T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

KONVOLÜSYONEL SİNİR AĞLARI KULLANARAK TÜRKÇE METİNLER için CÜMLE SINIFLANDIRMASI

Gurur PİRANA

DANIŞMAN
Prof. Dr. Ahmet SERTBAŞ

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

İSTANBUL-2020

Uygundur

Prof. Dr. Ahmet SERTBAŞ
Bilgisayar Mühendisliği
Bölüm Başkanı

Bu çalışma 25.02.2020 Tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı, Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı Yüksek Lisans Tezi olarak kabul edilmiştir.

TEZ JÜRİSİ

Prof. Dr. Ahmet SERTBAŞ
İstanbul Üniversitesi-Cerrahpaşa
Mühendislik Fakültesi

Dr.Öğr.Üyesi Oğuzhan ÖZTAŞ
İstanbul Üniversitesi-Cerrahpaşa
Mühendislik Fakültesi

Doç. Dr. Akhan AKBULUT
İstanbul Kültür Üniversitesi
Mühendislik Fakültesi

20.04.2016 tarihli Resmi Gazete'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, İstanbul Üniversitesi-Cerrahpaşa'nın abonesi olduğu intihal yazılım programı kullanılarak Lisansüstü Eğitim Enstitüsü'nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır.

ÖNSÖZ

Yüksek lisans eğitimim boyunca desteklerini eksik etmeyen, bilgi ve birikimleri ile bana yol gösteren çok değerli danışman hocalarım Prof. Dr. Ahmet SERTBAŞ ve Dr. Öğr. Üyesi Tolga ENSARİ' ye teşekkürlerimi sunarım.

Araştırmalarım boyunca bana her zaman destek olan ve rehberlik eden Sayın Ahmet BAHAR' a ve Minör Yazılım ekibine teşekkürlerimi sunarım.

Beni bugünlere getirmek için maddi manevi desteklerini hiçbir zaman esirgemeyen, bu dünyadaki gurur kaynağım olan aileme sonsuz teşekkür ederim.

Şubat 2020 Gurur PİRANA

İÇİNDEKİLER

Sayfa No

ÖNSÖZ ..... **iv**
İÇİNDEKİLER ..... **v**
ŞEKİL LİSTESİ ..... **vii**
TABLO LİSTESİ ..... **ix**
SİMGE VE KISALTMA LİSTESİ ..... **x**
ÖZET ..... **xi**
SUMMARY ..... **xiii**
1. GİRİŞ **1**
2. GENEL KISIMLAR **2**
2.1. MAKİNE ÖĞRENMESİ YÖNTEMLERİ 2
2.1.1. Naive Bayes Multinomial 4
2.1.2. Destek Vektör Makineleri 5
2.1.3. Lojistik Regresyon 8
2.2. DERİN ÖĞRENME YÖNTEMLERİ 10
2.2.1. Kelime/Sözcük çantası (Bow) 10
2.2.2. Uzun-Kısa Süreli Hafıza (LSTM) 12
2.2.3. Konvolüsyonel Sinir Ağları (CNN) 15
2.2.3.1. Konvolüsyon (Evrişim) Katmanı *16*
2.2.3.2. Havuzlama (Pooling) Katmanı *20*
2.2.3.3. Tam Bağlantı Katmanı (Fully Connected Layer) *21*
2.2.4. Bölgesel Konvolüsyonel Sinir Ağları (R-CNN) 22
3. MALZEME VE YÖNTEM **24**
3.1. PROGRAMLAMA DİLİ VE DONANIM 24
3.2. VERİ SETİNİN HAZIRLANMASI 25
3.3. KELİMELERİN VEKTÖR UZAYINDA İFADE EDİLMESİ (WORD2VEC) 26
3.3.1. Skip-Gram 26
3.3.2. Continous Bag of words (CBOW) 28
4. BULGULAR **30**

5. TARTIŞMA VE SONUÇ 36
KAYNAKLAR.....38
ÖZGEÇMİŞ 42

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1. Denetimli makine öğrenmesi örnek etiketlenmesi [8]…….….3

Şekil 2.2. Denetimli makine öğrenmesi modeli [8]…….....3

Şekil 2.3. Denetimsiz makine öğrenmesi modeli [8]…….…...4

Şekil 2.4. SVM veri kümesi ayrımı[17]……...6

Şekil 2.5. SVM kullanılarak en uygun çizgi belirlenmesi[17]……7

Şekil 2.6. Lojistik regresyon grafiği…….9

Şekil 2.7. Standart RNN mimarisi[23]……12

Şekil 2.8. Uzun-Kısa Süreli Hafıza mimarisi……..13

Şekil 2.9. Uzun-Kısa Süreli Hafıza Hücre Yapısı.....13

Şekil 2.10. Standart LSTM modeli[24]……...14

Şekil 2.11. Standart LSTM model formülleri[24]……...14

Şekil 2.12. Konvolusyonel Sinir Ağları Mimarisi[37]……15

Şekil 2.13. CNN LeNet Mimarisi[26]…….16

Şekil 2.14. Konvolüsyon katmanı girdi ve filtre matrisleri örneği[28]……...16

Şekil 2.15. Konvolüsyon katmanındaki matris çarpımının birinci adımı[28]…….17

Şekil 2.16. Konvolüsyon katmanındaki matris çarpımının son adımı[28]……..17

Şekil 2.17. Maksimum havuzlama işlemi[30]…….20

Şekil 2.18. Ortalama havuzlama işlemi[30]……21

Şekil 2.19. Düzleştirme (Flattening) işlemi.....21

Şekil 2.20. RCNN mimarisi [31]……22

Şekil 3.1. Skip-Gram model örneği [33].……27

Şekil 3.2. Skip-Gram modeli [34]……...28

Şekil 3.3. Continous Bag of Words (CBOW) model örneği [33]…….. 29

Şekil 3.4. Continous Bag of Words (CBOW) modeli [34].....29

Şekil 4.1. I. Veri seti kullanılarak eğitim doğruluk(accuracy) ve kayıp(loss) çıktısı.....31

Şekil 4.2. II. Veri seti kullanılarak eğitim doğruluk(accuracy) ve kayıp(loss) çıktısı.....32

Şekil 4.3. Flask kütüphanesi ile model test edilmesi……...32

Şekil 4.4. II numaralı veri seti ile oluşturulan modelin flask ile test edilmesi…….33

Şekil 4.5. Farklı epoch değerlerinin model üzerindeki etkisi……..33

Şekil 4.6. Zaman aralıklı model başarı grafiği……35

TABLO LİSTESİ

Sayfa No

Tablo 2.1. Popüler olarak kullanılan aktivasyon fonksiyonları[29].....19

Tablo 3.1. Model eğitiminde kullanılan veri kümesi……25

Tablo 4.1. Sınıflandırma modellerinin karşılaştırılması……30

Tablo 4.2. Zaman aralıklı model başarı tablosu……34

SİMGE VE KISALTMA LİSTESİ

Simgeler **Açıklama**

ms : Milisaniye

sn : Saniye

dk : Dakika

Kısaltmalar **Açıklama**

CNN : Convolutional Neural Network (Konvolüsyonel Yapay Sinir Ağları)

LSTM : Long Short Term Memory (Uzun Kısa Vadeli Hafıza Ağları)

RNN : Recurrent Neural Network (Tekrarlayan Yapay Sinir Ağları)

SVM : Support Vector Machine (Destek Vektör Makineleri)

MNB : Multinomial Naive Bayes

MLR : Multinomial Logistic Regression

RCNN : Region Convolutional Neural Network (Bölgesel Konvulüsyonel Sinir Ağı)

BOW : Kelime/Sözcük Çantası (Bag of Word)

CBOW **:** Devamlı Kelime/Sözcük Çantası **(**Continous Bag of Words)

TSA **:** Tekrarlayan Sinir Ağları

FC **:** Fully Connected (Tam Bağlı)

ÖZET

KONVOLUSYONEL SİNİR AĞLARI KULLANARAK TÜRKÇE METİNLER için CÜMLE SINIFLANDIRMASI

YÜKSEK LİSANS TEZİ

Gurur PİRANA

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Prof. Dr. Ahmet SERTBAŞ

Bu çalışmada derin öğrenme ve makine öğrenmesi yöntemleri kullanılarak cümle sınıflandırma problemi ele alınmıştır. Çalışmanın amacı, Konvolüsyonel Sinir Ağları (CNN-Convolutional Neural Networks), Bölge Konvolüsyonel Sinir Ağları (RCNN Region Convolutional Neural Networks),Uzun-Kısa Süreli Hafıza (LSTM-Long Short Term Memory) , Naive Bayes Multinomial , Lojistik Regresyon , Destek Vektör Makineleri gibi farklı derin öğrenme ve makine öğrenmesi metotlarının veri kümesi üzerinde başarımlarını incelemektir. Bu yöntemler ile elde edilen modellerin eğitim parametrelerinin değişiminin başarıma olan etkisi araştırılmıştır. Her bir model için başarımı en yüksek olan parametreler tespit edilmiş ve kullanılmıştır. Bu çalışmanın amacı, cümle sınıflandırılması için geliştirilen modelin, giriş verisini, yani cümleyi uygun olan sınıf ile eşleştirilip bu sınıfın karşılığındaki cevabın üretilmesidir. Çıktı değerleri değişken olup metin, dosya, resim veya bir url çıktısı olabilir. Girdi cümlesinin karşılığında çıktı aracılığıyla işlem yönlendirmesi de

yapılabilmektedir. Bu işlem sanal asistan gibi uygulamaların geliştirilmesinde ve kullanılmasında kolaylık sağlamaktadır. Gözlemlediğimiz önemli bir konu da veri kümesinde bulunan örnek sayısının model üzerindeki başarım etkisidir. Örnek sayısının artışı model başarımının artışını doğrudan etkilemektedir. Aynı zamanda derin öğrenme ve makine öğrenmesi yöntemlerinin eğitim sürelerinin arasındaki fark gözlemlenmiş ve aralarında karşılaştırma yapılmıştır. Sonuç olarak, oluşturduğumuz Türkçe veri kümesi ile en iyi başarımı Konvolüsyonel Sinir Ağları (CNN)' yöntemi ile elde etmiş bulunmaktayız.

