Yaklaşıma evrişimsel denilmesinin nedeni aynı ağırlıklar ile birinci dereceden komşuluk ilişkilerinin hesaplanmasıdır.

Komşuluk Matrisi (Adjacency Matrix), bir çizgenin düğümleri arasındaki ilişkileri temsil eden matristir. Çizgedeki bir düğümün diğer düğümlerle olan bağlantılarını gösterir. Eğer i ve j düğümleri arasında bir bağlantı varsa, komşuluk matrisinin i. satırı ve j. sütunu 1 değerine sahip olur. Eğer bağlantı yoksa 0 değeri kullanılır. Genellikle simetrik bir matris olarak kullanılır: A[i, j] = A[j, i]. Bu matris, GCN modelinin komşuluk bilgisini temsil etmek için kullanılır ve komşu düğümlerin bilgisini GCN katmanlarında kullanarak düğümlerin özelliklerinin güncellenmesinde rol oynar.



	A	B	C	D
A	0	1	1	0
B	1	0	1	1
C	1	1	0	0
D	0	1	0	0

Şekil 3.19 Bir çizge örneği ve çizgenin komşuluk matrisi

Derece (Degree) Matrisi, bir grafın (çizge) düğümlerinin derecelerini içeren bir kare matristir. Bir grafın derecesi, bir düğümünün kenarlarla olan bağlantı sayısıdır. Yani, bir düğümün kaç

tane komşusu olduğunu ifade eder. Derece matrisi ise, her düğümün derecesini teşkil eden bir matristir. Derece, her düğümün kaç tane komşusu olduğudur. Derece matrisi genellikle diyagonal matristir. Matrisin köşegenindeki elemanlar, grafın düğümlerinin derecelerini ifade eder. Diğer elemanlar ise genellikle sıfırdır. Matrisin boyutu, grafın düğüm sayısına eşittir.

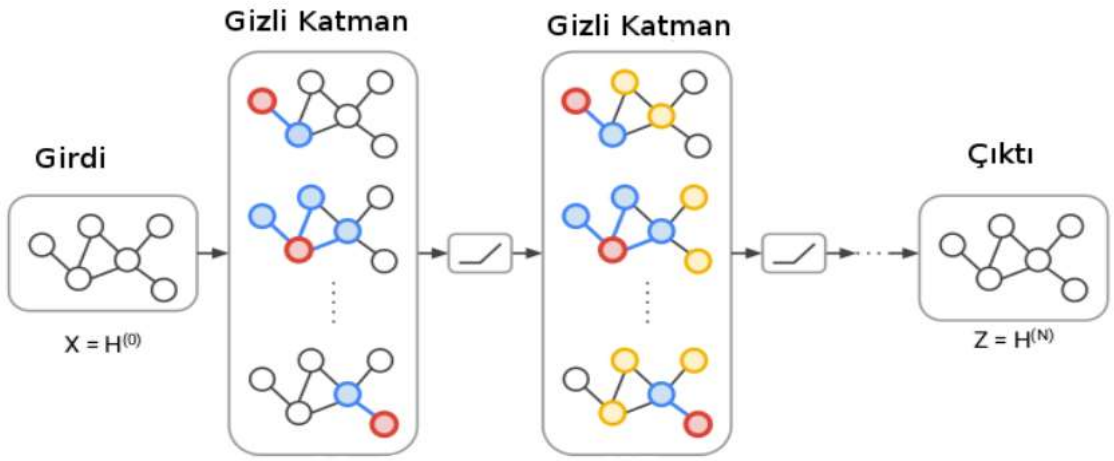
Çizge evrişimli sinir ağlarında, komşuluk matrisinin normalleştirilmiş hali modelde girdi olarak kullanılır. Komşuluk matrisi olan A matrisi, D derece matrisi kullanılarak aşağıdaki formülle normalleştirilir.

$$A_{hat} = D^{\frac{-1}{2}} * A * D^{-\frac{1}{2}} \tag{3.16}$$

A_{hat} matrisi, A matrisinin normalleştirilerek, düğümler arasındaki ilişkileri ve düğümlerin derecelerini dikkate alan bir formu olarak kullanılır. GCN gibi yöntemlerde A_{hat} matrisi, düğümlerin komşuluk bilgisini temsil eden bir özellik matrisine çarpan olarak kullanılarak graf üzerindeki işlemler gerçekleştirilir. Bu normalleştirme işlemi, GCN'nin daha istikrarlı ve etkili bir şekilde çalışmasına yardımcı olur

Özellik (Öznitelik) Matrisi, çizge düğümlerine ilişkin özelliklerin saklandığı matristir. Düğüm sayısıyla aynı boyuttadır. Çizge düğümlerinin vektör temsilidir. Genellikle 1 ve 0'lardan oluşan temsille gösterilir.

Bir GCN modelinde katmanlar, bir düğümün kaç komşuluk seviyesi geriye gideceğini gösterir. İkinci katmanda işlem yapmak için düğümün komşusunun komşusuna gidilir.



Şekil 3.20 GCN mimarisinde katmanlar. İlk seviye komşuluklar mavi renkte, ikinci seviye komşuluklar sarı renk ile gösterilmiştir [76].

Bu sinir ağlarına evrişimli denilmesinin nedeni ESA'daki gibi tüm düğümlerde aynı ağırlıkların kullanılmasıdır. Çizge evrişimli sinir ağlarında ileri yayılım sonucu elde edilen matris aşağıdaki şekilde hesaplanır.

$$X' = \sigma(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}XW) \tag{3.17}$$

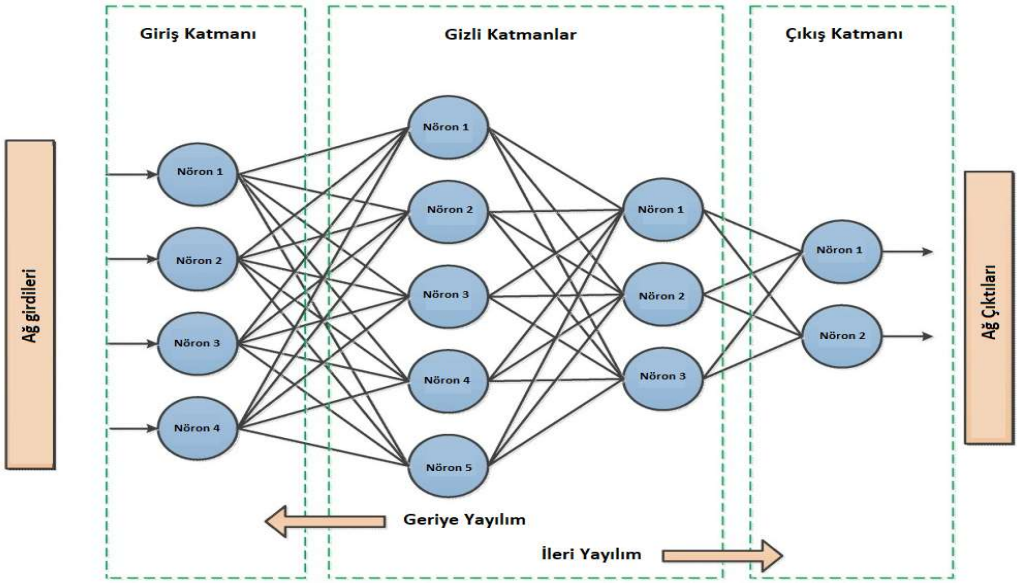
Normalleştirilmiş komşuluk matrisi, katman girdisi ve ağırlıklar ile çarpılarak X' elde edilir. X sinir ağının girdisini, W sinir ağındaki ağırlıkları, D derece matrisi, A komşuluk matrisi ve σ aktivasyon fonksiyonunu ifade etmektedir.

Çok katmanlı çizge evrişimli sinir ağında X, ilk katmanda düğümlerin öznitelik vektörlerini temsil eder. Sonraki katmanlarda X önceki katmanın çıktısıdır. Son katmanın çıktısı ise modelin çıktısıdır.

3.6. Sınıflandırma Modellerinin Eğitimi

Klasik makine öğrenmesi algoritmalarından Naive Bayes ile eğitim genellikle tek adımda gerçekleşir. Bu nedenle Naive Bayes algoritması için epoch veya iterasyon gibi kavramlar kullanılmaz. Naive Bayes sınıflandırma algoritması, önceden hesaplanmış olasılıklara dayanarak veri noktalarını sınıflandırır. Eğitim sürecinde, modelin parametreleri hesaplanır ve bu parametreler sonraki tahminlerde kullanılır. Her bir özelliğin sınıf üzerindeki etkisini hesaplamak için kullanılan Bayes teoremini kullanır. Eğitim setindeki örneklerin özellikleri ve sınıf etiketleri üzerinde hesaplamalar yapılır ve bu hesaplamalar sonucunda modelin parametreleri elde edilir. Bu parametreler tahmin için kullanılır.

Klasik makine öğrenmesinin aksine, yapay sinir ağları, verilerin karmaşık ilişkilerini öğrenmek için ağırlıkları ve bias değerlerini iteratif olarak güncelleyerek eğitilir. Eğitim sürecinde, yapay sinir ağları veri örneklerini besler, ileri yayılım (forward propagation) adı verilen bir işlemle çıktıları tahmin eder ve ardından geriye yayılım (backpropagation) adı verilen bir işlemle hata değerlerini geriye doğru ileterek ağırlıkları günceller. Bu süreç, belirli bir sayıda epoch veya iterasyon boyunca tekrarlanır.



Şekil 3.21 Sinir ağlarında ileriye ve geriye yayılım [77]

İleriye yayılım, verinin ağın girişinden başlayıp katmanlardan geçerek çıktıya doğru ilerlemesi işlemidir. İleriye yayılım, veri girişi üzerinden ağırlıkların ve biasların işlenmesini ve aktivasyon fonksiyonları tarafından hesaplanarak her katmandan geçişini içerir. Bu adımda, giriş verisi her katmana iletilir ve son katmanda çıktı elde edilir. İleriye yayılım süreci, ağın çıktısını hesaplarken ilgili verilerin aktarılmasını sağlar.

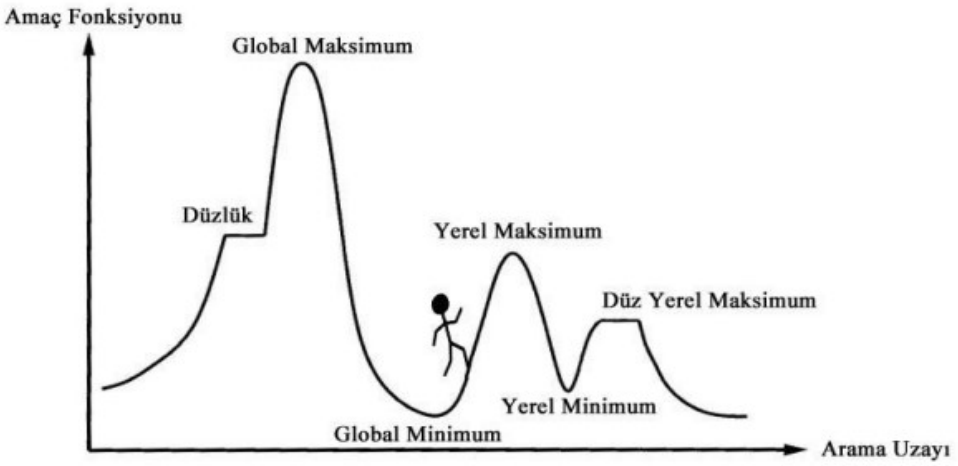
Geriye yayılım, ağın çıktısının hedef çıktıyla karşılaştırılması ve bu hata üzerinden ağın geriye doğru katmanlara doğru hata miktarını geriye yaymasıdır. Geriye yayılım süreci, hata kaybını azaltmak ve ağırlıkları güncellemek için kullanılır. Hatanın geriye yayılması, bayır inişi algoritması kullanılarak yapılır. Bu adımda, hata gradyanı, çıktı katmanından başlayarak geriye doğru katmanlara iletilir ve her bir ağırlığın ve biasın güncellenmesi için kullanılır.

3.6.1. Optimizasyon Algoritmaları

Optimizasyon algoritmaları, geriye yayılımın hesapladığı hatalardan yararlanarak ağın ağırlıklarını güncellemek ve performansı iyileştirmek için tasarlanmıştır. Bu algoritmalar, hata fonksiyonunu minimize etmek için ağırlık güncellemelerini yönlendirir. En yaygın kullanılan optimizasyon algoritmalarından biri bayır inişidir (gradient descent). Bayır inişi, ağın ağırlıklarını hata fonksiyonunun negatif gradyanına (türevine) göre günceller. Bu, ağırlıkları hatanın azaldığı yönde güncelleyerek optimum çözüme doğru ilerlemeyi sağlar. Diğer optimizasyon algoritmaları, bayır inişinin geliştirilmiş versiyonlarıdır. Örneğin, momentum, RMSProp, Adam gibi optimizasyon algoritmaları, bayır inişini daha hızlı ve etkili bir şekilde

gerçekleştirmek için farklı teknikler kullanır. Bu algoritmalar, gradyanın yönlendirildiği hız ve yönü kontrol ederek ağırlık güncellemelerini yapar.

Yapay sinir ağlarında optimizasyon işlemi, bir hata fonksiyonunun (loss function) belirli kısıtlara ve parametrelere bağlı olarak en düşük değere ulaşmasını sağlamayı amaçlar. Bu işlem, ağın eğitimi sürecinde gerçekleştirilir ve ağın tahminlerinin doğruluğunu artırmayı hedefler.



Şekil 3.22 Optimizasyon problemlerindeki temel kavramlar [78]

Optimizasyon işleminin temel amacı, kayıp veya maliyet fonksiyonu için bulunan minimum değerin yerel minimum yerine global minimuma ulaşmasını sağlamaktır. Eğitim kümesinde bulunan minimum değerler her zaman en iyi sonucu garanti etmeyebilir; optimizasyon işlemi yerel minimuma düşmüş de olabilir. Bu nedenle, hangi modelde hangi optimizasyon algoritmasının kullanıldığı büyük önem taşır. Yapay sinir ağlarında kullanılan çeşitli optimizasyon algoritmaları mevcuttur. Bayır inişi, stokastik bayır inişi, RMSprop, Adam, Adagrad ve Adadelta bunlardan bazılarıdır. Bu tez çalışmasında, adam optimizasyon algoritması kullanılmıştır.

3.6.1.1. *Adam Optimizasyon Algoritması*

Adam optimizasyonu, derin öğrenme modellerini eğitmek için stokastik bayır inişinin yerine kullanılan bir optimizasyon algoritmasıdır. Adam, AdaGrad ve RMSProp algoritmalarının en iyi özelliklerini birleştirerek gürültülü problemlerde seyrek gradyanları işleyebilen bir optimizasyon algoritması sağlar.

Adam algoritması, 2015 yılında Diederik Kingma ve Jimmy Ba [79] tarafından "Adam: A Method for Stochastic Optimization" başlıklı makalede tanıtılmıştır. Algoritmanın adı

“Adaptive Moment Estimation” (Uyarlanabilir Moment Tahmini) kelimelerinin kısaltmasından gelmektedir. Adam algoritması, stokastik hedef fonksiyonlarının birinci mertebeden gradyan tabanlı optimizasyonu için kullanılır. Algoritma, düşük mertebe momentlerinin uyarlanabilir tahminlerine dayanır ve kolayca uygulanabilir, hesaplama açısından verimli, az bellek gerektirir ve gradyanların diyagonal ölçeklendirmesine karşı değişmezdir. Ayrıca, veri ve parametreler açısından büyük problemler için uygundur ve çok gürültülü ve seyrek gradyanlarla çalışabilen problemler için de uygundur.

3.6.2. Aktivasyon Fonksiyonları

Bir aktivasyon fonksiyonu, giriş verileri ağırlıklarla çarpıldıktan ve önyargı(bias) kısmı eklendikten sonra tek bir değere dönüştürülen çok sayıda nöronun toplam çıktısını almaktan sorumludur. Toplam çıktının doğrusal (lineer) olmamasını sağlar. Doğrusal bir çıktı oluşturmayan aktivasyon fonksiyonun burada uygulanmasının nedeni, bütün gerçek hayat problemlerinin doğrusal bir çözüme sahip olmamasıdır. Bu nedenle, model grafiğini şekillendirmek ve eğitim veri noktalarına uymak için aktivasyon fonksiyonlarının kullanılması gereklidir. Aktivasyon fonksiyonları kullanılmasaydı, eğitim sırasında geriye yayılım kullanılarak modelin veriye daha uygun hale getirilmesi sağlanamazdı. Son zamanlarda araştırma çalışmalarında kullanılan bazı popüler aktivasyon fonksiyonları; sigmoid, RELU, hiperbolik tanjant ve sınıflamada olasılıksal çözüm üreten Softmax’tır. Bu aktivasyon fonksiyonlarının arasından bu çalışmada kullanılanlar açıklanacaktır.

3.6.2.1. *RELU Aktivasyon Fonksiyonu*

RELU (Rectified Linear Unit - Doğrultulmuş Doğrusal Birim), girdi değerlerinde pozitifliği arayan bir aktivasyon fonksiyonudur. Girdi değeri pozitif ise değeri olduğu gibi döndürür; aksi takdirde, değer sıfırdan küçükse, çıktı değerini sıfır (0) olarak döndürür.

3.6.2.2. *Softmax*

Softmax fonksiyonu, çıktı nöronları üzerinden normalleştirilmiş bir olasılık dağılımı döndürür. Çoğunlukla çok sınıflı sınıflandırmada kullanılır; çünkü her sınıf için maksimum değerin nihai sınıf olarak döndürüldüğü olasılık değerleri sağlar. Softmax, modelimizin son katmanları için bir aktivasyon fonksiyonu olarak kullanılmıştır. Örneğin; bir modelin son katmanında softmax aktivasyon fonksiyonu olarak kullanılıyorsa, kedi ve köpek fotoğrafları arasındaki sınıflandırmada bir kedi fotoğrafı için %80 kedi, %20 köpek şeklinde bir sonuç beklenir.

3.6.3. Kayıp Fonksiyonları

Kayıp fonksiyonları, makine öğrenme modelinin performansını ölçmek için kullanılır. Bir makine öğrenme modeli, verilen bir girdiye karşılık bir tahmin üretir. Bu tahminin doğruluğunu ölçmek için kayıp fonksiyonları kullanılır.

Kayıp fonksiyonları, modelin tahminini gerçek değerle karşılaştırarak bir hata değeri üretir. Bu hata değeri ne kadar düşükse, modelin performansı o kadar iyi demektir. Modelin eğitim sürecinde, kayıp fonksiyonunun değerini en aza indirmeye çalışırız. Böylece modelin tahminleri gerçek değerlere daha yakın hale gelir.

Kayıp fonksiyonları, modelin performansını ölçmek ve modelin eğitim sürecini yönlendirmek için kullanılır.

3.6.3.1. *Kategorik Çapraz Entropi*

Kategorik Çapraz Entropi (Categorical Cross-Entropy), çok sınıflı sınıflandırma problemlerinde kullanılan bir kayıp fonksiyonudur. Bu fonksiyon; modelin tahmin ettiği olasılık dağılımını, gerçek olasılık dağılımı ile karşılaştırarak bir hata değeri üretir. Formülü şu şekildedir:

$$H(p,q) = -\sum_i p_i \, log(q_i) \tag{3.18}$$

Burada p gerçek sınıf etiketlerini temsil eden bir vektördür ve q modelin tahmin ettiği olasılık dağılımını temsil eden bir vektördür.

Kategorik çapraz entropi, gerçek sınıf etiketleri ile modelin tahmin ettiği olasılık dağılımı arasındaki farkı ölçer. Modelin tahminleri gerçek sınıf etiketlerine ne kadar yakınsa, kategorik çapraz entropi değeri o kadar düşük olur.

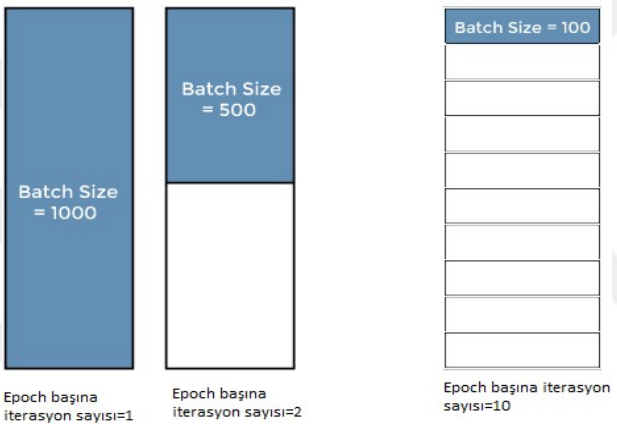
Örnek olarak, bir modelin 3 sınıfı ayırt etmesi gerektiğini düşünelim. Modelin tahmin ettiği olasılık dağılımı [0.1, 0.8, 0.1] olsun. Bu dağılım, modelin ilk sınıfın olasılığını %10, ikinci sınıfın olasılığını %80 ve üçüncü sınıfın olasılığını %10 olarak tahmin ettiğini gösterir. Gerçek olasılık dağılımı ise [0, 1, 0] olsun. Bu dağılım, verinin gerçek sınıfının ikinci sınıf olduğunu gösterir. Bu durumda Kategorik Çapraz Entropi kaybı aşağıdaki gibi hesaplanır:

loss = -log(0.8) = 0.097

Bu kayıp değeri ne kadar düşükse, modelin tahminleri gerçek değerlere o kadar yakın demektir.

3.6.4. Eğitim Tur Süresi, İterasyon ve Parti Boyutu

Eğitim tur süresi (Epoch), bütün veri setinin ağ boyunca ileri ve geri hareket etmesi anlamına gelir. Geri harekete geriye yayılım da denilebilir. Eğitim turları ile model ağırlıkları güncellenerek modelin veriye en uygun şekle gelmesi sağlanır. İterasyon, belirli bir parti boyutundaki verilerin ağ üzerinden ileri geri aktarımıdır. Bir eğitim tur süresi belirli parti boyutundaki verilere yapılan iterasyonların veri setindeki eğitim verilerine uygulanması ile tamamlanır. Parti boyutu (Batch size) ise eğitim için bir iterasyonda modele verilecek veri adedidir. Parti boyutu ne kadar büyük olursa, o kadar fazla bellek alanı gerekir. Büyük bir parti seçilirse, her iterasyon için eğitim örneği de büyük olacaktır. Örneğin; 2000 verilik bir veri seti için parti boyutu 1000 seçilirse bir epoch için iki adet iterasyon gerekir.



Şekil 3.23 Parti boyutu(Batch Size), epoch ve iterasyon [80]

Model eğitilirken daha yüksek bir epoch sayısı seçmek, modelin daha doğru sonuçlar vereceği anlamına gelmez. Aksine modelin ezberlemesine (overfitting) neden olabilir. Modelin eğitimi için epoch sayısı arttırılırsa ve diğer hiperparametre değerlerinden bağımsız olarak, bir noktadan sonra test doğruluğunun daha düşük olduğu gözlemlenebilir.

3.7. Sınıflandırma Modelinin Değerlendirilmesi

3.7.1. Hata Matrisi

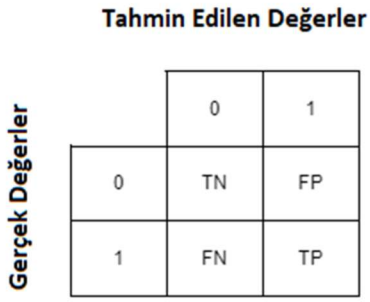
Derin öğrenme ve makine öğrenmesinde tasarlanan sınıflandırma modellerinin performansını değerlendirmek için hedef niteliğe ait tahmin edilen değerler ile gerçek değerlerin karşılaştırıldığı hata matrisi (confusion matrix) sıklıkla kullanılmaktadır. Hata matrisinin elemanları aşağıdaki şekildedir:

Gerçek Pozitif (True Positive - TP): Sınıfı doğru olan etiketi doğru şekilde tanımlamak

Gerçek Negatif (True Negative - TN): Sınıfı yanlış olan etiketi yanlış şekilde tanımlamak

Yanlış Pozitif (False Positive - FP): Sınıfı yanlış olan etiketi doğru şekilde tanımlamak

Yanlış Negatif (False Negative - FN): Sınıfı doğru olan etiketi yanlış şekilde tanımlamak



Şekil 3.24 İki sınıflı sınıflandırma problemine ilişkin hata matrisi

Sınıflandırma sonuçlarını doğru bir şekilde değerlendirmek ve farklı parametre değerleri ile karşılaştırmak için hata matrisi kullanılarak bazı ölçütlerin hesaplanması gerekmektedir. Hata matrisi kullanılarak hesaplanan bazı metrikler aşağıdaki gibidir:

Doğruluk (Accuracy): Doğru ya da yanlış olarak (TP+TN) kaç tane değeri doğru olarak tahmin ettiğimizin ölçüsüdür. Gerçek negatiflerin ve gerçek pozitiflerin daha önemli olduğu veya sınıf dağılımının dengeli olduğu durumlarda kullanılır. Örneğin, bir resimdeki nesneyi tanımak veya bir metnin konusunu belirlemek gibi problemlerde doğruluk tercih edilebilir. Doğruluk aşağıdaki şekilde hesaplanır:

$$Doğruluk = \frac{TP+TN}{TP+FP+TN+F} \tag{3.19}$$

Kesinlik (Precision): Pozitif olarak tahmin edilen değerlerin kaç tanesinin gerçekte pozitif olduğunun ölçüsüdür. Kesinlik aşağıdaki şekilde hesaplanır:

$$Kesinlik = \frac{TP}{TP+FP} \tag{3.20}$$

Duyarlılık (Recall): Veri kümesindeki tüm pozitif değerlerden kaç tanesinin doğru tahmin edildiğinin ölçüsüdür. Aşağıdaki şekilde hesaplanır:

$$Duyarlılık = \frac{TP}{TP+FN} \tag{3.21}$$

F1 Skoru: Kesinlik ve duyarlılık değerlerinin harmonik ortalamasıdır. Bu iki metriğin dengeli bir şekilde değerlendirilmesini sağlar. F1 skoru, sınıf dağılımının dengesiz olduğu durumlarda daha iyi sonuç verirken, doğruluk dengeli sınıf dağılımına ihtiyaç duymaktadır. Yanlış negatiflerin ve yanlış pozitiflerin çok önemli olduğu veya sınıf dağılımının dengesiz olduğu durumlarda kullanılır. Örneğin, bir hastalığı teşhis etmek veya dolandırıcılık tespiti yapmak gibi problemlerde F1 skoru tercih edilebilir. Aşağıdaki şekilde hesaplanır:

$$F1 = 2 * \frac{Kesinlik * Doğruluk}{Kesinlik + Doğruluk} \tag{3.22}$$

4. BULGULAR

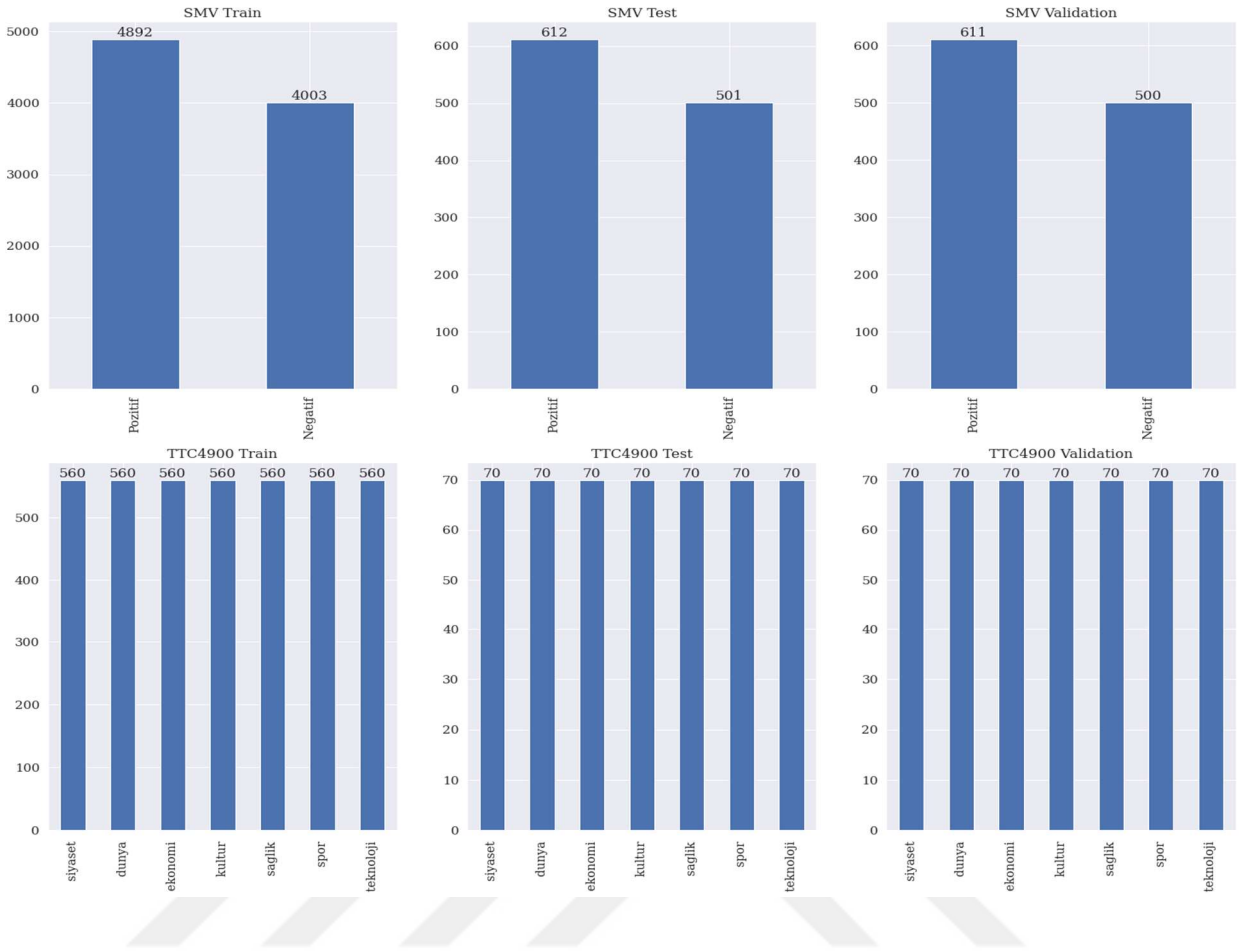
Çalışma sonucunda elde edilen bulguların değerlendirilmesine bu bölümde yer verilmiştir. Modelleri eğitirken ve test ederken kullanılan bilgisayarın özellikleri Tablo 4.1'de verilmiştir.