Şubat 2020, 56 . sayfa.

Anahtar kelimeler: Derin öğrenme , Makine öğrenmesi , CNN, LSTM , Naive Bayes , SVM

SUMMARY

SENTENCE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS FOR TURKISH TEXT

M.Sc. THESIS

Gurur PİRANA

Istanbul University-Cerrahpasa

Institute of Graduate Studies

Department of Computer Engineering

Supervisor : Prof. Dr. Ahmet SERTBAŞ

This paper investigates deep learning and machine learning method performance for virtual assistant applications about sentence classification. The classification is based in Turkish texts. For different methods we demonstrate the performance of each model. We investigate Convolutional Neural Network (CNN), Region Convolutional Neural Network (RCNN), Long Short Term Memory (LSTM), Naïve Bayes Multinomial, Logistic Regression, Support Vector Machine (SVM) deep and machine learning methods and compare the accuracy results of the related models. Furthermore, we aim to select the best classification model for our dataset. We have researched effect of the parameters to model accuracy and we used model parameters for each methods and we aimed to gain best performance for our dataset. This research helps applications like virtual assistant with classification of the sentence and giving the output of the class. The output of classification could be a text, document, image or

url. Benefit of this comparison of the methods we realized that instance number increases the model accuracy. The best method for our dataset was the Convolutional Neural Networks (CNN).

February 2020, 56 . pages.

Keywords: Deep Learning, Machine Learning, CNN, LSTM, Naive Bayes, SVM

1. GİRİŞ

Son yıllarda, derin öğrenmeye büyük bir ilgi duyulmaktadır [1]. Cümle sınıflandırma, yapay sinir ağlarının önemli dallarından biridir. Konvolüsyonel Sinir Ağları, ilk olarak LeCun tarafından ortaya atılmış [2], daha sonrasında ise geliştirilerek yaygınlaşmıştır [3]. Konvolüsyonel sinir ağlarında gözlemlenen sonuçlar, derinliğin iyi performans için kritik bir faktör olduğunu göstermektedir [4]. Farklı yaklaşımlar ile metin tabanlı sınıflandırma yapılabilmektedir. Bunlardan bazılarına örnek vermek gerekirse, Destek Vektör Makinesi (SVM) [5], Naive Bayes ile Destek Vektör Makinesinin birleştirilmesi [6], Tekrarlayan Sinir ve Konvolüsyonel Sinir Ağlarının birleştirilmesi [7], Tekrarlayan Sinir Ağları ile metin sınıflandırma [8], Uzun Kısa Süreli Bellek Kullanarak Metin Kategorilendirme [9] şeklinde sıralanabilir.

Bu çalışmalar dışında, CNN'i farklı açılardan geliştirmek için örneğin katman tasarımı, etkinleştirme fonksiyonu, kayıp fonksiyonu, düzenli hale getirme, optimizasyon ve hızlı hesaplama gibi literatürde birçok çalışma bulunmaktadır [4]. Konvolüsyonel Sinir Ağları, ileri beslemeli sinir ağlarıdır. CNN Mimarisinde konvolüsyonel katmanlar bulunmaktadır. Bu konvolüsyonel katmanlar, havuz katmanları (pool layers) ile birleştirilir. CNN yöntemi genellikle görüntü işleme uygulamalarında yaygın olarak kullanılmaktadır [10]. Aynı zamanda metin sınıflandırmada işleminde başarılı olduğunu gösteren araştırmalar mevcuttur [11].

Açık kaynak kodlu ekosistem olarak Google tarafından geliştirilen TensorFlow kütüphanesi, araştırmacıların çalışmalarında büyük ölçüde avantajları bulunmaktadır. Kütüphanenin en büyük katkısı çalışmaların geliştirilmesinde harcanan sürenin kısalmasıdır. Bu çalışmanın amacı, cümle sınıflandırılması için geliştirilen modelin, giriş verisini, yani cümleyi uygun olan sınıf ile eşleştirilip bu sınıfın karşılığındaki cevabın üretilmesidir. Çalışmamızda Türkçe cümlelerden oluşan veri kümesi kullanılmaktadır. Veri kümesi hazırlanırken yardımcı araçlar kullanılmaktadır. Veri setinin model eğitiminden önce verilerin ön işlemden geçirilmesi ile iyileştirmeler yapılmaktadır.

2. GENEL KISIMLAR

Metin tabanlı sınıflandırma için kullanılan farklı yöntemler bulunmaktadır. Bu yöntemlerin literatürdeki benzer çalışmaları incelenmiş ve sonrasında oluşturulmuş Türkçe veri seti üzerinde bu yöntemlerin başarımı incelenmiştir. Bu bölümde incelenmiş olan ve karşılaştırma çalışmamızda kullanılan derin öğrenme ve makine öğrenmesi yöntemleri hakkında detaylı bilgi verilmiştir.

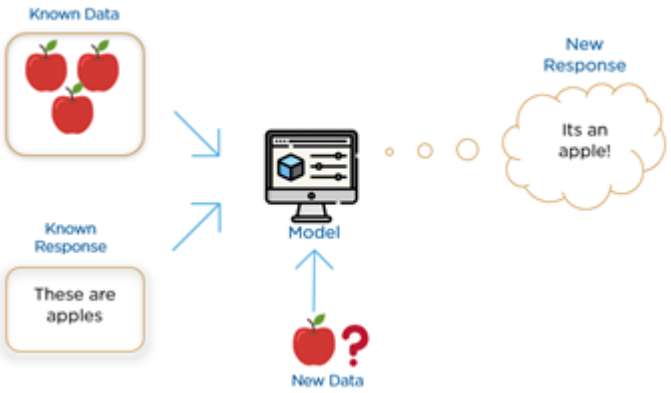
2.1. MAKİNE ÖĞRENMESİ YÖNTEMLERİ

Makine öğrenmesi sistemlerin açıkça programlanmadan, otomatik öğrenme ve öğrenilen deneyimlerin geliştirilme yeteneği sağlayan yapay zekânın uygulamasıdır. Öğrenme süreci veri setinde bulunan örneklere dayanarak gelecekte daha iyi kararlar almayı hedeflemektedir. Makine öğrenmesinin en temel amacı bilgisayarın insan müdahalesi veya yardımı olmadan otomatik olarak öğrenmesini sağlamak ve eylemleri buna göre ayarlamaktır. Makine öğrenmesi algoritmaları genellikle denetimli ve denetimsiz olarak kategorize edilmektedir. Fakat makine öğrenmesi algoritmalarını farklı kategorize eden kaynaklar literatürde mevcuttur [12].

Denetimli makine öğrenmesi algoritmaları geçmişten gelen verilerden örnek alınarak gelecekteki olayları tahmin edilmesinde kullanılmaktadır [12]. Eğitim veri setinin analizinden başlayarak, öğrenme algoritması çıktı değerleri hakkında tahmin yapmak için çıkarım işlevi üretmektedir. Yeterli eğitimden sonra sistem herhangi bir yeni girdi için hedefler sağlayabilmektedir. Öğrenme algoritması çıktısını doğru ve amaçlanan çıktı ile karşılaştırabilir ve modeli buna göre değiştirmek için hataları tespit edebilmektedir.

Şekil 2.1: Denetimli makine öğrenmesi örnek etiketlenmesi. [13]

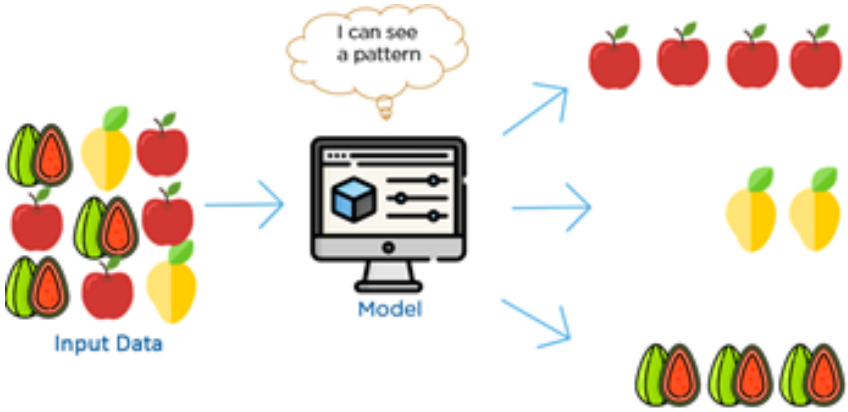
Şekil 2.1' de görüldüğü gibi denetimli makine öğrenmesi algoritmasında eğitim için oluşturulan veriler ön işlemden geçirilerek etiketlenmektedir [13]. Etiketlenmiş veri seti makine öğrenmesini tamamladıktan sonra girdi olarak alınan verinin etiketlenmiş kategoriler arasında tahmin edilmesini gerçekleştirmektedir.



Şekil 2.2: Denetimli makine öğrenmesi modeli[13].

Şekil 2.2'de görüldüğü gibi etiketlenmiş veri setinin eğitiminden sonra model üzerinden yeni veri girişi yapıldığında girilen verinin öğrenilen modelin üzerinde eşleştirileceği en uygun kategori seçilmektedir [13].

Denetimsiz makine öğrenmesi algoritmaları ise kullanılan bilgiler sınıflandırılmadığında veya etiketlenmediğinde kullanılmaktadır [12]. Denetimsiz öğrenme, sistemlerin etiketlenmemiş verilerden gizli bir yapı tanımlayan ve işlemlerin ortaya çıkmasını araştıran yapıdır. Sistem verileri araştırmakta ve etiketlenmemiş veriden gizli yapıları tanımlamak için veri kümelerinden çıkarımlar yapmaktadır.



Şekil 2.3: Denetimsiz makine öğrenmesi modeli[13].

Şekil 2.3 ‘ te görüldüğü üzere denetimsiz makine öğrenmesi algoritmasında veri setinde herhangi bir veri etiketlenmesi söz konusu değildir. Denetimsiz öğrenme algoritmasında girdi üzerindeki verilerin özellikleri incelenerek verilerin gruplanması, kümelenmesi ve/veya organize edilmesi gerçekleştirilir [13]. Bu algoritma sayesinde karmaşık büyük veriler üzerinde saklı olan bilgiler elde edilmektedir.

2.1.1. Naive Bayes Multinomial

Naive Bayes algoritması verilen bir veri seti üzerinde değerlerin sıklığını ve kombinasyonlarını sayarak olasılık kümesini hesaplamaktadır. Naive Bayes algoritması olasılık sınıflandırıcı bir algoritmadır. Algoritma Bayes Teoremine ve toplam olasılık teoremine dayanmaktadır.

Koşullu olasılıklara dayanarak bir değerin gerçekleşme ihtimalini araştırmaktadır. Bunun sonucunda hedef sınıfın değerini tahmin etmektedir. Bu işlemler denklem 2.1’deki Bayes formülü kullanılarak hesaplanmaktadır.

$$P(c \mid x) = \frac{P(c \mid x)\, P(c)}{P(x)} \tag{2.1}$$

Formül 2.1’deki değişkenleri kısaca tanıyacak olursak:

C: Tahmin edilmeye çalışılan sınıf.

X: Tahmin edilen sınıf.

P(C | X): X ‘ olayı gerçekleştiği zaman C’ olayının gerçekleşme olasılığı.

P(X | C): C’ olayı gerçekleştiği zaman X’ olayının gerçekleşme olasılığı.

P(X): X’ olayının gerçekleşme olasılığı.

P(C): C’ olayının gerçekleşme olasılığı.

Naive Bayes algoritması sonuç olarak hedef sınıfının hangi değerlerinin gerçekleşme olasılığı oranını bildirmektedir.

MNB algoritması ise genellikle metin tabanlı verilerin sınıflandırılmasında kullanılmaktadır. Belgelerdeki kelime dağılımlarının parametrik model üzerinden üretildiği ve parametrelerin eğitim verileri üzerinden tahmin edilebileceğini varsaymaktadır.

$$P(c|d) = \frac{P(c)\prod_{i=1}^{n} P(w_i|c)^{f_i}}{P(d)} \quad (2.2)[14]$$

Metin sınıflandırmada yaygın olarak denklem 2.2’deki parametrik formül kullanılmaktadır[14]. Denklem 2.2’deki formülde kullanılan değişkenleri kısaca tanıyacak olursak:

Fi: Bir belgenin d belgesinde Wi kelimesinde oluşum sayısıdır.

P(Wi | C): C kelimesinin C sınıfı göz önünde bulundurulduğunda, d kelimesinde C ifadesinin bulunabileceği koşullu olasılıktır.

N: d dosyası üzerindeki benzersiz kelime sayısıdır.

P(C): C sınıfı etiketli bir belgenin ortaya çıkma olasılığıdır.

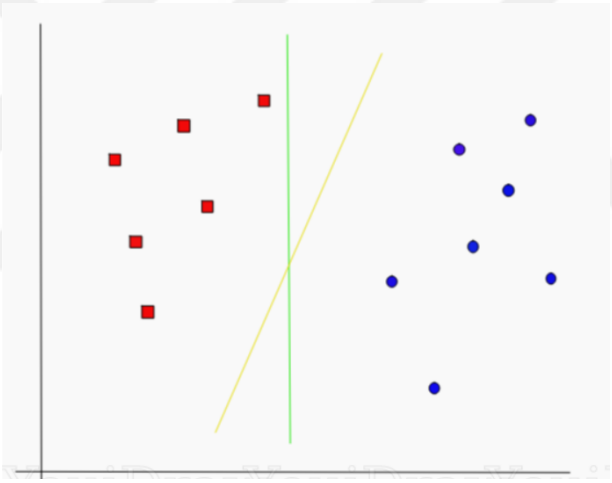
2.1.2. Destek Vektör Makineleri

SVM algoritmasın yaklaşımı iki sınıfın verileri arasında bir ayırma çizgisi veya hiper düzlem bulmaktır. Bu algoritma verileri girdi olarak alır ve mümkünse bu verilerden sınıfları ayıran bir çizgi çıkarmaya çalışan bir algoritmadır. Başka bir deyişle, etiketli eğitim verileri yani denetimli öğrenme verildiğinde algoritma yeni örnekleri sınıflandıran en uygun ayırma

çizgisini (hiper uçağı) çıkarır. Ayırma çizgisi iki boyutlu uzay boşluğunda her bir sınıfın iki tarafından da uzandığı ve uzay boşluğundaki noktaları iki parçaya böldüğü bir düzlemdir.

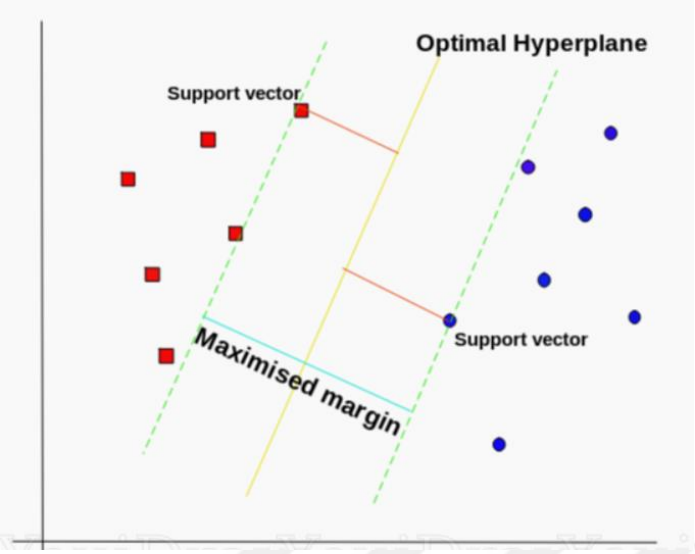
Standart SVM başlangıçta ikili sınıflandırmalar için tasarlanmıştır [15]. Fakat birçok pratik uygulama genellikle çok sınıflandırma probleminden oluşmaktadır ve bu durumlarda ise ikili olan sınıflandırmaya indirgenme gerçekleştirilmektedir. Şimdiye kadar çoklu sınıflandırma problemlerinin yeniden yapılandırılması ve ayrıştırılması konusunda çeşitli yöntemler önerilmiştir. Önerilen yöntemlerden en başta gelen Rifkin [16] tarafından önerilen yöntemdir.

SVM algoritmasında en temel zorluk verileri en ideal ayrım sağlayan düzlemin oluşturulmasıdır. Buna örnek verecek olursak aşağıdaki Şekil 2.4'teki gibi ayrım sağlayan düzlemler seçilebilir.



Şekil 2.4: SVM veri kümesi ayrımı[17].

Şekil 2.4'te iki farklı ayrım sağlayan çizgiler bulunmaktadır. Bu çizgileri inceleyecek olursak sarı renkte olan çizgi yeşil renktekine nazaran daha başarılı sınıflandırma gerçekleştirmektedir. Şekilde 2.4'te bulunan kırmızı ve mavi noktalar birer destek vektördür. Destek vektörler ve çizgi arasındaki mesafe marj olarak adlandırılmaktadır. Burada asıl amaç marjın maksimize edilmesidir. Optimum bir hiper düzlem, marjinin maksimum olduğu hiper düzlemdir. Dolayısıyla SVM iki sınıf arasındaki mesafenin mümkün olduğunca geniş olacağı şekilde karar sınırı vermeye çalışmaktadır.



Şekil 2.5: SVM kullanılarak en uygun çizgi belirlenmesi [17].

Şekil 2.5'te görüldüğü üzere en uygun çizginin belirlenmesi için destek vektörlerin çizgiye olan maksimum uzaklıkları hesaplanmaktadır.

Standart Destek Vektör Makineleri formülüne bakacak olursak eğitim veri kümesini $S=\{(x_1,y_1), (x_2,y_2), \dots ,(x_L,y_L)\}$ şeklinde ifade edersek $x_i \in R^n$ dır. Sınıf etiketi ise $y_i \in \{-1,+1\}$.
Vapnik orijinal formülüne göre SVM sınıflandırıcı formül 2.3' te gösterilmektedir[15].

$$D(x)= w^T \psi(x) + b \qquad (2.3)$$

$\psi(x)$: Doğrusal olmayan fonksiyondur. Bu fonksiyon x girdi uzayını bir özellik uzayına eşlemektedir. Eşlenen uzay muhtemelen sonsuzdur.
w^T : M boyutlu bir vektörü temsil etmektedir.
b : Skaler bir değeri temsil etmektedir.

Verileri özellik alanında doğrusal olarak ayırmak için karar yöntemi aşağıdaki koşulları sağlamaktadır.

$$y_i (w^T \psi(x) + b) \geq 1 \qquad i=1, \dots , L \qquad (2.4)$$

Optimum ayrıştırma hiper düzlemini belirlemek için iki sınıf arasında maksimum marjı olan optimizasyon formülünü aşağıdaki şekilde ifade edebiliriz.

$\min_{w,b} J(w,b) = ½\, w^T w$ öyle ki,

$y_i (w^T \psi(x) + b) \geq 1 \qquad i=1, \ldots, L \qquad (2.5)$

Eğitim verileri doğrusal olarak birbirinden ayrılamaz ise kullandığımız 2.4 formülüne ek gevşek değişken ξ_i eklememiz gerekmektedir.