Tablo 4.1 Bilgisayarın özellikleri

Donanım/Yazılım	Detaylar
CPU	11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz
RAM	Samsung 16 GB DDR 3.2GHZ
GPU	Intel(R) Iris(R) Xe Graphics
SSD	Samsung PM991a NVMe 512GB
OS	Windows 10 Enterprise 64-bit (10.0, Build 19045)

Bu tez çalışmasında Spyder programının 5.3.3 ve TensorFlow kütüphanesinin 2.8.2 versiyonu kullanılmıştır. 4 farklı algoritma, 2 adet Türkçe veri seti ile eğitilmiş ve 8 adet model elde edilmiştir. Birinci model için için klasik makine öğrenmesi algoritmalarından Naive Bayes kullanılmıştır. İkinci model derin öğrenme algoritmalarından sadece bir katmanında ESA olan 6 katmanlı ve üçüncü model ise bir katmanı UKSB olan 5 katmanlı sinir ağları ile tasarlanmıştır. Son model ise 2 adet çizge evrişim katmanı olan GCN ile tasarlanmıştır. Metrik olarak modellerin test doğruluğu, f1 Skoru, hata matrisi ve modelin eğitimi boyunca elde edilen kayıp epoch ve doğruluk epoch grafikleri ile modeller karşılaştırılmıştır.

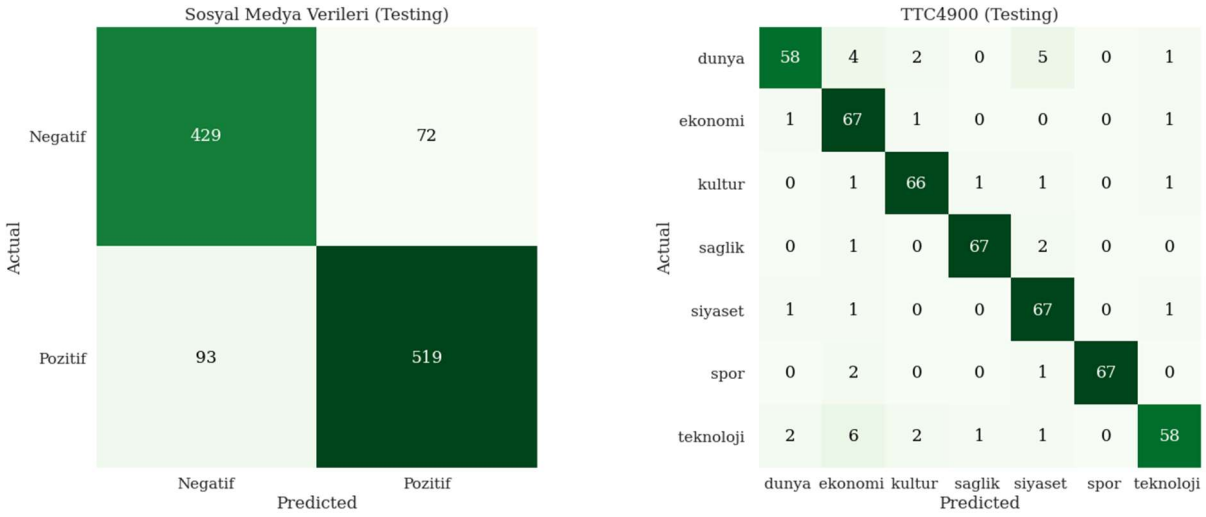
TTC4900 ve Sosyal Medya Verileri veri setlerinin %10'u eğitim %10'u doğrulama ve %80'i eğitim için ayrılmıştır. Eğitim test ve doğrulama için her sınıftan eşit sayıda veri olması sağlanmıştır. Ancak Sosyal Medya Verileri veri setinde kategorilerde eşit sayıda veri bulunmamaktadır. Bu nedenle kategorilerdeki verilerin yaklaşık olarak eşit olması sağlanmıştır. Eğitim, doğrulama ve test veri kümelerindeki kategorilerin sayısına ilişkin grafikler Şekil 4.1'de verilmiştir. Ayrıca bazı modellerde bütün veri setindeki eğitim verileri yüzdesel olarak artırılıp azaltılarak modellerin doğrulama doğruluklarındaki değişim gözlemlenmiştir. UKSB algoritması ile oluşturulan modelin TTC4900 modeli ile eğitiminin uzun sürmesi üzerine her epoch başına geçen zamana ilişkin grafik çizdirilmiştir.



Şekil 4.1 Veri setlerinde eğitim test ve doğrulama için ayrılan verilerin sınıflara göre dağılımları. SMV, Sosyal Medya Verileri veri setidir.

4.1. Naive Bayes Sınıflandırıcı ile Oluşturulan Modellere İlişkin Bulgular

Multinominal Naive Bayes algoritması ile oluşturulan modelin TTC4900 ve Sosyal Medya Verileri veri setinin eğitim için ayrılan %80'i ile eğitimine ilişkin test verilerinin heatmap şeklinde hata matrisi Şekil 4.2'deki gibidir. SMV (Sosyal Medya Verileri) ile eğitilen birinci modelde eğitim için kullanılan pozitif kategorileri içeren veri setinin daha yüksek olması nedeniyle pozitif tahmin ettikleri değerler daha fazladır. Ancak model yüzdesel olarak negatif kategori için daha iyi bir tahminde bulunmuştur. TTC4900 ile eğitilen ikinci modelde, model dünya ve teknoloji kategorisinde diğer kategorilere göre daha az miktarda doğru tahminde bulunmuştur.



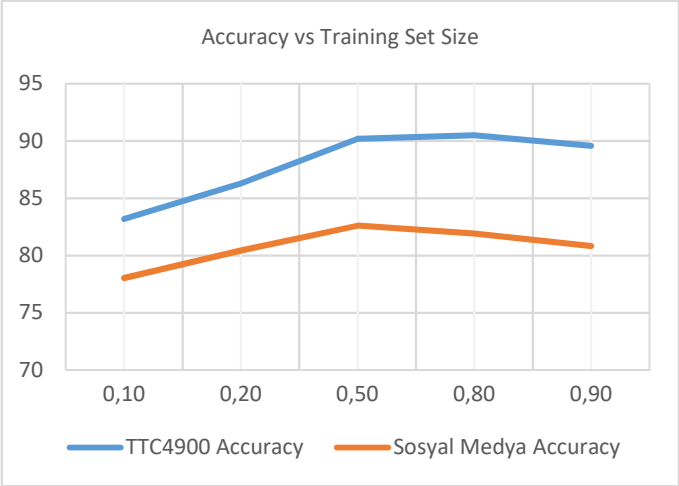
Şekil 4.2 Naive Bayes sınıflandırıcı algoritması ile oluşturulan iki adet modelin SMV ve TTC4900 ile eğitimi sonucunda oluşan test veri setleri için heatmap şeklinde hata matrisi

Multinomial Naive Bayes sınıflandırıcı ile oluşturulan modellerin eğitim veri setlerindeki yüzdesel değişime karşı doğrulama doğruluklarını gözlemlemek amacıyla modeller farklı oranlarda eğitim veri setleri ile yeniden eğitilmiştir. Eğitime ilişkin eğitim için kullanılan veri setleri oranı ve doğruluk yüzdeleri Tablo 4.2'de verilmiştir. Modelin genel olarak eğitim verisi arttıkça doğrulama doğruluğu artmıştır. Sürekli olarak artışa karşı istisna; eğitim verileri ayrılırken eğitim için kullanılan verilerin değişmesinden kaynaklanmaktadır. Bazı eğitim verileri modelin daha ayırt edici kelimelerle kelime dağarcığı oluşturmasını sağlar. Ancak bazı verilerde model ayırt edici kelimeleri tespit edemez.

Tablo 4.2 Multinomial Naive Bayes sınıflandırıcı ile oluşturulan modellerin TTC4900 ve Türkçe Sosyal Medya Verileri veri setlerindeki doğrulama doğruluğu

Veri Seti Adı	TTC4900	Türkçe Sosyal Medya Verileri
Eğitim Verisi Oranı	Doğruluk Yüzdesi	Doğruluk Yüzdesi
0.1	83.20	78.05
0.2	86.28	80.46
0.5	90.20	82.62
0.8	90.51	81.92
0.9	89.59	80.85

Artışı grafiksel olarak görmek için Tablo 4.2'nin grafiğe dönüştürülmüş hâli Şekil 4.3'te verilmiştir.



Şekil 4.3 Tablo 4.2'nin grafik hali

Naive Bayes sınıflandırıcı ile oluşturulan ve TTC4900 veri seti ile eğitilen modelin yanlış sınıflandırdığı metin örneği Şekil 4.4'te görülmektedir.

```
Text:  türkiye nin ilk spor turizmi fuarı türkiye nin ilk spor turizmi fuarı 1 3 kasım
tarihleri arasında antalya da gerçekleştirildi türkiye a nin konusunda ilk fuarı olma
özelliğine sahip a anfaş sporturkey a 01 a 03 kasım 2012 tarihleri arasında antalya expo
center a da gerçekleştirildi ülkemizin en geniş hizmet ağına sahip spor iletişim firması
olan euroasiasports a un fuar için özel hazırladığı ülkemizi ile tesislerimizi anlatan
kitap ve cd a yi tanıttığı anfaş sporturkey fuarını yaklaşık 15 bin kişi ziyaret etti
fuara katılan ulusal ve uluslar arası birçok firma ikili görüşmelerle ülkemizin spor
turizmi alanındaki tanıtımını gerçekleştirip gelecek yıl için rezervasyonlar yaptırdı
fuara birçok türk ve yabancı ülke spor federasyonu yanı sıra otel ve seyahat acenteleri
katıldı
Tahmini Etiket: ekonomi
Gerçek Etiket: spor
```

Şekil 4.4 Naive Bayes sınıflandırıcısının yanlış sınıflandırdığı metin örneği

Tüm modellerin de eğitiminde kullanılan %80 veri setleri ile eğitilen modellerin %10 test veri kümeleri ile test edilmesi sonucu oluşan test doğruluğu ve f1 skoru tabloları da aşağıda görülmektedir.

Tablo 4.3 Eğitilen Naive Bayes modellerin SMV ve TTC4900 veri kümelerine ait test verilerinin doğruluk ve f1 skoru (yüzdesel olarak)

Veri Setleri	Doğruluk	F1 Skoru
SMV	85	85
TTC4900	91.8	91.8

4.2. ESA ile Oluşturulan Modellere İlişkin Bulgular

Deneydeki modellerin karşılaştırılabilmeleri için tüm modellerde aynı eğitim ve test veri setleri kullanılmalıdır. Bu nedenle evrişimli sinir ağları ile oluşturulan modeller, tüm modellerde eğitim için ayrılan TTC4900 ve SMV veri setlerinin %80'lik dilimi ile eğitilmiştir. Bu modellere ilişkin bulgular bu kısımda verilecektir.

Modeller eğitilmeden önce veri setlerindeki metinler, ön işlem adımlarından geçer ve metinlerdeki kelimelerin vektör temsilleri oluşturulur. Eğer sınıflandırma için ESA kullanılacaksa metindeki vektör boyutlarının eşit olması gerekir. Bu nedenle dolgulama (padding) yapılarak tüm metin verilerinin aynı uzunlukta vektöre dönüştürülmesi gerekmektedir. Veri kaybı olmaması için maksimum boyut vektör dizilerindeki en uzun cümle uzunluğuna göre belirlenerek, eğitim ve test veri setlerine aşağıdaki şekilde dolgulama (padding) işlemi yapılmıştır.

```
# En uzun cümle uzunluğunu bulma
max_length = max([len(seq) for seq in X_train_vec + X_test_vec])

# Vektörleri aynı uzunluğa getirme (padding)
X_train_vec = pad_sequences(X_train_vec, maxlen=max_length)
X_test_vec = pad_sequences(X_test_vec, maxlen=max_length)
```

Şekil 4.5 Dolgulama (padding) işlemi ile ilgili kod parçası

Veri kümesindeki sınıflandırma etiketleri numerik olmadığı için ya numerik temsilleri ya da one-hot encoding vektör temsilleri oluşturulur.

```
y_train_encoded = pd.get_dummies(y_train)
y_test_encoded = pd.get_dummies(y_test)
```

Şekil 4.6 Etiketleri sayısal forma dönüştüren kod parçası

Bu tez çalışmasındaki, deneylerde kullanılan ESA modelinin katmanlarına ilişkin kod Şekil 4.7'de görülmektedir.

```
# CNN modelini oluşturma
model = Sequential()
model.add(Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=100, input_length=max_length))
model.add(Conv1D(128, 5, activation='relu'))
model.add(GlobalMaxPooling1D())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(y_train_encoded.shape[1], activation='softmax'))

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

Şekil 4.7 Modelin oluşturulması

Modeldeki katmanlara ve işlevlerine kısa bir genel bakış şu şekildedir:

- **Embedding:** Bu katman, tam sayı kodlu metin verilerini girdi olarak alır ve her tam sayıyı (bir kelimeyi temsil eden) sabit boyutta bir yoğun vektöre eşler (bu durumda 100). Elde edilen çıktı (batch_size, input_length, output_dim) şeklinde bir 3B tensördür.

- **Conv1D:** Bu katman girdi verilerine 1D evrişim işlemi uygular. 128 filtre ve 5 çekirdek boyutuna sahiptir ve ReLU aktivasyon işlevini kullanır. Bu katmanın çıktısı, girdi verilerindeki yerel desenleri veya özellikleri temsil eden bir özellik haritasıdır.

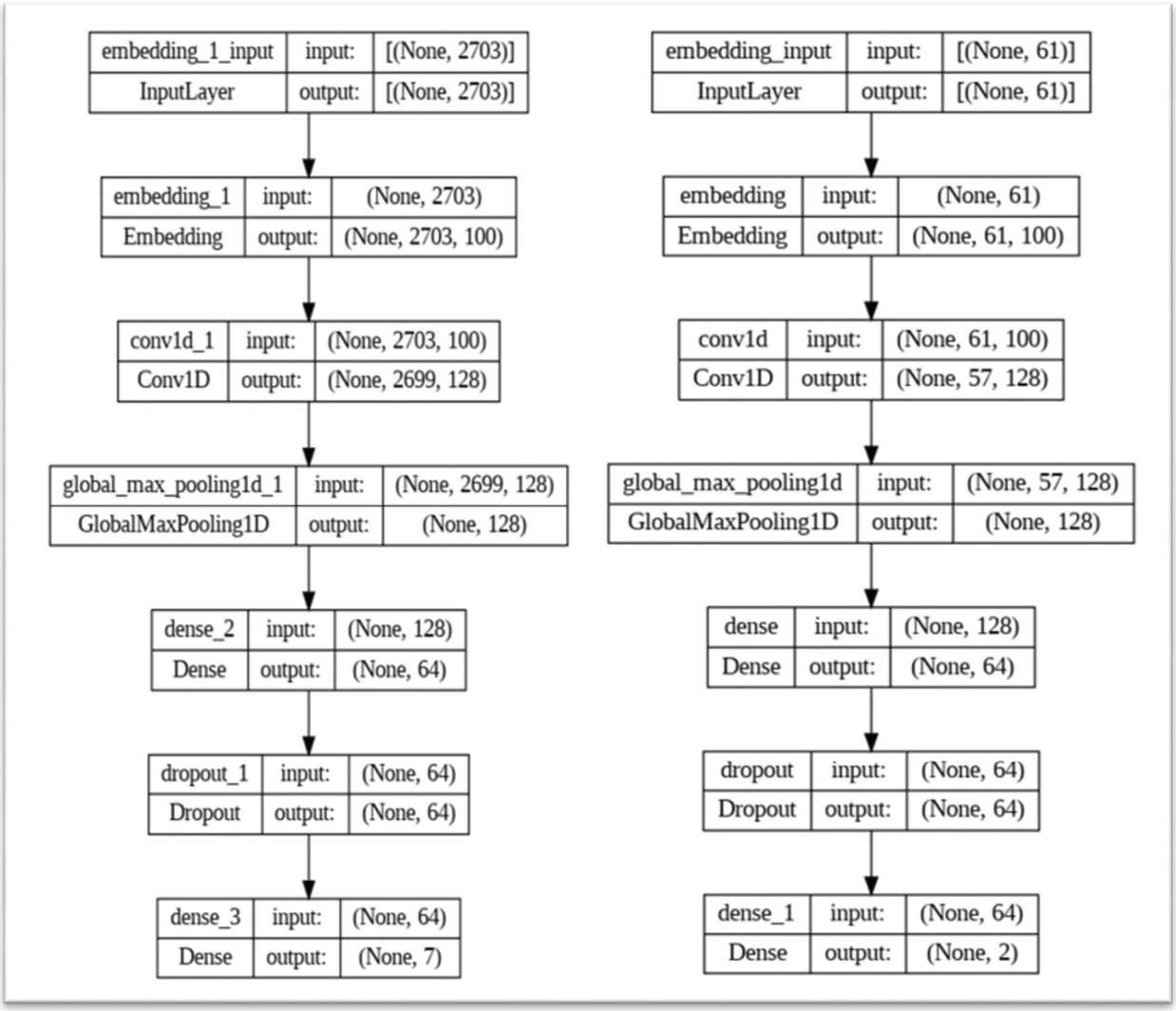
- **GlobalMaxPooling1D:** Bu katman, önceki katman tarafından üretilen özellik haritasına global max pooling uygular. Her filtre için tüm özellik haritası üzerinde maksimum değeri alarak verilerin boyutunu azaltır. Bu katmanın çıktısı (batch_size, num_filters) şeklinde bir 2B tensördür.

- **Dense:** Bu katman 64 birimli tamamen bağlı bir katmandır ve ReLU aktivasyon işlevini kullanır. Önceki katmanın çıktısını girdi olarak alır ve doğrusal bir dönüşüm uyguladıktan sonra eleman bazında doğrusal olmayan bir aktivasyonla yeni bir özellik kümesi üretir.

- **Dropout:** Bu katman, önceki katmanın çıktısına 0.2 oranında dropout düzenlemesi uygular. Aşırı uyumu önlemek için eğitim sırasında bazı girdi birimlerini rastgele 0 olarak ayarlar.

- **Dense:** Bu modelin çıkış katmanıdır. Hedef verilerdeki sınıflar kadar birime sahiptir ve sınıf olasılıklarını üretmek için softmax aktivasyon işlevini kullanır.

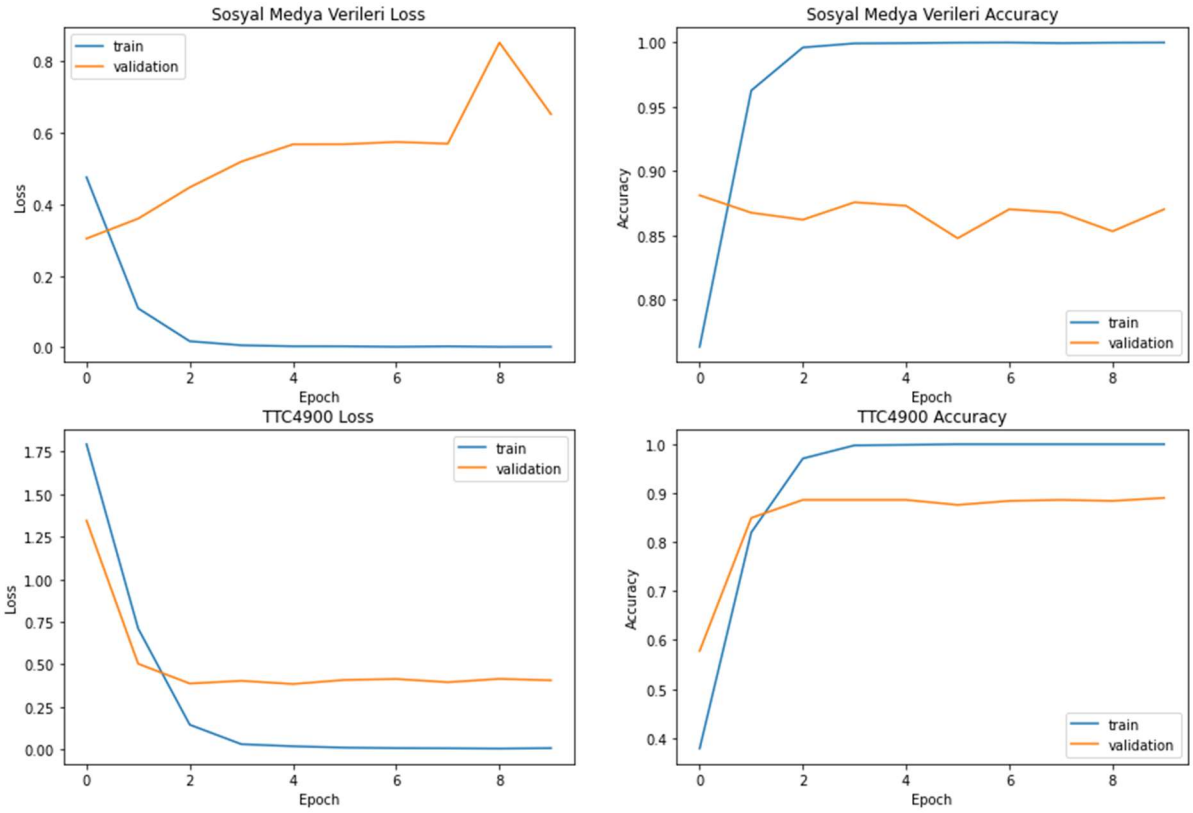
ESA ile oluşturulan modellerin katmanları Şekil 4.8'de görülmektedir.



(a) **(b)**

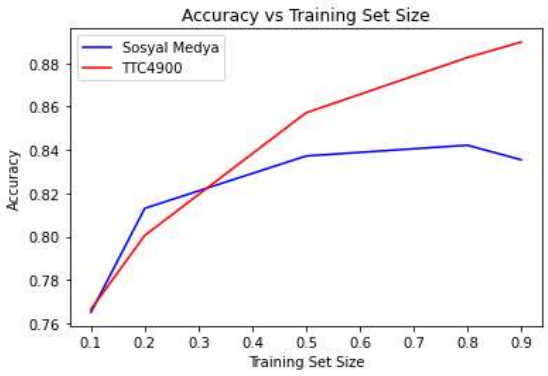
Şekil 4.8 ESA kullanılan modelin katmanları. **a**:TTC4900, **b**: Sosyal Medya Verileri (Veri setine göre output ve input katmanları değişmektedir.)

Modellerde kayıp fonksiyonu olarak kategorik çapraz entropi, optimizasyon algoritması olarak adam optimizer kullanılmıştır. Bu şekilde tasarlanan modeller 10 epoch eğitilmiştir. Modellerin eğitimi ve doğrulamasına ilişkin epoch başına eğitim kaybı ve doğruluğu, doğrulama kaybı ve doğruluğu grafikleri Şekil 4.9'da görülmektedir. Sosyal Medya Verileri ile eğitilen modelin eğitim kaybı sürekli azalırken doğrulama kaybı artmaktadır. Bu göstergeye dayanarak dropout miktarı artırılmıştır. Ancak doğrulama kaybının azalmadığı görülmüştür. Bu durumun modelin eğitiminde kullanılan eğitim veri setinin yetersiz olmasından kaynaklandığı düşünülmektedir.



Şekil 4.9 ESA ile oluşturulan modellerin SMV ve TTC4900 ile eğitiminde her epoch için kayıp ve doğruluk grafikleri

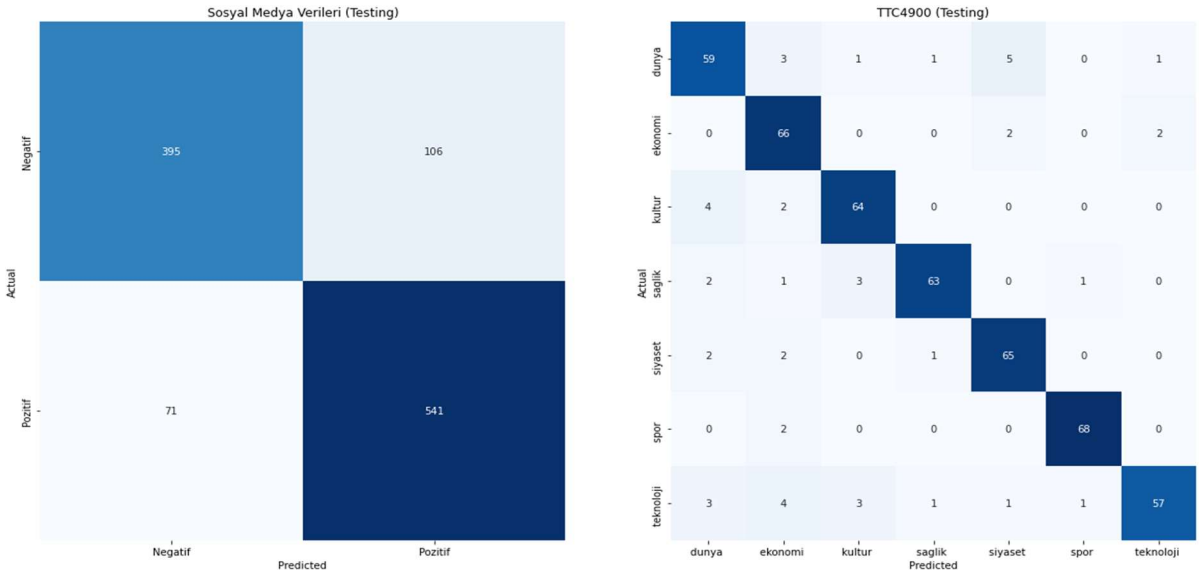
ESA ile oluşturulan modellerin eğitim veri setlerindeki yüzdesel değişime karşı doğrulama doğruluklarını gözlemlemek amacıyla modeller farklı oranlarda eğitim veri setleri ile yeniden eğitilmiştir. Eğitime ilişkin veri setleri ve doğruluk grafiği Şekil 4.10'da verilmiştir.



Şekil 4.10 Eğitim veri seti oranı ve doğruluk grafiği

Eğitilen modelin performansını ölçen diğer yöntem hata matrisinin oluşturulmasıdır. Daha önce de bahsedildiği üzere hata matrisinden modele ilişkin başarım metrikleri hesaplanabilir. Veri setinde birden fazla sınıf ya da sınıflar arasında dengesizlik olduğu durumlarda sadece

sınıflandırma doğruluk metriğini kullanmak yanıltıcı olabilir. Bu durumda diğer metriklerin kullanılması gerekir. Bu metriklerin hesaplanmasında hata matrisine bakılabilir.



Şekil 4.11 ESA ile oluşturulan modellerin test veri setindeki tahminlerinin heatmap şeklindeki hata matrisi

Şekil 4.11'den SMV ile eğitilen modelde pozitif kategorideki verilerin doğru tahmin oranının daha yüksek olduğu görülmektedir. TTC4900 veri seti ile eğilen verilerden dünya ve teknoloji kategorisindeki verilerde doğru tahmin oranı diğer kategorilerdeki verilere göre düşüktür.

Modelin 10 epoch eğitimi sonundaki test doğruluk ve f1 skoru değerleri Tablo 4.4'te görülmektedir.

Tablo 4.4 Eğitilen ESA modellerinin SMV ve TTC4900 veri kümelerine ait test verilerinin doğruluk ve f1 skoru (yüzdesel olarak)

Veri Setleri	Doğruluk	F1 Skoru
SMV	85	84
TTC4900	90	90

4.3. UKSB ile Oluşturulan Modellere İlişkin Bulgular

UKSB (Uzun-kısa süreli bellek) modelinde de metin öncelikle ön işlemden geçirilir. Metindeki kelimelerin vektör temsilleri oluşturulur. UKSB modelleri farklı boyuttaki girdi verilerini işleyebilir. Ancak bu tez çalışmasındaki modelde Tensorflow ve Keras kütüphaneleri kullanıldığı ve bu kütüphanelerde girişte sabit bir veri boyutu tanımlandığı için giriş verilerinin

aynı boyutta olması gerekmektedir. Bunun için dolgulama işlemi yapılır. Veri işleme aşamasında dolgulu değerleri ihmal etmek için maskeleme tekniği kullanılabilir.

```
model.add(Embedding(input_dim=input_dim + 1, output_dim=100, mask_zero=True))
```

Şekil 4.12 Embedding katmanında mask_zero=True yapılarak dolgu değerleri ihmal edilir.

Sınıflandırma için kullanılacak etiketler numerik değilse numerik hale getirilir. Bunun için one-hot encoding yöntemi ya da manuel bir şekilde her kategoriye karşılık bir değer atanabilir. UKSB modeline ilişkin katmanlar Şekil 4.13'te görülmektedir.

```
model = Sequential()
model.add(Embedding(input_dim=input_dim + 1, output_dim=100, mask_zero=True))
model.add(LSTM(128))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(output_dim, activation='softmax'))

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

Şekil 4.13 UKSB modeli

UKSB modelindeki katmanlara ve işlevlere kısa bir genel bakış şu şekildedir:

- **Embedding:** Bu katman, tam sayı kodlu metin verilerini girdi olarak alır ve her tam sayıyı (bir kelimeyi temsil eden) sabit boyutta bir yoğun vektöre eşler (bu durumda 100). Elde edilen çıktı (batch_size, input_length, output_dim) şeklinde bir 3B tensördür.