$y_i (w^T \psi(x) + b) \geq 1 - \xi_i \qquad i=1, \ldots, L \qquad (2.6)$

$\xi_i \geq 0 \qquad i=1, \ldots, L \qquad (2.7)$

Optimum ayırma düzlemi elde etmek için aşağıdaki 2.8 formülünü kullanmamız gerekmektedir.

$$\min_{w,b,\xi} J(w,b,\xi) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^{l} \xi_i \,, \qquad (2.8)$$

Öyle ki,

$y_i (w^T \psi(x) + b) \geq 1 - \xi_i \qquad i=1, \ldots, L$

$\xi_i \geq 0 \qquad i=1, \ldots, L$

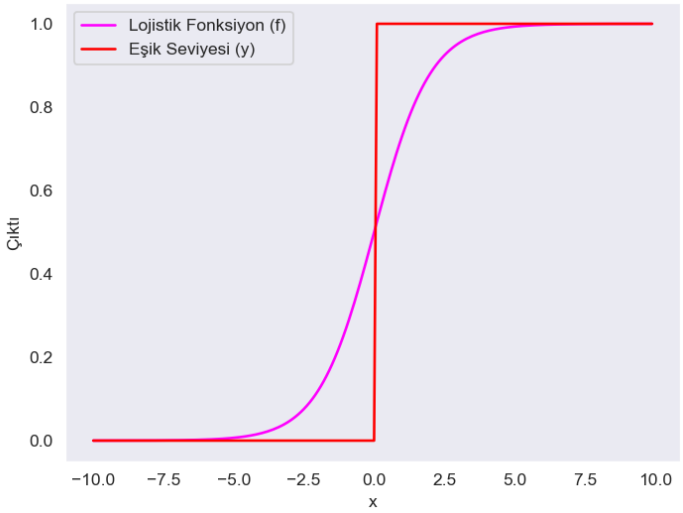
γ parametresi maksimum ayrım ve minimum sınıflandırma hatası arasındaki değiş tokuşu belirleyen bir parametredir.[15]

2.1.3. Lojistik Regresyon

İkili (Binary) sınıflandırma problemini çözmeye yarayan bir sınıflandırma algoritmasıdır. İkili sınıflandırmada genellikle sonuç 0 veya 1 olarak tanımlanmaktadır.

$$\phi(z) = \frac{1}{1 + e^{-z}} \qquad (2.9)$$

Lojistik fonksiyon (f(x)), -∞/+∞ aralığındaki tüm değerleri almaktadır. Girdi olarak aldığı değerlere karşılık 0 ve 1 olarak çıktı değerleri üretmektedir. Şekil 2.6'da fonksiyonun üretmiş olduğu grafikte de görüldüğü üzere fonksiyon -∞/+∞ değerler alarak 0 ve 1 çıktısı üretebilmektedir.



Şekil 2.6: Lojistik regresyon grafiği.

Lojistik regresyon fonksiyonuna matematiksel olarak bakıldığında çoklu lineer bir fonksiyonu tahmin etmektedir.

$$\text{logit}(S) = \{ b_0 + b_1M_1 + b_2M_2 + b_3M_3 \ldots . b_kM_k \}$$

Burada S istenen özelliğin olma olasılığını temsil etmektedir. M değeri kestirici değeri temsil etmektedir. b ise modelin kesişmesini temsil etmektedir [18].

Multinomiyal Lojistik Regresyon (MLR) birden fazla bağımsız değişkeni temel alan bağımlı bir değişkende kategorik üyeliği olasılığını tahmin etmeye çalışmaktadır. MLR terimi bağımlı veya sonuç değişkeninin ikiden fazla kategorisine izin veren ikili lojistik regresyon algoritmasının bir uzantısıdır [19]. Lojistik regresyondaki gibi MLR' de kategorik üyelik olasılığını tahmin etmek için maksimum olasılık tahminini kullanmaktadır.

Kullandığı hipotez fonksiyonundan dolayı Softmax Regresyonu olarak da bilinmektedir. MLR denetimli öğrenme algoritmasında yer alan ve metin sınıflandırmasını da dahil olmak üzere birçok problemde yaygın olarak kullanılmaktadır [19]. MLR algoritmasının potansiyelini ispatlayan ve veri madenciliği algoritmaları ile karşılaştırılması konusunda çok fazla araştırma mevcuttur [20].

Eğitimde kullanılan veri seti $M(x_i,y_i)$ çiftlerinden oluşmaktadır ve k tüm olası sınıfların sayısı kabul edilmektedir. Aynı zamanda kelime çantası kullanılarak $\{w_1, , \ldots , w_n\}$ metinlerimizde görünebilecek n kelime kümesini temsil etmektedir [19].

2.2. DERİN ÖĞRENME YÖNTEMLERİ

2.2.1. Kelime/Sözcük çantası (Bow)

Kelime/Sözcük çantası modeli bir metinin makine öğrenme algoritmaları yardımıyla modellenirken metin verilerinin temsil edilmesinin bir yöntemidir. Sözcük çantası modeli basit, anlaşılır ve uygulanması kolay bir model olup belge sınıflandırma ve dil modellenmesi gibi problemlerde büyük başarılar elde etmektedir.

Makine öğrenmesi ve derin öğrenme algoritmaları doğrudan ham metin verileri üzerinde çalışmamaktadır. Metin içerikli veriler spesifik olarak sayı vektörlerine dönüştürülmesine ihtiyaç duymaktadır. Çeşitli dil özelliklerini ifade etmek için metin verilerinden x vektörleri türetilmektedir [21]. Bu işlem özellik kodlaması ya da özellik çıkarımı olarak tanımlanmaktadır. Bu yaklaşımda metin içerisinde bulunan kelimelerin histogramı incelenmektedir, yani her kelimenin bir özellik teşkil ettiği düşünülmektedir [21]. Sözcük çantası istediğimiz kadar basit veya karmaşık olabilir. Karmaşıklık bilinen kelimeler için kelime haznesinin nasıl tasarlanacağına karar verirken ortaya çıkmaktadır.

Kelime sözcük çantası yöntemini daha iyi anlamak için bir örnek üzerinde incelemek için birkaç metin satırına ihtiyaç duymaktayız.
“It was the best of times”
“It was the worst of times”
“It was the age of wisdom”
“It was the age of foolishness”

Her cümleyi ayrı bir belge olarak değerlendirirsek ve noktalama işaretlerini değerlendirmeden burada bulunan dört cümleden kelimelerin listesini oluşturursak aşağıdaki sonucu elde etmiş oluruz.

‘It’, ‘was’, ‘the’, ‘best’, ‘of’, ‘times’, ‘worst’, ‘age’, ‘wisdom’, ‘foolishness’

Bir sonraki adım vektörlerin yaratılmasıdır. Yaratılan vektörler makine öğrenmesi algoritmaları tarafından kullanılabilecek metin değerleridir. Bu aşamada birinci belgeyi ele alırsak on adet benzersiz kelimeden oluşan kelimelerin sıklığını kontrol edelim.
“it” = 1
“was” = 1

“the” = 1
“best” = 1
“of” = 1
“times” = 1
“worst” = 0
“age” = 0
“wisdom” = 0
“foolishness” = 0

Belgelerin geri kalanları da aşağıdaki gibidir:

“It was the best of times” = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
“It was the worst of times” = [1, 1, 1, 0, 1, 1, 1, 0, 0, 0]
“It was the age of wisdom” = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0]
“It was the age of foolishness” = [1, 1, 1, 0, 1, 0, 0, 1, 0, 1]

Bu yaklaşımda her kelime “gram” olarak adlandırılmaktadır. İki kelime çiftli bir kelime haznesini yaratma yaklaşımını ise bigram modelidir [21]. Birinci belge için bigram modelini inceleyecek olursak aşağıdaki gibi bir çıktı elde edeceğiz.

“it was”
“was the”
“the best”
“best of”
“of times”

Bigram çantası, sözcük çantasından çok daha performanslı sonuçlar üretmektedir aynı zamanda çoğu yönü ile bigram çantası daha üstün kabul edilmektedir. Bir kelime haznesi seçildikten sonra bu kelime haznesindeki belgelerin kelimelerinin skorlanması gerekmektedir. Bu işlem Scoring olarak adlandırılmaktadır ve farklı yöntemlerle gerçekleştirmek mümkündür.

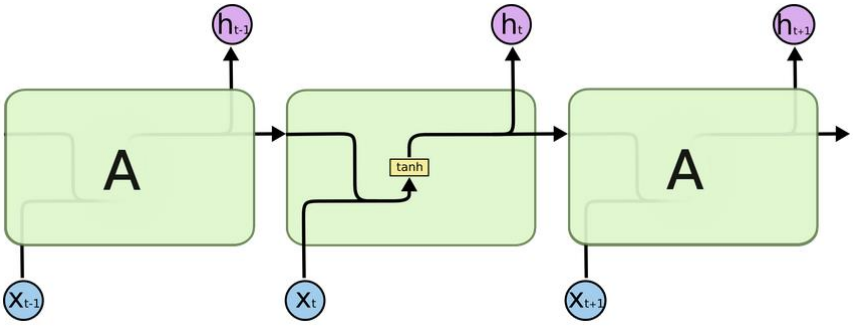
Binary Scoring: BoW genelde bu yöntemi kullanır ve belgede olup olmaması durumuna göre 0 veya 1 değerini vermektedir.

Count: Her kelimenin belgede görüldüğü sayının hesaplanması
Frequencies: Her kelimenin belgede görülme sıklığını belgedeki tüm kelimeler kullanılarak hesaplanması

Metin tabanlı veriler BoW ile sayısal vektörlere dönüştürüldükten sonra makine öğrenmesi ve derin öğrenme algoritmalarında kullanılmaktadır.

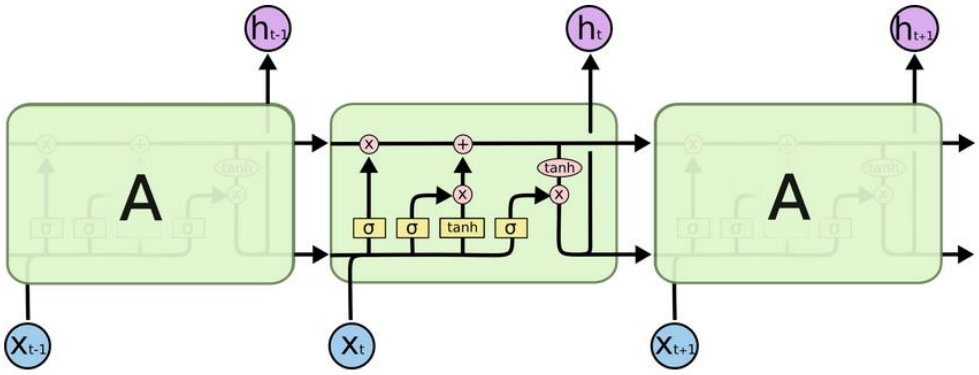
2.2.2. Uzun-Kısa Süreli Hafıza (LSTM)

LSTM uzun-kısa vadeli bağlamsal öğrenme özelliklerine sahiptir ve geçmiş bağlam bilgilerini saklayan derin öğrenme algoritmalarından biridir [22]. Uzun kısa süreli hafıza blokları, tekrarlayan sinir ağları algoritmasının bir parçasıdır. TSA insan düşüncesini daha etkin bir şekilde taklit edilmesine yardımcı olabilecek yapay bellek işlemlerinden faydalanmak için geliştirilmiştir. Araştırmalarda yaygın olarak kullanılan LSTM algoritması çok çeşitli problemler üzerinde çok performanslı çalışmaktadır. Özel bir RNN algoritması türü olan LSTM algoritması uzun vadeli bağımlılıkları öğrenebilmektedir. RNN algoritmasında döngü olarak tekrar eden bir modül zinciri biçimindedir [23]. Şekil 2.7 de RNN algoritmasının tekrarlayan modül yapısı ve tek bir basit tanh katmanı yapısı görülmektedir.