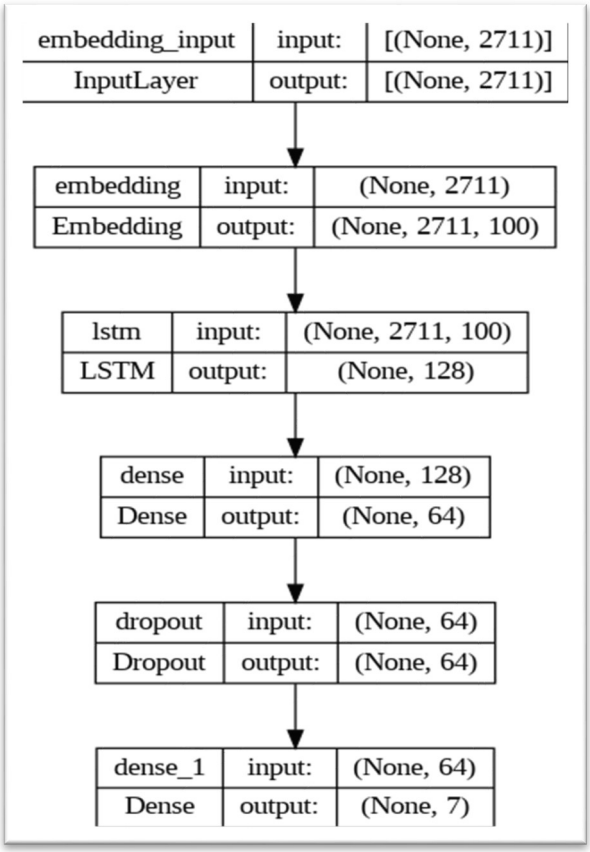
- **LSTM:** Bu katman, Uzun Kısa Vadeli Bellek (LSTM) birimlerini kullanan bir tekrarlayan sinir ağı (RNN) türüdür. 128 birime sahiptir ve önceki katmanın çıktısını girdi olarak alır. LSTM katmanı, dâhili bellek hücrelerini kullanarak girdi verilerindeki uzun vadeli bağımlılıkları yakalayabilir.

- **Dense:** Bu katman 64 birimli tamamen bağlı bir katmandır ve ReLU aktivasyon işlevini kullanır. Önceki katmanın çıktısını girdi olarak alır ve doğrusal bir dönüşüm uyguladıktan sonra eleman bazında doğrusal olmayan bir aktivasyonla yeni bir özellik kümesi üretir.

- **Dropout:** Bu katman, önceki katmanın çıktısına 0.2 oranında dropout düzenlemesi uygular. Aşırı uyumu önlemek için eğitim sırasında bazı girdi birimlerini rastgele 0 olarak ayarlar.

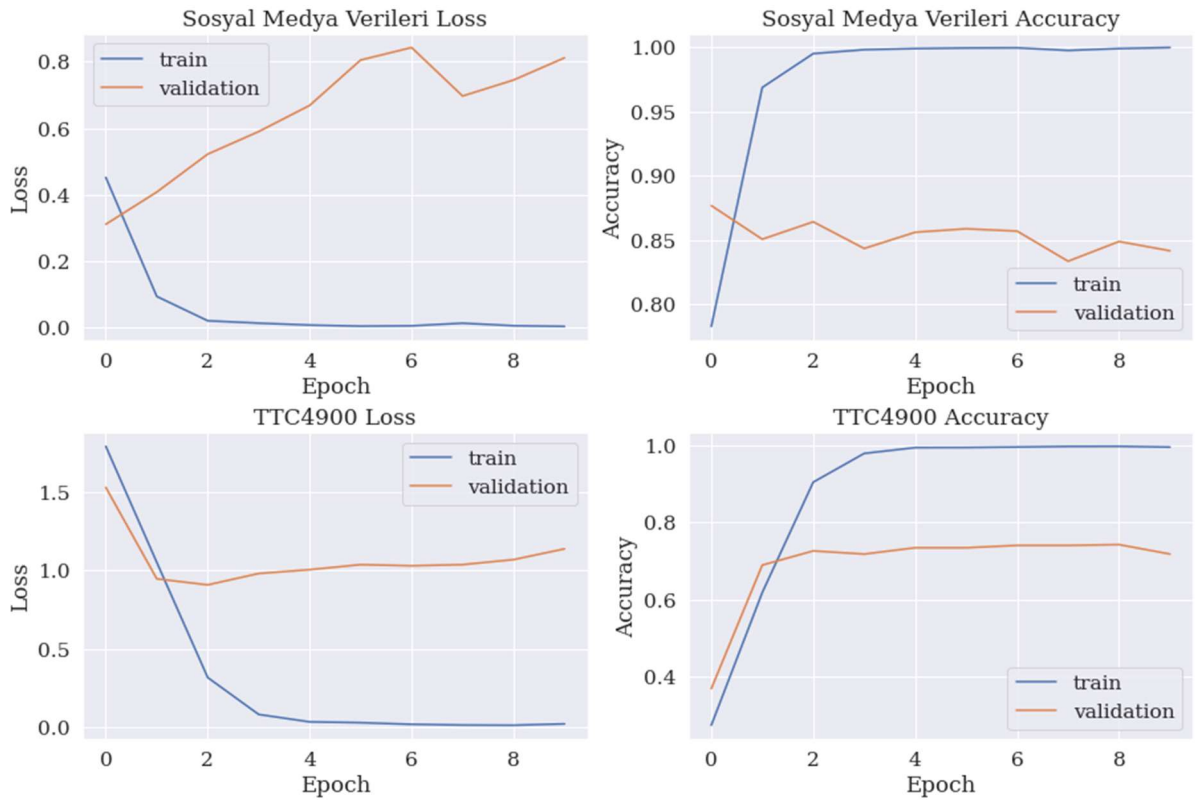
- **Dense:** Bu modelin çıkış katmanıdır. Hedef verilerdeki sınıflar kadar birime sahiptir ve sınıf olasılıklarını üretmek için softmax aktivasyon işlevini kullanır.

Model, çok sınıflı sınıflandırma problemleri için yaygın olarak kullanılan kategorik çapraz entropi kayıp işlevi ile derlenmiştir ve Adam optimizer kullanılmıştır. TTC4900 ile eğitilen modelin katmanları Şekil 4.14'te görülmektedir.



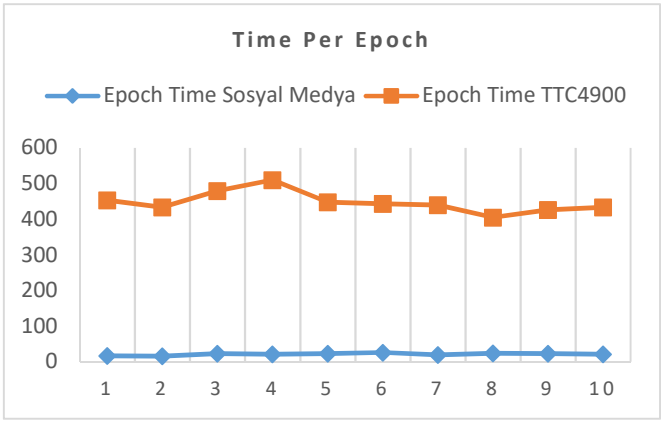
Şekil 4.14 TTC4900 veri seti ile eğitilen LSTM modelinin katmanları

Modellerin eğitimi ve doğrulamasına ilişkin epoch başına eğitim kaybı ve doğruluğu doğrulama kaybı ve doğruluğu grafikleri Şekil 4.15'te görülmektedir.



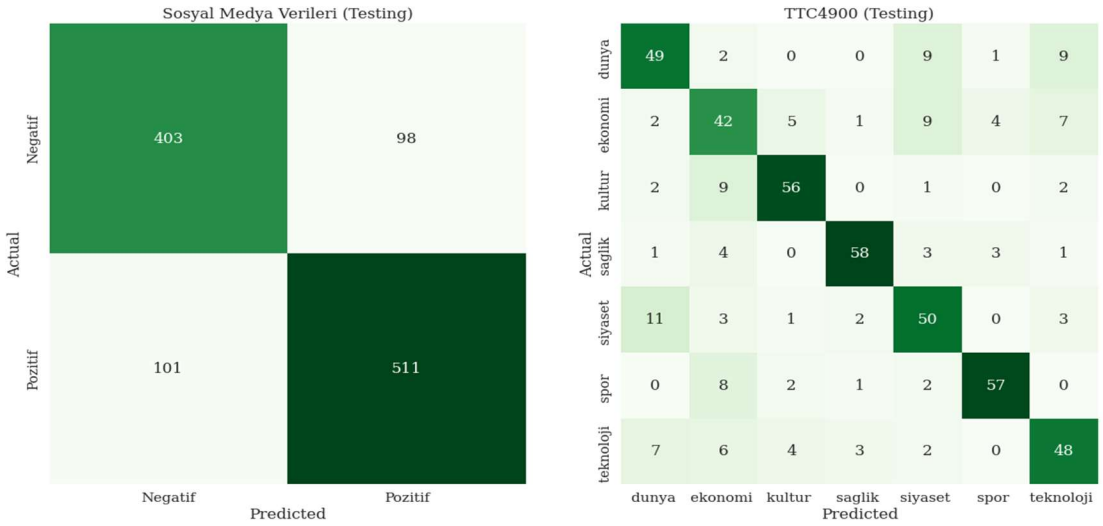
Şekil 4.15 UKSB ile oluşturulan modellerin SMV ve TTC4900 ile eğitiminde her epoch için kayıp ve doğruluk grafikleri

Şekil 4.15'de görüldüğü üzere model sosyal medya verileri veri seti ile yapılan eğitimde 1. Epoch'tan itibaren aşırı uyum sağlamıştır. ESA ile oluşturulan modelde de aynı veri seti için aynı sorunlar görülmüştür. Ayrıca, UKSB modeli TTC4900 ile eğitilirken epoch başına eğitim uzun sürmüş ve çok zaman kaybedilmiştir. İki modelin veri setleri ile eğitilirken epoch başına oluşan zaman kaybı Şekil 4.16'da gösterilmektedir.



Şekil 4.16 UKSB modeli eğitilirken epoch başına geçen zaman

Modellerin test verileri ile test edilmesine ilişkin hata matrisi Şekil 4.17'de görülmektedir.



Şekil 4.17 UKSB sınıflandırıcı algoritması ile oluşturulan iki adet modelin SMV ve TTC4900 ile eğitimi sonucunda oluşan test veri setleri için heatmap şeklinde hata matrisi

Modelin 10 epoch eğitimi sonundaki test doğruluk ve f1 skoru değerleri Tablo 4.5'te görülmektedir.

Tablo 4.5 Eğitilen UKSB modellerinin SMV ve TTC4900 veri kümelerine ait test verilerinin doğruluk ve f1 skoru (yüzde olarak)

Veri Setleri	Doğruluk	F1 Skoru
SMV	82	82
TTC4900	73.5	73.7

4.4. GCN ile Oluşturulan Modellere İlişkin Bulgular

Çizge evrişimli sinir ağları ile sınıflandırma yapabilmek için metin verisinin bir çizge şeklinde ifade edilmesi gereklidir. Oluşturulan iki katmanlı GCN modeline eğitim için özellik matrisi, komşuluk matrisi ve eğitim veri kümesinin sınıf etiketleri verilir. Modele eğitim için verilmeden önce sınıf etiketlerinin one-hot temsilleri oluşturulur. Özellik matrisi düğüm sayısı ile aynı boyutta birlerden oluşan köşegen matrisidir. Çizge düğümlerinin one-hot temsili de denilebilir.

Metin tabanlı GCN modelinde [1], metnin çizge şeklinde ifade edilmesi için öncelikle PMI ve TF-IDF değerleri hesaplanır. PMI hesaplaması için kelime eşzamanlılığı (co-occurence) hesaplanır. Belirli bir pencere boyutuna(window size) göre kelime çiftleri ve sayısı hesaplanır.

Bu tez çalışması için Yao vd. [1] çalışmalarında da kullandığı gibi pencere boyutu 20 seçilmiştir. Tespit edilen kelime çiftlerinden PMI değeri Şekil 4.18'deki gibi hesaplanır.

```
row = []
col = []
weight = []

# pmi as weights

num_window = len(windows)

for key in word_pair_count:
    temp = key.split(',')
    i = int(temp[0])
    j = int(temp[1])
    count = word_pair_count[key]
    word_freq_i = word_window_freq[vocab[i]]
    word_freq_j = word_window_freq[vocab[j]]
    pmi = log((1.0 * count / num_window) /
              (1.0 * word_freq_i * word_freq_j/(num_window * num_window)))
    if pmi <= 0:
        continue
    row.append(train_size + i)
    col.append(train_size + j)
    weight.append(pmi)
```

Şekil 4.18 Kelime çiftlerinden PMI değerinin hesaplanması

PMI kelime-kelime frekansını, TF-IDF ise doküman-kelime frekansını hesaplar. PMI ve TF-IDF yöntemleri, genellikle doğal dil işleme ve metin madenciliği gibi alanlarda kullanılır. Bu yöntemler sayesinde kelime-kelime ilişkileri analiz edilir ve doküman içindeki kelimelerin önem dereceleri belirlenir.

Komşuluk matrisi, bir veri setindeki öğelerin birbirleriyle olan ilişkilerini temsil eden bir matristir. PMI ve TF-IDF hesaplamalarından elde edilen değerler kullanılarak komşuluk matrisi oluşturulur. Bu matris, öğelerin birbirleriyle olan benzerlik veya ilişkilerini gösteren bir yapıdır. TF-IDF hesabı ve komşuluk matrisinin oluşturulması ile ilgili kod parçası Şekil 4.19'da görülmektedir. Kodda Adj değeri komşuluk matrisidir. PMI ve TF-IDF hesaplamalarından gelen row, col ve weight ile matris oluşturulur.

```
# doc word frequency
doc_word_freq = {}

for doc_id in range(len(shuffle_doc_words_list)):
    doc_words = shuffle_doc_words_list[doc_id]
    words = doc_words.split()
    for word in words:
        word_id = word_id_map[word]
        doc_word_str = str(doc_id) + ',' + str(word_id)
        if doc_word_str in doc_word_freq:
            doc_word_freq[doc_word_str] += 1
        else:
            doc_word_freq[doc_word_str] = 1

for i in range(len(shuffle_doc_words_list)):
    doc_words = shuffle_doc_words_list[i]
    words = doc_words.split()
    doc_word_set = set()
    for word in words:
        if word in doc_word_set:
            continue
        j = word_id_map[word]
        key = str(i) + ',' + str(j)
        freq = doc_word_freq[key]
        if i < train_size:
            row.append(i)
        else:
            row.append(i + vocab_size)
        col.append(train_size + j)
        idf = log(1.0 * len(shuffle_doc_words_list) /
                  word_doc_freq[vocab[j]])
        weight.append(freq * idf)
        doc_word_set.add(word)

node_size = train_size + vocab_size + test_size
adj = sp.csr_matrix(
    (weight, (row, col)), shape=(node_size, node_size))
```

Şekil 4.19 TF-IDF hesabı ve komşuluk matrisinin oluşturulması

Komşuluk matrisi adj'nin normalize edilmesi gereklidir. Matris normalizasyonu, bir matrisin değerlerini belirli bir ölçekte birleştirerek ve aralıklarını değiştirerek matrisi daha karşılaştırılabilir hale getirmeyi amaçlar. Normalleştirme işlemi, matrisin en büyük özdeğerini 1'e eşitlemek için yapılır. Bu sayede, ağdaki katman sayısı arttıkça değerlerin patlamasını önlemek için kullanılır.

Bir komşuluk matrisini normalleştirmek, çizge evrişimli sinir ağlarında sayısal kararlılığı sağlar. Daha önce de denklem 3.16'da belirtildiği gibi matris normalleştirme işleminin matematiksel formülü 3.16'daki gibidir.

$$A_{hat} = D^{\frac{-1}{2}} * A * D^{-\frac{1}{2}} \tag{3.16}$$

Matris normalleştirme işlemine ilişkin fonksiyonların kodu Şekil 4.20'da görülmektedir.

```
def normalize_adj(adj):
    adj = sp.coo_matrix(adj)
    rowsum = np.array(adj.sum(1))
    d_inv_sqrt = np.power(rowsum, -0.5).flatten()
    d_inv_sqrt[np.isinf(d_inv_sqrt)] = 0.
    d_mat_inv_sqrt = sp.diags(d_inv_sqrt)
    return adj.dot(d_mat_inv_sqrt).transpose().dot(d_mat_inv_sqrt).tocoo()


def preprocess_adj(adj):
    adj_normalized = normalize_adj(adj + sp.eye(adj.shape[0]))
    return sparse_to_tuple(adj_normalized)
```

Şekil 4.20 Komşuluk matrisini normalleştirme fonksiyonları

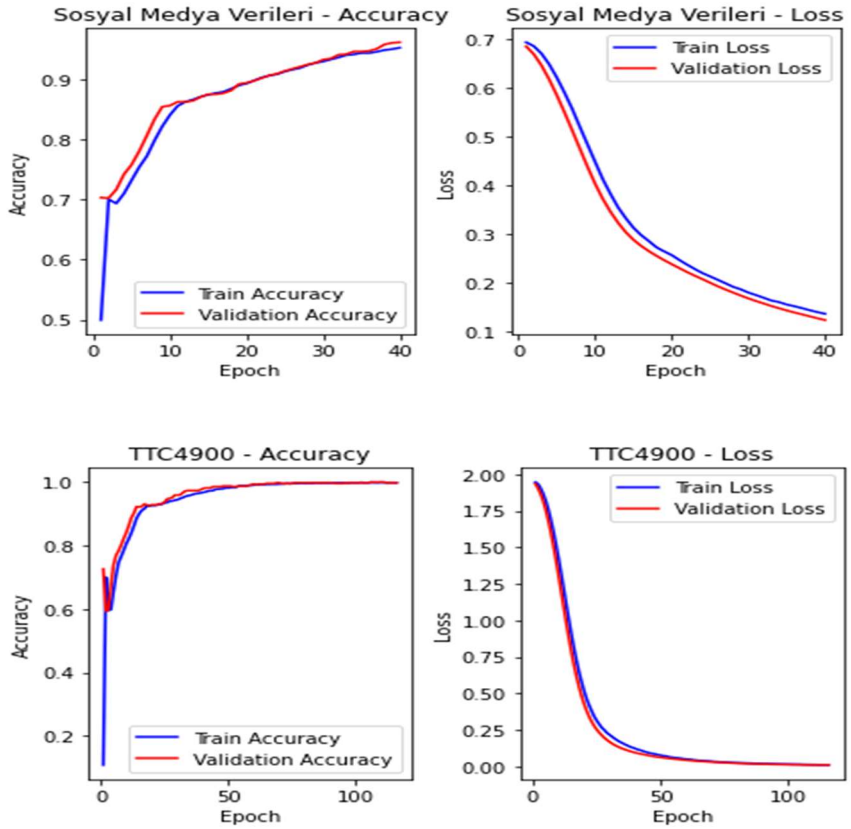
Normalleştirilmiş komşuluk matrisi, özellik matrisi ve eğitim verilerine ilişkin sınıf etiketleri GCN modelinin eğitiminde kullanılır. Bu tez çalışmasında iki katmanlı GCN modeli oluşturulmuştur.

Eğitime ilişkin hiperparametreler aşağıda gösterilmektedir.

Tablo 4.6 GCN modelindeki hiperparametreler

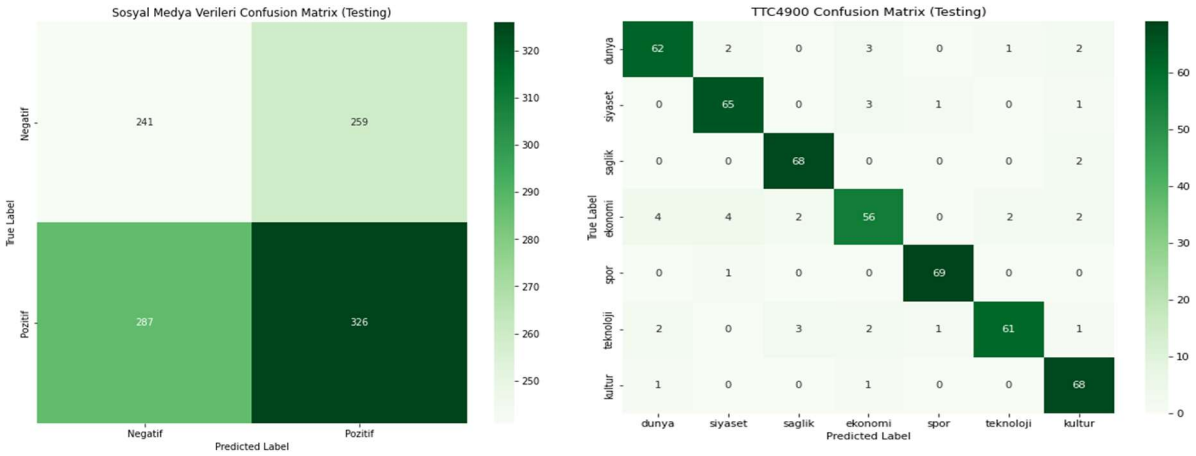
Hiperparametre	Değer	Açıklama
Model	GCN	Çizge evrişimli sinir ağı
Learning rate	0.02	Başlangıç öğrenme oranı
epochs	200	Eğitim dönem sayısı
dropout	0.5	Dropout oranı
early_stopping	10	Erken durdurma toleransı (dönem sayısı)

GCN modeli yukarıdaki parametreler ile eğitildiğinde erken durdurma (early stopping) kullanıldığı için 200 parametre eğitim beklenmeden erken durdurulmuştur. Erken durdurma parametresi 10 olarak ayarlandığı için modelin doğrulama verilerindeki performansı 10 tur boyunca gelişmezse eğitim erken durdurulur. Modelin eğitimi sonucu elde edilen grafikler Şekil 4.21'de görülmektedir.



Şekil 4.21 GCN ile oluşturulan modellerin SMV ve TTC4900 ile eğitiminde her epoch için kayıp ve doğruluk grafikleri

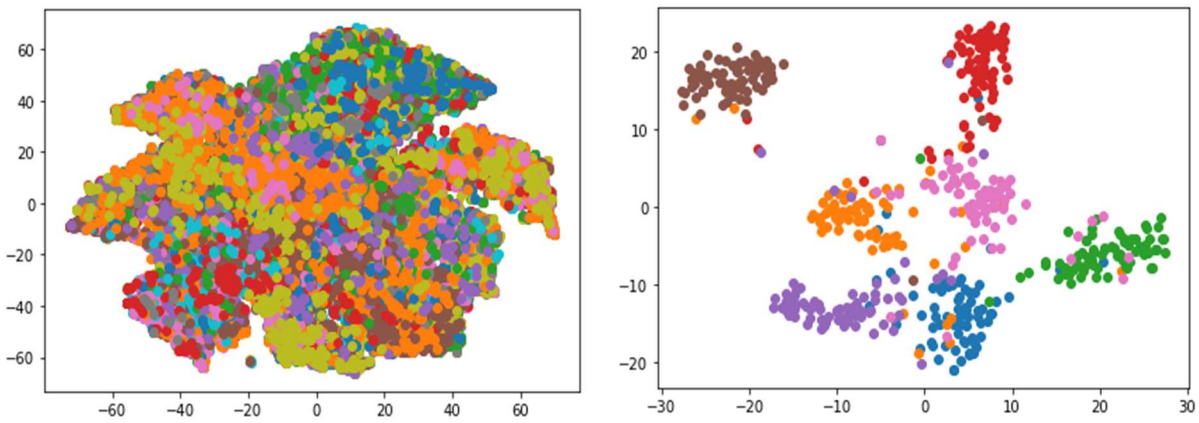
Modelin performansını değerlendirmek için hata matrisi oluşturulmuştur. Modellere ilişkin hata matrisi Şekil 4.22'de verilmiştir.



Şekil 4.22GCN ile oluşturulan iki adet modelin SMV ve TTC4900 ile eğitimi sonucunda oluşan test veri setleri için heatmap şeklinde hata matrisi

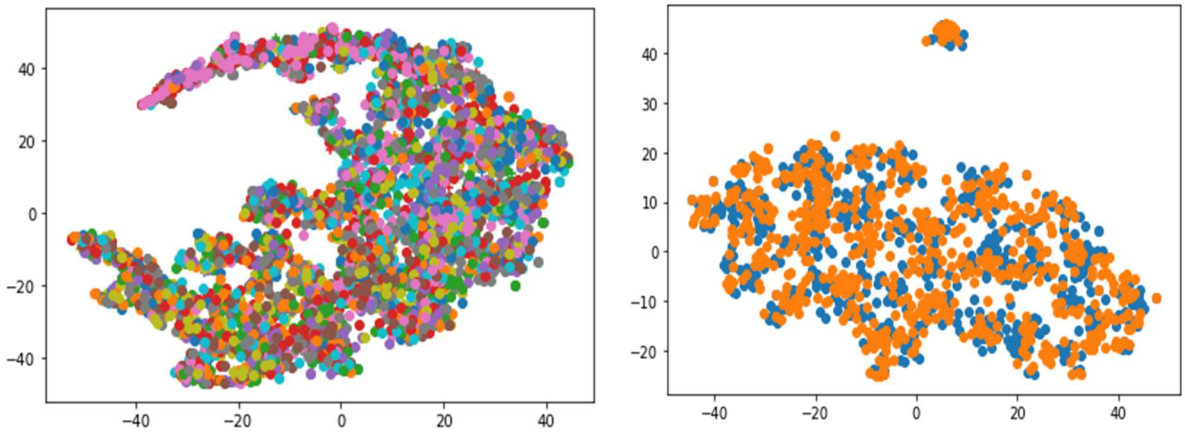
TextGCN modeli, bir metin grafiği oluşturarak hem kelimeler hem de belgeler için gömme öğrenir. Modelin ikinci katmanında, kelimelerin gömmeleri, metin grafiğindeki komşu kelimelerin gömmelerinin ortalaması alınarak hesaplanır. Bu şekilde, kelimelerin gömmeleri, metin kümesindeki kelime eşzamanlılığına ve belge kelime ilişkilerine dayalı olarak güncellenir.

Metin tabanlı GCN modelinin ikinci katmanında öğrenilen kelime temsillerinin t-SNE görselleştirmesi Şekil 4.23 ve Şekil 4.24'te gösterilmektedir. Kelime vektör temsilleri oluşturulduktan sonra en yüksek değere sahip boyut kelime etiketi olarak belirlenir. Örneğin, [0.2, -0.5, 0.7, 0.1] şeklinde bir gömme vektörü olsun. Bu vektörde en yüksek değer 0.7'dir ve en yüksek değere sahip boyut ise 3'tür. En yüksek değere sahip boyutu bir kelimenin etiketi olarak belirlemek, kelimeleri gömme uzayında farklı bölgelere ayırmak için bir yöntemdir. Bu yöntemde aynı etikete sahip veriler birbirine yakın olarak görülmektedir. TTC4900 veri seti özelinde kelime ve belge sınıflarının benzer olması çoğu kelimenin belge sınıflarıyla yakından ilişkili olduğunu gösterebilir. Sosyal medya verileri kelime temsillerinde ise kelimeler daha dağınık bir görünüm sergilemektedir. Bu da belirli sınıfa ait belirli kelime gömmesinin model tarafından tespit edilemediği anlamına gelebilir.



(a) **(b)**

Şekil 4.23 (a) TTC4900 veri seti için kelime vektörlerinin t-SNE görselleştirmesi. (b) TTC4900 veri seti için iki boyutlu uzayda belge vektörlerinin t-SEN görselleştirmesi. Her nokta bir kelimeyi(a) veya belgeyi(b)temsil eder ve farklı sınıflar farklı renklerle ifade edilir.



(a) **(b)**

Şekil 4.24 (a) Sosyal Medya Verileri veri seti için iki boyutlu uzayda kelime vektörlerinin t-SNE görselleştirmesi. (b)Sosyal Medya Verileri veri seti için iki boyutlu uzayda belge vektörlerinin t-SNE görselleştirmesi. Her nokta bir kelimeyi (a) ya da belgeyi (b) ifade eder. Farklı sınıflar farklı renklerle gösterilir.

Tablo 4.7 Eğitilen UKSB modellerinin SMV ve TTC4900 veri kümelerine ait test verilerinin doğruluk ve f1 skoru (yüzdesel olarak)

Veri Setleri	Doğruluk	F1 Skoru
SMV	50.9	50.9
TTC4900	91.63	91.56

4.5. Nihai Sonuçlar

Modellerin doğruluk metriği üzerinden değerlendirilmesi sonucunda oluşan model doğruluk yüzdeleri aşağıdaki tabloda verilmiştir.

Tablo 4.8 Eğitilen modellerin test doğruluğu yüzdesi

Sınıflandırıcı Model	Metrik	TTC4900	Türkçe Sosyal Medya Verileri
Naive Bayes	Doğruluk(%)	91.8	85
ESA	Doğruluk(%)	90	85
UKSB	Doğruluk(%)	73.5	82
GCN	Doğruluk(%)	91.63	50.9

5. TARTIŞMA

Metin sınıflandırma, doğal dil işleme alanında önemli bir konudur. Metin sınıflandırma, metin verilerini belirli kategorilere ayırmayı amaçlar. Bu sayede, metin verileri organize edilebilir, yapılandırılabilir ve sınıflandırılabilir.

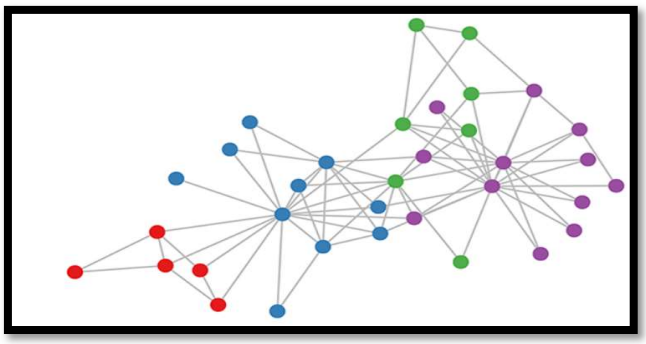
Metin verilerinin yapısal olmaması metin sınıflandırmayı zorlaştırmaktadır. Belirli bir algoritmaya ve kurallara göre çalışan bilgisayar yazılımları ile metin sınıflandırma işleminde metin dillerin karmaşık yapısı nedeniyle oldukça zordur. Bu bilgisayar yazılımlarına çok sayıda istisna tanımlanması gerekir. Her dile ve metne özgü istisnalar farklı olduğundan, kural seti oluşturmak yerine yapay zekâ yöntemlerine başvurulmaktadır. Bu tez çalışmasında yapay zekâ yöntemlerinden klasik makine öğrenmesi modeli Naive Bayes, derin öğrenme modellerinden ESA, UKSB ve GCN ile metin sınıflandırma yapılmaktadır.