Şekil 2.7: Standart RNN mimarisi [23].

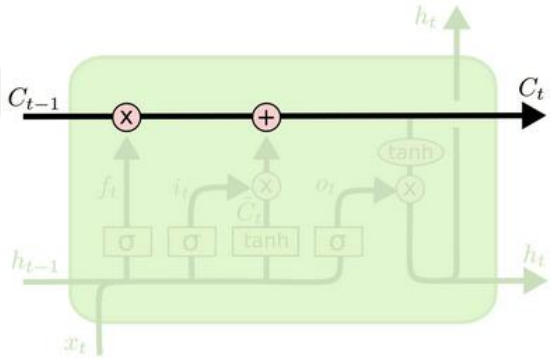
RNN algoritmasında görüldüğü gibi LSTM algoritmasında da birbirini takip eden yapılar bulunmaktadır. LSTM algoritmasında RNN' e göre bir sonraki hücre tek bir sinir ağı katmanı yerine birbiriyle etkileşimde olan dört parçadan oluşmaktadır. Şekil 2.8' deki görselde tüm hücrelerin vektör girişleri başka bir hücrenin çıktısından taşınmaktadır [23].



Şekil 2.8: Uzun-Kısa Süreli Hafıza mimarisi.

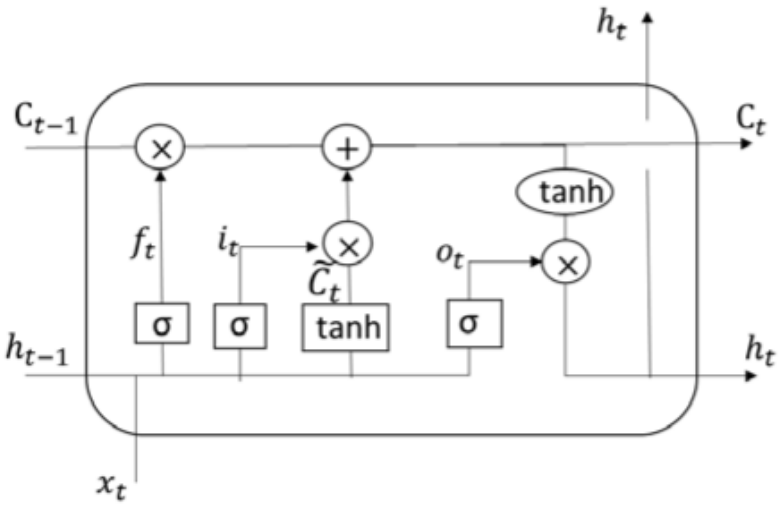
Şekil 2.8' de görülen pembe daireler vektör eklenmesi gibi noktalara yönelik işlemleri temsil etmektedir. Sarı kutular ise sinir ağı katmanlarını öğrenen yapıları temsil etmektedir.

LSTM algoritmasının sırrı hücre durumlarıdır, yatay bir çizgi hücre diyagramının üzerinden geçmektedir.



Şekil 2.9: Uzun-Kısa Süreli Hafıza Hücre Yapısı.

Şekil 2.9'da görüldüğü gibi hücre durumu bir tür taşıma bandı gibi görülmektedir. Bilginin taşınması durumunda küçük doğrusal etkileşimler ile hücre boyunca uzanmaktadır. Bilginin taşınması sırasında etkileşim yapılmadan taşımak çok kolaydır. LSTM algoritması hücre durumuna bilgi eklenmesi ve çıkarılması yetkilerine sahiptir ve bu işlem kapılar tarafından düzenlenmektedir [24]. Kapılar isteğe bağlı olarak bilgilerin akışını kontrol etmektedir. Kapılar sigmoid yapay sinir ağları ve noktasal çarpma işlemlerinden oluşmaktadırlar. Sigmoid katmanı çıktı olarak 0 ve 1 arasında değerler üretmektedir. Üretilen çıktı değeri bileşenlerin ne kadarının geçmesi gerektiğini açıklamak için kullanılmaktadır. Üretilen çıktı değeri 0 ise hiçbir bileşenin geçmesine izin verilmediği anlamına gelmektedir. Üretilen çıktı değeri 1 ise her şeyin geçmesine izin verildiği anlamına gelmektedir.



Şekil 2.10: Standart LSTM modeli[24].

LSTM algoritmasının üç kapıdan oluşmaktadır. Bunlar sırasıyla forget gate, input gate, output gate olarak algoritma yapısını oluşturmaktadır [24]. Birinci sırada gelen forget gate hangi bilgilerin hücreden silineceğine karar vermektedir. İkinci sırada gelen input gate hangi bilgilerin hücre durumuna güncelleyeceğine karar vermektedir. Bu iki nokta belirlendikten sonra hücre durumu güncellenebilir duruma gelmektedir. Üçüncü sırada gelen output gate ağın son çıkısına karar vermektedir. Bu işlemlerin gerçekleştirildiği denklemler sırasıyla Şekil 2.11'deki gibidir [24].

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \#(1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \#(2)$$

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t]) + b_C \#(3)$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \#(4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \#(5)$$

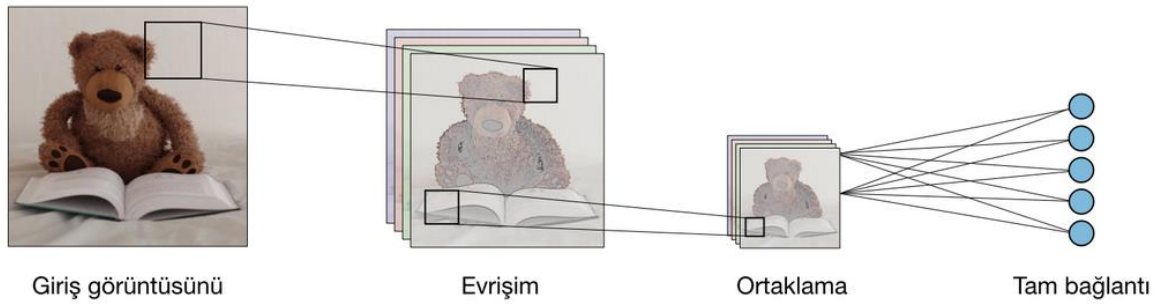
$$h_t = o_t * \tanh(C_t) \#(6)$$

Şekil 2.11: Standart LSTM model formülleri

2.2.3. Konvolüsyonel Sinir Ağları (CNN)

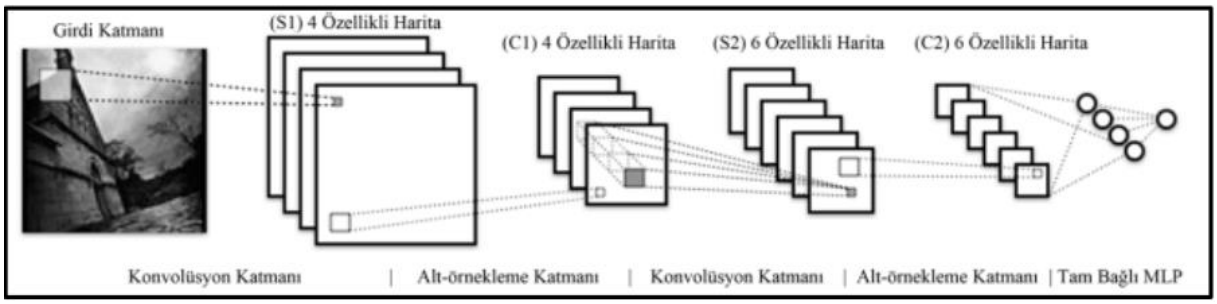
Konvolüsyonel sinir ağları 1980 yıllarında geliştirilmiş ve görüntü, metin, konuşma ve ilaç keşif problemlerinde uygulanan derin öğrenme algoritmalarından biridir [25]. Konvolüsyonel sinir ağları ileri beslemeli sinir ağları olup çok katmanlı algılayıcı ağların bir türüdür. Temel olarak hayvanların görme merkezinden örnek alınarak geliştirilmiş bir algoritmadır.

CNN' algoritması yaygın olarak görüntü işleme probleminde kullanılmaktadır, fakat görüntü işleme dışında birçok alanda özellikle doğal dil işleme problemlerinde kullanıldığını gösteren kaynaklar bulunmaktadır. Doğal dil işlemede sınıflandırma, semantik ayrım, tahmin etme, cümle modelleme gibi problemlerde çok iyi sonuçlar elde edilmiştir [26]. İlaç keşif probleminde kullanılan AtomNet mimarisi Atomwise şirketi tarafından geliştirilmiştir [26,27]. Geleneksel konvolüsyonel sinir ağları mimarisi konvolüsyon(evrişim) katmanı, ortaklama(havuzlama) katmanı ve tam bağlantı katmanı olmak üzere 3 katmandan oluşmaktadır.



Şekil 2.12: Konvolusyonel Sinir Ağları Mimarisi [36]

Şekil 2.12'de görüldüğü üzere konvolüsyon(evrişim) katmanı giriş özelliklerindeki bilgileri öğrenme ve nesnelerin tespit edildiği katmandır. Havuzlama(ortaklama) katmanı ise alt örneklemelerin oluşturulduğu ve çözünürlüğün azaltarak özellik haritasının hesaplanmasını hızlandırmaktadır. Tam bağlantı katmanı mimarinin en son kısmında bulunmaktadır ve tüm nöronlara bağlıdır. Bu katman sayesinde sonuçların optimize edilmesinde kullanılmaktadır.