Naive Bayes, olasılık temelli bir sınıflandırma yöntemidir. Bu yöntem, metin verilerini bir kelime çantası olarak ele alır ve her bir kelimenin sınıflar arasındaki olasılıklarını hesaplar. Daha sonra bu olasılıkları kullanarak belgeleri sınıflandırır. Naive Bayes gibi klasik makine öğrenmesi yöntemleri, elle hazırlanmış özelliklere dayanır ve karmaşık metin sınıflandırma görevlerinde iyi performans göstermeyebilir. Ayrıca, derin öğrenme modellerinden daha hızlı eğitilebilir ve daha kolay yorumlanabilirler. Derin öğrenme modelleri, genellikle daha karmaşık ve parametrik yapılara sahiptir. Bu modellerin eğitimi daha fazla veri ve hesaplama gücü gerektirebilir. Ayrıca, büyük veri setlerinde derin öğrenme modellerinin eğitimi zaman alıcı olabilir. ESA ve UKSB gibi derin öğrenme yöntemleri metin verilerini sıralı bir yapı olarak ele alır ve bu yapı üzerinde evrişim ve tekrarlı işlemler gerçekleştirir. Bu yöntemler metin verilerinin yerel ve global yapısını öğrenerek sınıflandırma işlemini gerçekleştirir. GCN'ler ise metin verilerini bir çizge olarak temsil eder ve bu çizge üzerinde evrişim işlemleri gerçekleştirir. Bu sayede, GCN'ler metin verilerinin yapısını ve ilişkilerini öğrenerek sınıflandırma işlemini gerçekleştirir. Çizge Evrişimsel Ağlar (GCN), metin verilerini grafik yapılar olarak modelleyerek metin sınıflandırma işlemlerinde kullanılabilir. GCN'ler, metin verilerinin global özelliklerini çıkarmada güçlüdür ancak kelimelerin sırasını dikkate almazlar. Bu nedenle, kısa metin sınıflandırmada zayıf bir etkiye sahiptirler. Buna karşılık, Evrişimli Sinir Ağları (ESA) bir cümle içindeki yerel bağlamı yakalayabilir. Hem uzun hem de kısa metni etkili bir şekilde

sınıflandırmak için önerilen bir çözüm, GCN ve ESA'yı birleştirerek bir toplu evrişim ağı kullanmaktır. Bu yöntemde, GCN global bilgileri yakalar ve ESA yerel özellikleri çıkarır [81].

Bu tez çalışmasında; UKSB modeli TTC4900 gibi uzun metinler içeren veri seti ile eğitilirken, eğitim için uzun zaman harcanmıştır. Ayrıca UKSB modelinin TTC4900 için test başarısı oldukça düşüktür. UKSB modelindeki zaman kaybı ve başarımın düşük olmasına çözüm olarak [82]'de çok uzun dizilerle çalışırken kısaltma, özetleme, rastgele örnekleme ve kesikli zaman geriye yayılımın kullanılabileceği belirtilmiştir. Ancak bu tez çalışması çok sayıda model içerdiği için UKSB modelinde iyileştirme yapılmadan model performansları karşılaştırılmıştır. UKSB modeli kısa metinler içeren Sosyal medya verileri veri setinde TTC4900'e göre iyi performans göstermiştir. Ayrıca eğitim süresi de TTC4900 veri setine göre daha kısadır. Sosyal medya verileri veri seti ile UKSB modeli eğitildiğinde %84.26 ile tüm modeller içinde en iyi başarım sağlanmıştır. Ancak modelin ilk epochtan itibaren aşırı uyum gösterdiği görülmektedir. Bu da veri setinin özelliğinden kaynaklanmaktadır. Veri seti kısa ve dilsel hatalar içermektedir.

Bu tez çalışmasında, GCN modeli olarak Yao vd. [1]'de sunduğu TextGCN modeli kullanılmıştır. Veri setlerinin %10'u test, %10'u doğrulama ve %80'i modellerin eğitimi için kullanılmıştır. Bu verilere göre modellerin doğruluk ve f1 skoru gibi başarımları gözlemlenmiştir. TextGCN modelinin diğer modellere göre sınıflandırmadaki avantajı, Yao vd. [1] ve Kipf vd. [44] TextGCN modeli ile yaptıkları çalışmada tespit ettikleri deneysel sonuçlara göre; çok az eğitim verisinde bile textGCN modeli diğer son teknoloji metin sınıflandırma modellerine göre daha dayanıklı bir duruş sergilemesidir. Bu konuyla ilgili Kipf'in [44] Zachary'nin karate kulübü veri seti ile yaptığı deneysel çalışmada [83], her sınıftan sadece bir etiketli düğümle bile modelin iyi bir sınıflandırma performansı gösterdiği gözlemlenmiştir.



Şekil 5.1 Zachary'nin karate kulübü [83]

Metin tabanlı GCN'nin sınıflandırma konusunda dezavantajları da vardır. Bunlardan biri, eğitilmiş model ile çevrimiçi test yapmanın oldukça zor olmasıdır. Bütün korpustan tek bir çizge oluşturulduğu için yeni gelen veri çizgenin yeniden oluşturulmasını modelin yeniden eğitilmesini gerektirebilir. GCN ile çevrimiçi test konusuna çözüm olarak Huang vd. [47] yaptıkları çalışmada korpustan tek bir çizge oluşturmak yerine her bir belge için ayrı bir çizge oluşturmuşlardır. Sınıflandırıcı olarak GCN'i kullandıkları bu yöntemde son teknoloji (state-of-the-art) modellerdeki performansı yakalamışlardır. Bu yöntem çevrimiçi testi desteklemesinin yanında bellek tüketimi konusunda da [1]'deki yönteme göre avantajlıdır.

Veri setlerine dair literatürdeki çalışmalar incelendiğinde Sosyal Medya Verileri veri setine ilişkin çalışmaya rastlanmamıştır. TTC4900 veri setine ilişkin literatür çalışmalarına bakıldığında Yıldırım vd. [49] farklı kelime temsil yöntemlerinin ve sınıflandırıcıların başarımlarını F1 skoru ile karşılaştırmışlardır. Bu tez çalışmasında kullanılan modeller ile Yıldırım ve diğerlerinin çalışmalarında kullandığı modellerin başarımlarının karşılaştırılması Tablo 5.1'de görülmektedir.

Tablo 5.1 TTC4900 veri seti ile oluşturulan modeller için f1 skoru

	Sınıflandırma Modelleri			
İlgili Çalışmalar	Naive Bayes	ESA	UKSB	TextGCN
[49]	90	73	75	-
Bu çalışma	91.8	90	73.7	91.56

Nihai olarak; Türkçe veri setleri kullanılarak yapılan bu tez çalışmasında, kullanılan sınıflandırıcı ve yöntemlerde metnin dili önemsizdir. Metnin dili bağlamına bakıldığında sadece gramer hatalarının olup olmaması önemlidir. Tezdeki yöntemler ile metin sınıflandırılmasında metnin dilinin önemli olduğu tek nokta metindeki kelimelerin eklerine köklerine ayrılması ve gereksiz kelimelerin çıkarılması gibi ön işleme adımlarıdır. Elde edilen başarım sonuçlarına göre modelleri iyileştirmek için daha iyi veri ön işleme ve temizleme yöntemleri kullanılabilir. Ayrıca daha büyük veri seti ve transfer öğrenme yöntemleri kullanıldığında daha iyi bir başarım elde edilebilir.

6. SONUÇ VE ÖNERİLER

Bu tez çalışmasında, Çizge Evrişimli Sinir Ağları kullanılarak metin sınıflandırma problemi ele alınmıştır. Çizge Evrişimli Sinir Ağları, evrişimin çizgelere uygulanması için geliştirilen bir yapay sinir ağı modelidir. Çizge evrişimli sinir ağlarının metin sınıflandırma alanında kullanılabilmesi için metinlerin çizge şeklinde temsil edilmesi gereklidir. Bu çalışmada GCN'nin metin sınıflandırma alanında performansını değerlendirmek için, iki adet Türkçe veri seti olan TTC4900 ve Sosyal Medya Verileri veri setleri kullanılmıştır. Bu veri setleri farklı konu ve kategorilere sahip metinler içermektedir. GCN'nin karşılaştırıldığı diğer modeller ise Naive Bayes, ESA ve UKSB'dir. Metin sınıflandırma için kullanılan modellerin eğitimi ve testi için, veri setleri eğitim, test ve doğrulama kümelerine ayrılmıştır. Eğitim seti modelin parametrelerini öğrenmesi için kullanılırken, doğrulama seti modellerin hiperparametrelerini ayarlamak ve test seti ise modellerin eğitimde kullanmadığı veriler ile başarımını ölçmek için kullanılmıştır.

Metin verileri ile modelin eğitilebilmesi için öncelikle metin verilerinin sayısal temsillere dönüştürülmesi gereklidir. Metnin sayısal bir vektör olarak ifade edilmesine vektör temsili denir. Vektör temsili yöntemleri arasında TF-IDF, Sayma Vektörleştirici (Kelime Çantası), Word2Vec, GloVe ve BERT gibi yöntemler bulunmaktadır. Bu yöntemler metin verilerinin anlamsal ve sözdizimsel özelliklerini yakalamaya çalışır. Bu tez çalışmasında, kategorik verilerin numerik temsilinde Naive Bayes sınıflandırıcı için sayma vektörleştici, ESA ve RNN modelleri için Keras vektörizer yöntemi kullanılmıştır. GCN modeli için ise TF-IDF ve PMI hesaplamasına dayalı vektör temsilleri çizgenin oluşturulmasında kullanılmıştır.

Deney sonuçlarına göre, modeller genel olarak TTC4900 veri seti için daha yüksek doğrulama başarımı göstermiştir. Ancak bu sonuçlar kullanılan veri kümelerine, seçilen özelliklere ve benimsenen yönteme bağlı olarak değişiklik gösterebilir. Aynı veri kümeleri kullanılmayıp başka veri kümeleri eğitim için kullanıldığında GCN modelinin diğer modellere göre yüksek başarım sergilediği gözlemlenmiştir. Ayrıca seçilen parametreler de bazı durumlarda bir modelin performansını etkileyebilir.

Metin tabanlı GCN modelini metin sınıflandırma alanında diğer derin öğrenme yöntemlerine göre tercih etmenin bazı avantajları vardır. Bunlar aşağıdaki gibidir:

- Metinleri çizge yapısı şeklinde temsil ederek, kelimeler arasındaki bağlantıları ve anlamsal ilişkileri iyi bir şekilde öğrenebilir.
- Metinlerin uzunluğuna veya sırasına bağlı olmadan esnek bir şekilde çalışabilir.
- Evrişimli veya tekrarlayan katmanlar yerine çizge evrişimli katmanlar kullanarak daha az parametre ile daha iyi sonuçlar elde edebilir.

GCN modelini sınıflandırma alanında kullanmanın dezavantajları ise şunlardır:

- Metinleri çizge yapısı şeklinde temsil etmek ekstra işlem gerektirir.
- Çizge yapısının oluşturulması için TF-IDF ve PMI gibi istatistiksel yöntemlere bağımlıdır.
- Çevrimiçi (online) veriye izin vermez. Yeni bir metnin eklenmesi veya silinmesi durumunda tüm çizgenin yeniden hesaplanması gerekir.
- Bellek tüketimi yüksektir, çünkü tüm belgelerden oluşan tek bir çizgeyi saklamak ve işlemek gerekir.

Sonuç olarak, bu tez çalışması metin tabanlı GCN'nin metin sınıflandırma problemine etkili bir çözüm sunabileceğini göstermiştir. Ancak metin tabanlı GCN'nin dezavantajları da bulunmaktadır. Bu nedenle metin tabanlı GCN'nin performansını artırmak ve dezavantajlarını azaltmak için gelecekteki çalışmalarda şu konulara odaklanılabilir:

- Metinleri çizge yapısı şeklinde temsil etmek için farklı yöntemler denenebilir.
- Çizge yapısının oluşturulması için TF-IDF ve PMI yerine daha gelişmiş yöntemler kullanılabilir.
- Online veriyi desteklemek için çizgenin dinamik olarak güncellenmesine imkân sağlayan yöntemler geliştirilebilir.
- Bellek tüketimini azaltmak için çizgenin alt kümelere bölünmesine veya boyutunun indirgenmesine yönelik yeni yöntemler araştırılabilir.

Ayrıca dikkat mekanizmaları ve GCN'in bir arada kullanılması ile metin sınıflandırma görevlerinde daha iyi performans sağlanabilir. Dikkat mekanizması, GCN'nin dikkatini metindeki önemli bölgelere yönlendirebilir ve GCN, bu dikkate değer bölgelerdeki ilişkileri daha iyi modelleyebilir. GCN, bu dikkate değer bölgelerdeki kelimeler veya cümleler arasındaki ilişkileri analiz ederek daha kapsamlı bir metin temsili elde edebilir ve daha doğru sınıflandırma sonuçları üretebilir.

KAYNAKLAR

[1] L. Yao, C. Mao and Y. Luo, "Graph Convolutional Networks for Text Classification," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 33, no. 01, pp. 7370-7377, 2019.

[2] Ş. E. Şeker, "Metin Madenciliği (Text Mining)," *YBS Ansiklopedi,* vol. 2, no. 3, 2015.

[3] G. K. T. Muppala, "Recipe Text Classification using Graph Neural Networks," Stanford University, Stanford , 2019.

[4] A. Khan, B. Baharudin, L. H. Lee and K. Khan, "A review of machine learning algorithms for text-documents classification," *Journal of advances in information technology,* vol. 1, no. 1, pp. 4-20, 2010.

[5] S. Çelik, "Metin Madenciliği ile Shakespeare Külliyatının İncelenmesi," *MANAS Sosyal Araştırmalar Dergisi,* vol. 9, no. 3, pp. 1343-1357, 2020.

[6] S. Baghbani, "Text Classification: process and Algorithms," in *3rd. International Conference on Research in Engineering, Science and Technology*, Batumi,Georgia, 2016.

[7] A. Khan, B. B. Bahurdin and K. Khan, "An Overview of E-Documents Classification," in *2009 International Conference on Machine Learning and Computing*, Singapore, 2009.

[8] H. Wu, Y. Liu and J. Wang, "Review of Text Classification Methods on Deep Learning," *Computers, Materials & Continua (CMC),* vol. 63, no. 3, pp. 1309-1321, 2020.

[9] H. Peng, J. Li, Y. He, Y. Liu, B. Mengjiao, L. Wang, Y. Song and Q. Yang, "Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN," in *WWW '18: Proceedings of the 2018 World Wide Web Conference*, 2018.

[10] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys,* vol. 34, no. 1, pp. 1-47, 2002.

[11] D. Delen and M. D. Crossland, "Seeding the survey and analysis of research literature with text mining," *Expert Systems with Applications,* vol. 34, no. 3, pp. 1707-1720, 2008.

[12] A. Ahadi , A. Singh, M. Bower and M. Garrett, "Text Mining in Education—A Bibliometrics-Based Systematic Review," *Education Sciences,* vol. 12, no. 3, p. 210, 2022.

[13] S. Kumar, A. K. Kar and P. V. Ilavarasan, "Applications of text mining in services management: A systematic literature review," *International Journal of Information Management Data Insights,* vol. 1, no. 1, p. 100008, 2021.

[14] H. Zhu and L. Lei, "The Research Trends of Text Classification Studies (2000–2020): A Bibliometric Analysis," *SAGE Open,* vol. 12, no. 2, p. 21582440221089963, 2022.

[15] M. M. Mirończuk and . J. Protasiewicz, "A recent overview of the state-of-the-art elements of text classification," *Expert Systems with Applications,* vol. 106, no. 1, pp. 36-54, 2018.

[16] M. Thangaraj and M. Sivakami, "Text Classification Techniques: A Literature Review," *Interdisciplinary Journal of Information, Knowledge, and Management,* vol. 13, pp. 117-135, 2018.

[17] D. Kilinç, A. Ozcift, F. Bozyiğit, P. Yildirim Taser, F. Yucalar and E. Borandağ, "TTC-3600: A new benchmark dataset for Turkish text categorization," *Journal of Information Science,* vol. 43, pp. 174-185, 2017.

[18] D. Kavi, "Turkish Text Classification: From Lexicon Analysis to Bidirectional Transformer," *arXiv preprint arXiv:2104.11642,* 2020.

[19] M. F. Amasyali and B. Diri, "Automatic Turkish Text Categorization in terms of Author, Genre and Gender," in *Natural Language Processing and Information Systems*, Berlin, Heidelberg, Springer Berlin Heidelberg, 2006, pp. 221-226.

[20] F. Turkoglu, B. Diri and M. F. Amasyali, "Author Attribution of Turkish Texts by Feature Mining," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, Berlin, Heidelberg, Springer Berlin Heidelberg, 2007, pp. 1086-1093.

[21] Z. Çataltepe, Y. Turan and F. Kesgin, "Turkish document classification using shorter roots," in *IEEE 15th Signal Processing and Communications Applications Conference*, Eskişehir, Turkey, 2007.

[22] V. E. Levent and B. Diri, "Türkçe Dokümanlarda Yapay SinirAğları ile Yazar Tanıma," in *Akademik Bilişim'14*, Mersin, 2014.

[23] B. Açıkalın and N. Güler Bayazıt, "The importance of preprocessing in Turkish Text classification," in *24th Signal Processing and Communication Application Conference (SIU)*, Zonguldak, Turkey, 2016.

[24] D. Ayata, M. Saraçlar and A. Özgür, "Turkish tweet sentiment analysis with word embedding and machine learning," in *2017 25th Signal Processing and Communications Applications Conference (SIU)*, 2017, pp. 1-4.

[25] C. Wong, "Analyzing Easy Data Augmentation Techniques for Text Classification," Harvard, 2021.

[26] Z. S. Harris, "Distributional Structure," *WORD,* vol. 10, no. 2-3, pp. 146-162, 1954.

[27] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *Proceedings of Workshop at ICLR,* vol. 2013, 2013.

[28] J. Pennington, R. Socher and C. D. Manning, "The Stanford Natural Language Processing Group," [Online]. Available: https://nlp.stanford.edu/projects/glove/. [Accessed 11 Nisan 2023].

[29] M. M. ARAT, "Github," 20 Mart 2020. [Online]. Available: https://mmuratarat.github.io/2020-03-20/glove. [Accessed 11 Nisan 2023].

[30] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *CoRR,* vol. abs/1810.04805, 2018.

[31] S. Chua, F. Coenen and G. Malcolm, "Classification Inductive Rule Learning with Negated Features," in *Advanced Data Mining and Applications*, Berlin, Heidelberg, 2010.

[32] M. Thangaraj and M. Sivakami, "Text Classification Techniques: A Literature Review," *Interdisciplinary Journal of Information, Knowledge, and Management,* vol. 13, pp. 117-135, 2018.

[33] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu and L. He, "A Survey on Text Classification: From Traditional to Deep Learning," *ACM Transactions on Intelligent Systems and Technology,* vol. 13, no. 2, pp. 1-41, 2022.

[34] P. Potrimba, "What is Semi-Supervised Learning? A Guide for Beginners," Aralık 2022. [Online]. Available: https://blog.roboflow.com/what-is-semi-supervised-learning/. [Accessed Mayıs 2023].

[35] T. N. Rubin, A. Chambers, P. Smyth and M. Steyvers, "Statistical topic models for multi-label document classification," *Machine Learning,* vol. 88, no. 1, p. 157–208, 2012.

[36] G. Shobha and S. Rangaswamy, "Chapter 8 - Machine Learning," in *Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications*, Elsevier, 2018, pp. 197-228.

[37] D. C. Cireşan, M. Ueli, J. Masci, L. M. Gambardella and J. A. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Twenty-Second international joint conference on Artificial Intelligence*, 2011.

[38] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[39] C. J. Maddison, A. Huang, I. Sutskever and D. Silver, "Move Evaluation in Go Using Deep Convolutional Neural Networks," in *ICLR 2015*, 2015.

[40] [Online]. Available: https://en.wikipedia.org/wiki/Convolutional_neural_network. [Accessed Mayıs 2023].

[41] "Computer Science Bytes," Mayıs 2023. [Online]. Available: http://www.computersciencebytes.com/array-variables/graphs/.

[42] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Processing Magazine,* vol. 34, no. 4, pp. 18-42, 2017.

[43] M. L. Özbilen, "Hisse Senedi Fiyat Tahmininde Otokodlayıcı ve Graf Evrişimli Ağının Uygulanması," İstanbul Teknik Üniversitesi-Lisansüstü Eğitim Enstitüsü, 2022.

[44] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *CoRR,* vol. abs/1609.02907, 2016.

[45] L. Huang, D. Ma, S. Li, X. Zhang and H. Wang, "Text Level Graph Neural Network for Text Classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019.

[46] "text4gcn 1.0.0," Ağustos 2022. [Online]. Available: https://pypi.org/project/text4gcn/. [Accessed Mayıs 2023].

[47] L. Huang, D. Ma, S. Li, X. Zhang and H. Wang, "Text Level Graph Neural Network for Text Classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019.

[48] Y. Lin, Y. Meng, X. Sun, Q. Han, K. Kuang, J. Li and F. Wu, "BertGCN: Transductive Text Classification by Combining GCN and BERT," 2022. [Online]. Available: arXiv preprint arXiv:2105.05727.

[49] S. Yıldırım and T. Yıldız, "A Comparison of Different Approaches to Document Representation in Turkish Language," *SDU Fen Bilimleri Enstitüsü Dergisi,* vol. 22, no. 2, pp. 569-576, 2018.

[50] "Kemik Doğal Dil İşleme Grubu," Yıldız Teknik Üniversitesi, [Online]. Available: http://www.kemik.yildiz.edu.tr/. [Accessed Mayıs 2023].

[51] "Türkçe Sosyal Medya Paylaşımı Veri Seti," Kaggle, [Online]. Available: https://www.kaggle.com/datasets/mrtbeyz/trke-sosyal-medya-paylam-veri-seti.

[52] A. A. Akın, "Zemberek-nlp," github, [Online]. Available: https://github.com/ahmetaa/zemberek-nlp. [Accessed Mayıs 2023].

[53] O. Doguc, "Lemmatizer: Akıllı Türkçe Kök Bulma Yöntemi," *Turkish Studies-Information Technologies and Applied Sciences,* vol. 15, no. 3, pp. 289-299, 2020.

[54] A. Aksoy. [Online]. Available: https://github.com/ahmetax/kalbur.

[55] A. A. A. and A. M. D., "Zemberek, an open source nlp framework for turkic languages,," pp. 1-5, 2007.

[56] "zeyrek 0.1.3," [Online]. Available: https://pypi.org/project/zeyrek/. [Accessed Mayıs 2023].

[57] M. F. Porter, "Snowball: A language for stemming algorithms," 2001.

[58] shristikotaiah, "geeksforgeeks," Mayıs 2023. [Online]. Available: https://www.geeksforgeeks.org/snowball-stemmer-nlp/.

[59] M. Tripathi, "freeCodeCamp," [Online]. Available: https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3. [Accessed Nisan 2023].

[60] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Naive_Bayes_classifier. [Accessed 2023].

[61] D. Sanghi, "geeksforgeeks," [Online]. Available: https://www.geeksforgeeks.org/applying-multinomial-naive-bayes-to-nlp-problems/. [Accessed 2023].

[62] V. Nguyen, N. Anh and H.-J. Yang, "Real-time event detection using recurrent neural network in social sensors," *International Journal of Distributed Sensor Networks,* vol. 15, 2019.

[63] A. Kumar , "Data Analytics," Kasım 2021. [Online]. Available: https://vitalflux.com/real-world-applications-of-convolutional-neural-networks/. [Accessed Mayıs 2023].

[64] A. Lopez-del Rio, M. Martin, A. Perera-Lluna and R. Saidi, "Effect of sequence padding on the performance of deep learning models in archaeal protein functional prediction," *Scientific Reports,* vol. 10, no. 1, p. 14634, 2020.

[65] T. Joseph, "Introducing Recurrent Neural Networks," Eylül 2020. [Online]. Available: https://towardsdatascience.com/introducing-recurrent-neural-networks-f359653d7020. [Accessed Mayıs 2023].

[66] B. Ay Karakuş, "Derin Öğrenme ve Büyük Veri Yaklaşımlar ile Metin Analizi," Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ, 2018.

[67] C. Jupyter, "Google Colab," [Online]. Available: https://colab.research.google.com/github/goodboychan/goodboychan.github.io/

blob/main/_notebooks/2020-12-06-01-RNN-Many-to-one.ipynb. [Accessed Mayıs 2023].

[68] J. Dancker, "A Brief Introduction to Recurrent Neural Networks," Aralık 2022. [Online]. Available: https://towardsdatascience.com/a-brief-introduction-to-recurrent-neural-networks-638f64a61ff4.

[69] W. Hamilton, Z. Ying and J. Leskovec, " Inductive Representation Learning on Large Graphs," *Advances in Neural Information Processing Systems,* vol. 30, pp. 1024-1034, 2017.

[70] D. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik and R. Adams, "Convolutional networks on graphs for learning molecular fingerprints," *Advances in Neural Information Processing Systems,* p. 28, 2015.

[71] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li and M. Sun, "Graph Neural Networks: A Review of Methods and Applications," arXiv, 2021.

[72] S. Mallat, A wavelet tour of signal processing, Elsevier, 1999.

[73] J. Bruna, W. Zaremba, A. Szlam and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *International conference on learning representations (ICLR)*, 2014.

[74] M. Defferrard, X. Bresson and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[75] Ö. İ. Sever, "Mesh Segmentation from Sparse Face labels using Graph Convolutional Neural Networks," Middle East Technical University (Yok Ulusal Tez Merkezi), Ankara, 2020.

[76] "Evrişimsel Çizit Ağları (Graph Convolutional Networks)," [Online]. Available: https://burakbayramli.github.io/dersblog/algs/algs_185_graphconv/evrisimsel_cizit_aglari__graph_convolutional_networks_.html. [Accessed Mayıs 2023].

[77] SaffronEdge, "Feedforward vs Backpropagation ANN," Şubat 2023. [Online]. Available: https://www.linkedin.com/pulse/feedforward-vs-backpropagation-ann-saffronedge1/. [Accessed Mayıs 2023].

[78] A. C. ÇINAR, "Kısıtlı ve Ayrık Optimizasyon Problemlerinin Çözümü için Ağaç-Tohum Algoritmasının Uyarlanması ve Analizi (Doktora Tezi)," Konya Teknik Üniversitesi, Konya, 2020.

[79] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015*, San Diego, CA, USA, 2015.

[80] "Epoch in Machine Learning," [Online]. Available: https://www.javatpoint.com/epoch-in-machine-learning. [Accessed Mayıs 2023].

[81] F. Zeng, N. Chen, D. Yang and Z. Meng, "Simplified-Boosting Ensemble Convolutional Network for Text Classification," *Neural Processing Letters,* vol. 54, no. 6, pp. 4971-4986, 2022.

[82] J. Brownlee, "Techniques to Handle Very Long Sequences with LSTMs," 2017. [Online]. Available: https://machinelearningmastery.com/handle-long-sequences-long-short-term-memory-recurrent-neural-networks/. [Accessed Mayıs 2023].

[83] T. Kipf, "Graph Convolutional Networks," Eylül 2016. [Online]. Available: https://tkipf.github.io/graph-convolutional-networks/. [Accessed Mayıs 2023].

[84] Q. Xu and Z. Liu, "Automatic Chinese text classification based on NSVMDT-KNN," *2008 IEEE International Conference on Fuzzy Systems & Knowledge Discovery,* vol. 2, pp. 410-414, 2008.

[85] S. K. Sah Tyagi, V. Dogra, S. Verma, Kavita, P. Chatterjee, J. Shafi, J. Choi and M. F. Ijaz, "A Complete Process of Text Classification System Using State-of-the-Art NLP Models," *Computational Intelligence and Neuroscience,* vol. 2022, p. 26, 2022.

[86] D. C. Cireşan, M. Ueli, J. Masci, L. M. Gambardella and J. A. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Twenty-Second international joint conference on Artificial Intelligence*, 2011.

[87] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[88] C. J. Maddison, A. Huang, I. Sutskever and D. Silver, "Move Evaluation in Go Using Deep Convolutional Neural Networks," in *ICLR 2015*, 2015.

EKLER

EK 1. Çalışma alanlarına ilişkin n-gram örnekleri

Çalışma Alanı	Birim	Örnek dizi	1-Gram	2-Gram	3-Gram
Protein dizilimi	Amino Asit	…Cys-Gly-Leu-Ser-Trp…	..., Cys, Gly, Leu, Ser, Trp, ...	..., Cys-Gly, Gly-Leu, Leu-Ser, Ser-Trp, ...	..., Cys-Gly-Leu, Gly-Leu-Ser, Leu-Ser-
DNA dizilimi	Baz Çifti	...AGCTTCGA...	..., A, G, C, T, T, C, G, A, ...	.., AG, GC, CT, TT, TC, CG, GA, ...	..., AGC, GCT, CTT, TTC, TCG, CGA, ...
Hesaplamalı Dil Bilimleri	Karakter	…olmak_ya_da_olmamak ...	…, o, l, m, k,_, y, a, _, d, a, _, o, l, m, a, m, a, k, ...	…, ol, lm, ma, ak, _y, ya, _d, da, o, ol, lm, ma, ma, ak, …	…,olm, lma, mak, ak, k_y, ya, ya, a_d, da, da, a_o, _ol, olm, lma, mam, ama, mak,…
Hesaplamalı Dil Bilimleri	Kelime	…olmak ya da olmamak …	…, olmak, ya, da, olmamak, …	…, olmak ya, ya da, da olmamak, …	…, olmak ya da, ya da olmamak, …