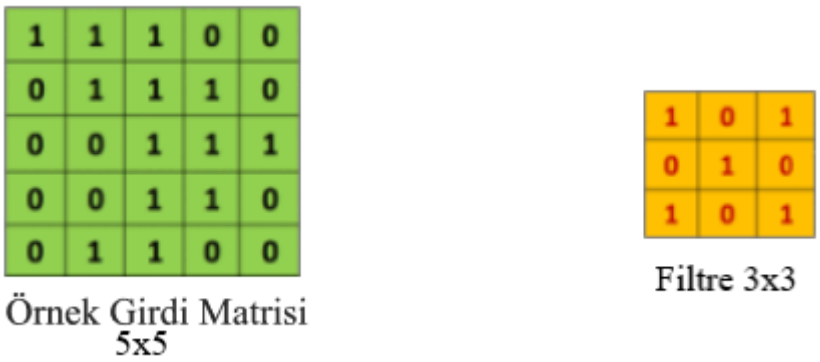


Şekil 2.13: CNN LeNet Mimarisi.[26]

Şekil 2.13'te görüldüğü üzere her bir alt örnekleme katmanında girdi üzerinde çıkarılan özellik haritaları bir sonraki konvolüsyon katmanına aktarılır. Konvolüsyon katmanlarının sayısının arttırılması daha detaylı öğrenme ve özellik çıkarımı etkilemektedir.

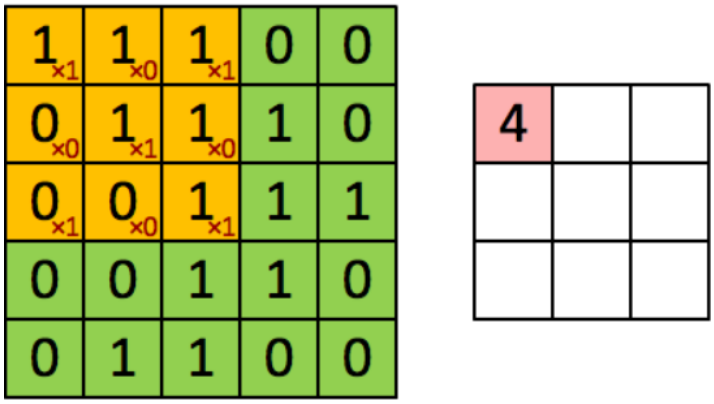
2.2.3.1. Konvolüsyon (Evrişim) Katmanı

Konvolüsyon katmanı CNN algoritmasının en önemli katmanı olup girdi üzerindeki özellikleri çıkarmaktadır. Özelliklerin çıkarımı filtre uygulanarak gerçekleştirilmektedir. Filtre uygulama işlemi aslında matris çarpımı işlemidir. Filtre girdi matrisi üzerinde kaydırılarak uygulanmaktadır. Uygulanan filtre sonucu yeni bir matriste tutulmaktadır ve bu matris özellik haritası olarak adlandırılmaktadır. Uygulanan filtre boyutları 2x2, 3x3 ya da 5x5 gibi değişkenlik göstermektedir. Basit bir örnek ile Şekil 2.15 ve Şekil 2.16'daki gibi bir girdi matrisine uygulanan filtre işlemini ifade etmektedir.



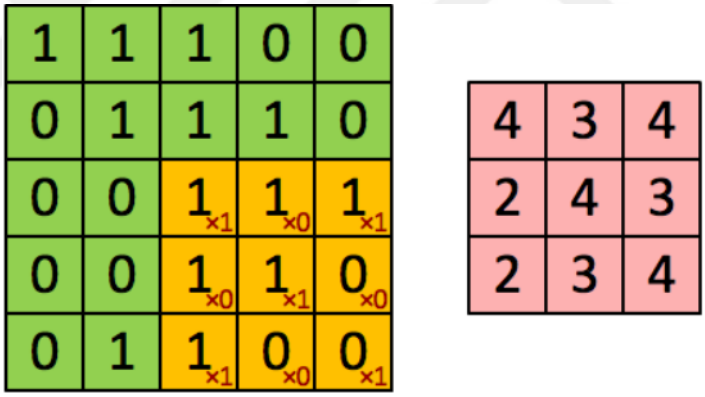
Şekil 2.14: Konvolüsyon katmanı girdi ve filtre matrisleri örneği [28]

Matris çarpım işleminin adımlarını gösterecek olursak birinci adım şekil 2.15'teki gibidir.



Şekil 2.15: Konvolüsyon katmanındaki matris çarpımının birinci adımı[28]

Şekil 2.15'te görüldüğü üzere girdi matrisi ve filtre matrisi çarpılarak elde edilen sonuç değeri yeni bir matris üzerinde tutulmaktadır. Matris çarpımı her adımda kaydırılarak devam ettirilir ve sonuç olarak yeni elde edilen Şekil 2.16'daki matris özelliklerin tutulduğu matris kabul edilmektedir.



Şekil 2.16: Konvolüsyon katmanındaki matris çarpımının son adımı[28]

Matris iç çarpım işlemi tamamlandıktan sonra elde edilen sonuç matrisi bir sonraki katmanın girdisi olarak işlemlere devam etmektedir. Çarpım sonrası oluşan matris negatif değerler içeriyorsa ReLU aktivasyon fonksiyonu kullanılarak bu negatif değerler sıfır değerine çekilir. ReLU aktivasyon fonksiyonuna gelen değer pozitif bir değer ise girdi değeri çıktı olarak yansıtılırken, gelen değer negatif bir değer ise girdi değeri sıfır çıktısını oluşturmaktadır.

Aktivasyon fonksiyonu bir düğümün bitişinde ya da iki düğüm arasında kullanılmaktadır. Bu fonksiyonlar yardımı ile üretilecek çıktı değerleri üzerinde değişiklikler yapılmaktadır.

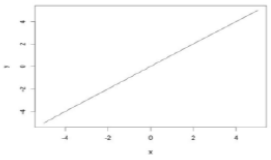
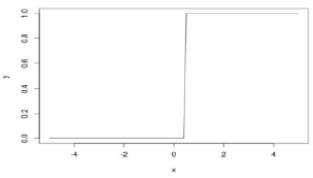
Aktivasyon fonksiyonları ya da diğer bir adı ile Transfer fonksiyonları iki ana bölümde incelenebilir.

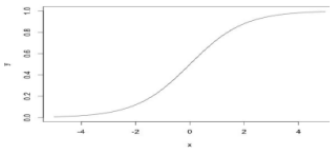
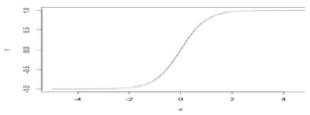
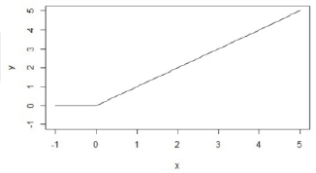
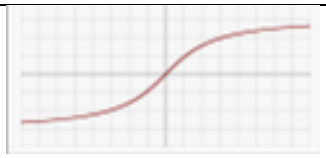
- Doğrusal Aktivasyon Fonksiyonları
- Doğrusal Olmayan Aktivasyon Fonksiyonları

Doğrusal aktivasyon fonksiyonları herhangi bir aralık arasında sınırlandırılmadan giriş değerini çıkışa yansıtmaktadır. Yapay sinir ağlarının beslendiği verilerin karmaşıklığını indirgemede ve parametrelerindeki işlemlere yardımcı olmamaktadır.

Doğrusal olmayan aktivasyon fonksiyonları en yaygın olarak kullanılan fonksiyonlardır. Çıktının ayırt edilmesini ve uyarlanmasını sağlayarak kolaylık sağlamaktadır. Doğrusal olmayan aktivasyon sonucu oluşturulan çıktı bir sonraki tabakanın girdisi olarak işlem görmektedir. Sinir ağlarında yaygın olarak kullanılan aktivasyon fonksiyonları aşağıdaki Tablo 2.1'de anlatılmaktadır.

Tablo 2.1: Popüler olarak kullanılan aktivasyon fonksiyonları[29]

Aktivasyon Fonksiyonu	Açıklama	Formül	Uygulama
Doğrusal (Lineer)	Nöron çıktısını hiç değiştirmeyen en temel fonksiyondur.	$f(x) = x$	
Adım veya Eşik (Step)	Çok basit bir fonksiyondur ve belirtilen eşik değerinin üzerindeki değerlere 1 değerini döndürür. Eşik değerin altındaysa 0 değeri döndürür.	$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$	

Sigmoid veya Lojistik	İleri beslemeli sinir ağlarında popüler olan ve sadece pozitif değerler üreten fonksiyondur.	$f(x) = \frac{1}{1+e^{-x}}$	
Tanjant Hiperbolik	-1 ve 1 arasında çıktı üretilmesi gereken sinir ağları için popüler fonksiyondur.	$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	
ReLU	Negatif girdi değerleri karşılığında 0 değeri döndüren fonksiyondur. Pozitif girdi değerlerini çıktı olarak üretir.	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	
ArcTan	ArcTan aktivasyon fonksiyonu – π/2 ve π/2 aralığındaki giriş değerlerini eşler.	$f(x) = \tan^{-1}(x)$	
Parametrik ReLU (PReLU)	Relu fonksiyonuna göre PReLU fonksiyonu negatif değerler için 0 değeri üretmez. Sabit bir sayı ile girdinin çarpımını döndürür.	$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	

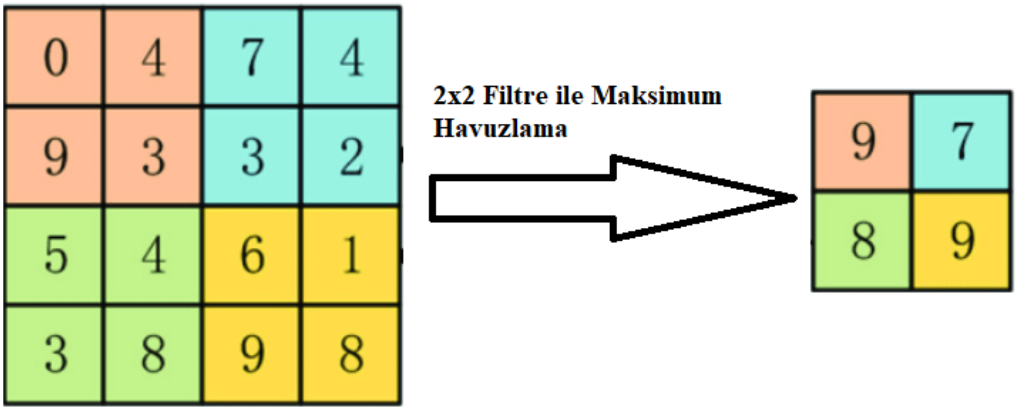
Tablo 2.1 (devam): Popüler olarak kullanılan aktivasyon fonksiyonları[29]

2.2.3.2. Havuzlama (Pooling) Katmanı

Havuzlama katmanı konvolüsyon katmanları arasında görülmektedir. Havuzlama katmanı ağın parametre ve hesaplama sayılarını azaltmaktadır. Bu katman aşırı öğrenmeyi, uzay büyüklüğünü küçülterek kontrol etmektedir. Havuzlama katmanında temel olarak iki işlem yapılmaktadır. Bunlar:

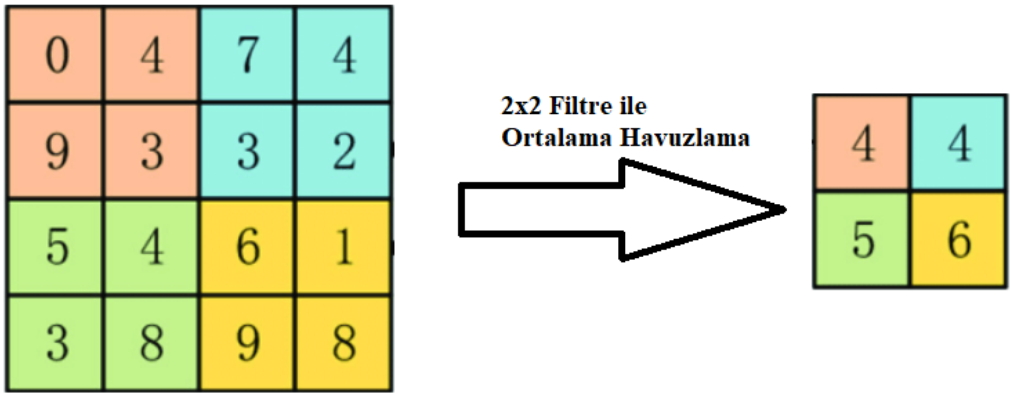
- Maksimum(Maximum) Havuzlama
- Ortalama (Average) Havuzlama

Maksimum havuzlamada adından da anlaşılacağı gibi matris üzerindeki en yüksek değeri almaktadır. Kayan filtrenin uygulanmasında oluşan matristeki her adımda maksimum olan değer alınır ve bunun yardımıyla alt örnekleme yapılmaktadır. Konvolüsyon katmanından farklı olarak, havuzlama katmanı ağın derinliğini etkilememektedir. Şekil 2.17' de görüldüğü gibi maksimum ortalama m*m boyutlu girdi matrisi üzerinde 2x2 boyutlu filtre matrisi kaydırılarak her adımda maksimum değer alınmaktadır.



Şekil 2.17: Maksimum havuzlama işlemi[30]

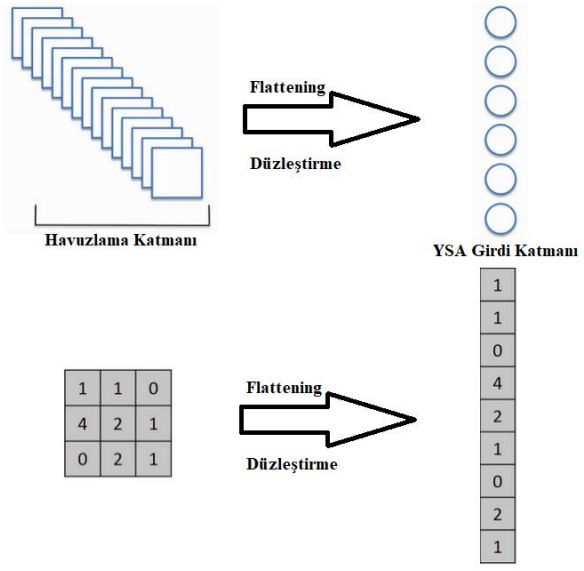
Ortalama havuzlama işleminde matris üzerindeki değerlerin ortalaması alınmaktadır. Kayan filtrenin uygulanmasında oluşan matristeki her adımda ortalama değer hesaplanıp bu değer kullanılmaktadır. Şekil 2.18' de görüldüğü gibi maksimum ortalama m*m boyutlu girdi matrisi üzerinde 2x2 boyutlu filtre matrisi kaydırılarak her adımda ortalama değer alınmaktadır.



Şekil 2.18: Ortalama havuzlama işlemi[30]

2.2.3.3. Tam Bağlantı Katmanı (Fully Connected Layer)

Tam bağlantı katmanında tanımdan da anlaşılacağı gibi bir önceki düğümden gelen bütün nöronlar bu katmandaki nöronlara bağlanmaktadır. Son havuzlama katmanının çıktısı tam bağlantılı katmanın girdisi olarak işlev görmektedir. Basit olarak tam bağlantı katmanı, havuzlama katmanından gelen doğrusal olmayan özellikleri öğrenir ve yüksek düzey özelliklere bağlı kalarak sınıflandırma işlemi gerçekleştirmektedir. Havuzlama katmanından aktarılan bilgiler düzleştirme işleminden geçmektedir. Düzleştirme işleminden geçirilmesinin sebebi havuzlama katmanından gelen bilgilerin bir sinir ağına yerleştirilmesidir. Şekil 2.19'da düzleştirme işlemi görülmektedir.

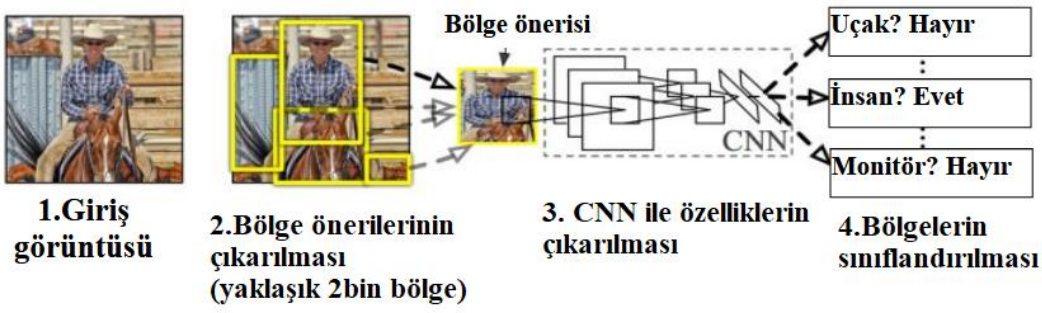


Şekil 2.19: Düzleştirme (Flattening) işlemi

Düzleştirilmiş matris sınıflandırma işlemi yapabilmek için tamamen bağlı katmandan geçmektedir. FC sıklıkla girdiyi olasılık dağılımına dönüştürmek için softmax aktivasyon fonksiyonunu kullanmaktadır. Modelin tahmin etmeye çalıştığı sınıflandırma etiketlerinin her biri için sıfır ila bir (0-1 arasında bir değer) arasında bir sayı üretmektedir.

2.2.4. Bölgesel Konvolüsyonel Sinir Ağları (R-CNN)

Bölgesel konvolüsyonel sinir ağları yaygın olarak görüntü işleme tekniğinde kullanılmaktadır. Görüntü üzerinde bulunan objelerin tahmini üzerinde çalışmaktadır. Çok sayıda obje seçme probleminin çözülmesi için Ross Girshick ve arkadaşları tarafından önerilen sinir ağları algoritmasıdır [31]. Bu algoritma yardımıyla girdi görüntüsü üzerinden seçici arama algoritması yardımıyla önerilen yaklaşık 2000 bölge seçilmektedir. Seçilen bölgeler algoritmanın devamında CNN algoritmasının girdisi olarak işlem görmektedir. RCNN algoritması sayesinde görüntü üzerinde çok sayıda bölgeyi sınıflandırmak yerine sadece önerilen bölgelerin sınıflandırılması algoritmanın performansını arttırmaktadır.



Şekil 2.20: RCNN mimarisi [31].

Şekil 2.20' de görüldüğü üzere sistem girdi görüntüsünü aldıktan sonra yaklaşık olarak 2000 bölge önerisi çıkarmaktadır ve önerilen bölge çıktıları CNN algoritması yardımı ile bu bölgelerdeki özelliklerin çıkarımları yapılmaktadır. Son aşamada ise CNN' algoritmasından çıkan özellikler yardımı ile SVM algoritması kullanılarak sınıflandırma işlemi uygulanmaktadır.

Ross Girshick tarafından önerilen model üç modülden oluşmaktadır. Birinci modül kategoriden bağımsız olarak bölgesel önerilerin çıkarıldığı modüldür. Öneriler tahmin etme işlemi için aday tespit kümesini tanımlamaktadır. İkinci modül her bir önerilen bölge için sabit uzunlukta özellik vektörünün çıkarımının yapıldığı geniş konvolüsyonel sinir ağıdır.

Üçüncü ve son modül ise sınıf parametresine özgü doğrusal destek vektör makineleridir (SVM).

3. MALZEME VE YÖNTEM

3.1. PROGRAMLAMA DİLİ VE DONANIM

Bu bölümde çalışmada kullanılan programlama teknolojileri ve donanım bilgileri anlatılmaktadır. Derin öğrenme alanında son yıllarda açık kaynak kodlu yazılımlarda artış görülmektedir. Microsoft'un ML.NET, Google'ın Tensorflow gibi kütüphaneler ve önde gelen şirketlerin paylaştıkları yazılımlar açık kaynak olarak paylaşılmaktadır. Genişleyen açık kaynak ekosistemi sayesinde derin öğrenme ve makine öğrenmesi alanında çalışmalar günden güne artmaktadır.

Derin öğrenme ve makine öğrenmesi algoritmaları genellikle Python programlama dili ile yazılmaktadır. Çalışmada kullanılan kaynak kod Python programlama dili ile yazılmıştır. Aynı zamanda Anaconda yazılımının dağılımı olan Jupyter Notebook kullanılmaktadır. Jupyter Notebook kullanım alanları genellikle veri temizleme, veri düzenleme, sayısal simülasyon, istatistiksel modelleme, makine öğrenmesi, derin öğrenme gibi birçok alanda yaygın olarak kullanılmaktadır. Aynı zamanda makine öğrenmesi için uçtan uca açık kaynak kodlu ekosistem olan Tensorflow kütüphanesi kullanılmaktadır. Hızlı geliştirme yapmak için geliştirilen Keras derin öğrenme kütüphanesi kullanılmaktadır. Flask kütüphanesi oluşturulan modelin canlı olarak web servis üzerinden test edilmesi işleminde kullanılmaktadır. Numpy kütüphanesi ile çok boyutlu dizi ve matris işlemlerinde kullanılmaktadır. Kullanılan açık kaynak kodlu yazılımlar, yapılan çalışmada kolaylıklar sağlayarak hızlı geliştirme ile zamandan tasarruf edilmesini sağlamaktadır.

Derin öğrenme ve makine öğrenmesi uygulamalarını istatistiksel ve matematiksel işlemler gerçekleştirdiği için güçlü donanım ile desteklenmelidir. Donanımın yetersiz olması durumlarında eğitim süresinde artış gözlemlenmektedir. Bu nedenle efektif olarak yüksek hesaplamalar gerçekleştirebilen donanım bileşenlerinin kullanılması önerilmektedir. Bu kapsamda çalışmada kullandığımız donanım bilgileri aşağıdaki gibidir:

İşlemci(CPU): Intel Core i7-8750H 2.20GHz
Bellek(RAM): 16 GB 2400MHz

Ekran Kartı(GPU): 8GB NVIDIA GTX1070
Depolama: 512GB SSD

3.2. VERİ SETİNİN HAZIRLANMASI

Derin öğrenme ve makine öğrenmesi algoritmaları tahmin ve sınıflandırma yapabilmek için önceden eğitilmiş veri setlerine ihtiyaç duymaktadır. Bu sebeple bu algoritmaları kullanmak için model eğitiminde kullanılacak veriler toplanır ve ön işlemlerden geçirilmektedir. Denetimli öğrenme algoritmaları kullanılacak ise veri setindeki verilerin etiketlenmesi gerekmektedir. Etiketleme işlemi veri setinin ön işlemler aşamasında gerçekleştirilmektedir. Çalışmamızda iki farklı veri kümesi kullanılmıştır. Tablo 3.1'de veri kümesi hakkında örnek sayısı ve sınıf sayısı bilgileri görüntülenmektedir.

Tablo 3.1: Veri Kümesi

Veri Seti	Örnek Sayısı	Sınıf Sayısı	Eğitim Oranı	Test Oranı
I	10.056	120	%80	%20
II	9686	2	%80	%20

Tablo 3.1'de sunulan I numaralı veri seti makine öğrenmesi ve derin öğrenme yöntemleri ile eğitilerek elde edilen model test edilmiştir. Tablo 3.1'de sunulan II numaralı veri seti ise sadece CNN algoritması ile eğitilerek oluşturulan model test edilmiştir. Veri kümeleri arasındaki farklılık verilerin etiketlendiği sınıf sayısıdır. Tablo üzerinde de görüldüğü gibi I numaralı veri setinde 120 sınıf bulunurken II numaralı veri setinde 2 sınıf bulunmaktadır. I numaralı veri setinde 120 sınıfın içeriği bireysel emeklilik işlemleri ile ilgili olan soruların etiketlenmiş olduğu sınıfları temsil etmektedir. II numaralı veri seti ise argo (istenmeyen hakaret ifadeler içeren) cümlelerin tespit edilmesi için geliştirilmiştir. Var/Yok olmak üzere iki farklı sınıftan oluşmaktadır. Bir cümlenin içinde istenmeyen kelimelerin var olup olmadığını tespit etmek için kullanılmaktadır.

I numaralı veri seti içeriğinde bir emeklilik bankasının müşteri ilişkileri bölümünde bulunan kayıtların belirlenen sınıflar ile ilişkili olan cümlelerinden oluşmaktadır. Müşteri ilişkileri bölümünden alınan veriler ön işlemden geçirilerek ilgili sınıf bilgisiyle etiketlenerek veri kümesi oluşturulmuştur. Alan uygulamasına yönelik bir çalışma olduğu için sınıf sayısı alanda kullanılacak kategoriler ile bağlantılıdır. I numaralı veri setindeki sınıf içeriğine örnek vermek gerekirse Bireysel Emeklilik, Hayat Sigortası, Fon Bilgileri, Ürün Bilgileri, Katkı Payı gibi sınıflardan oluşmaktadır.

II numaralı veri seti veri tarayıcı(data crawler) yardımcı araç kullanılarak oluşturulmuştur. Sosyal medya üzerinde toplum erişimine açık olan yorumlar toplanmıştır. Yorumların küfür(argo) içerikli olanlar ve olmayanlar olarak filtrelenerek toplanmıştır. Toplanan veriler ön işlemden geçirilerek küfür(argo) kelimelerin içerip içermediği durumlara göre etiketlenmiştir. Etiketleme işleminde iki sınıf (var,yok) bulunmaktadır. II numaralı veri setinin amacı oluşturulan model üzerinde gelen cümlelerin küfür(argo) içerikli kelimeler içerip içermediğini tespit etmektir.

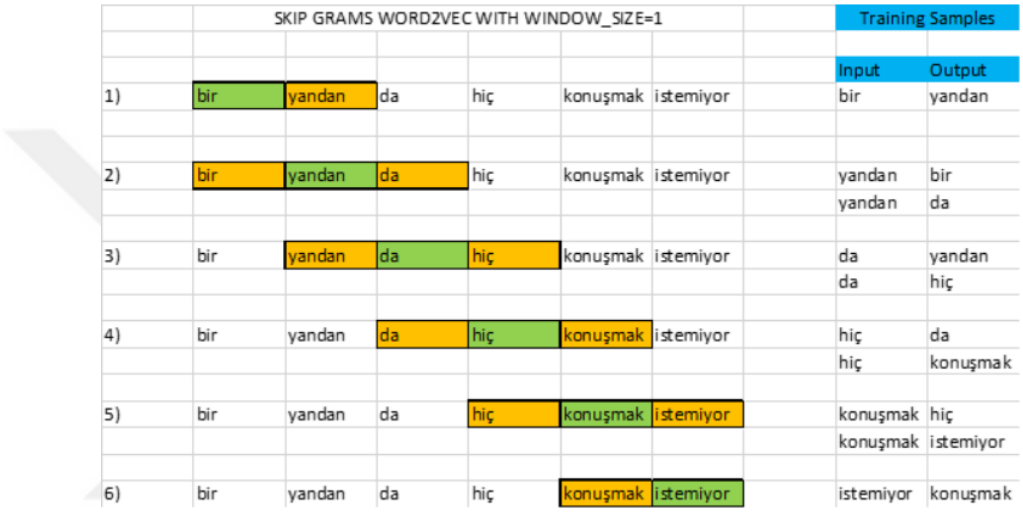
3.3. KELİMELERİN VEKTÖR UZAYINDA İFADE EDİLMESİ (WORD2VEC)

Tomas Mikolov ve arkadaşları tarafından 2013 yılında geliştirilen denetimsiz makine öğrenmesi modelidir [32]. Word2Vec metin tabanlı veriler üzerinde işlem yapan iki katmanlı bir sinir ağıdır. Girdi olarak metinler alarak karşılığında bu metinlerin vektör değerlerini hesaplamaktadır. Word2Vec benzer kelimeleri vektör uzayında birbirine yakın bir şekilde gruplamaktadır. Mimarisinde derin bir sinir ağı içermiyor olsa da, metin tabanlı verilerde derin sinir ağlarının işlem yapabileceği formatta sayısal değerler üretmektedir. Word2Vec işleminden elde ettiğimiz çıktı bir sözlük(kelime haznesi) olup girdi olarak alınan bütün sözcüklerin vektör karşılığını barındırmaktadır. Bu sözlük(kelime haznesi) yardımıyla derin öğrenme ağının beslenmesi gerçekleştirilir ya da kelimelerin arasındaki ilişkiler tespit edilebilmektedir. Word2Vec işleminin iki alt yöntem kullanılarak uygulanabilir. Skip-Gram ve Continous Bag of Word(CBOW) alt yöntemleri ile Word2Vec işlemi yapılmaktadır.

3.3.1. Skip-Gram

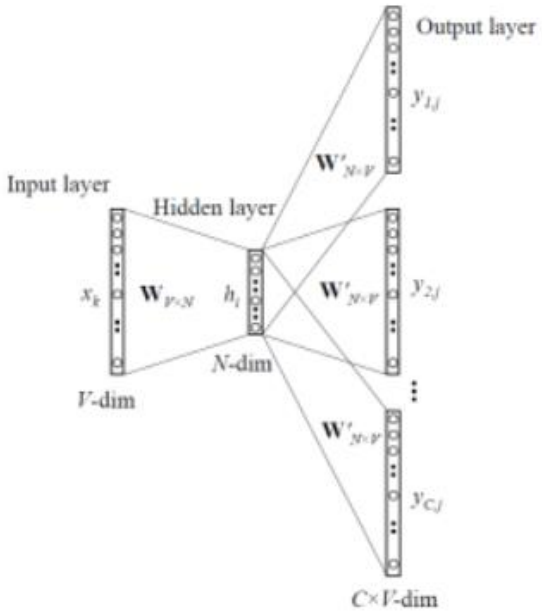
Bu modelin amacı tahmin etmede kullanılmak için cümle veya dosyadaki kelimelerin vektör karşılıklarını elde etmektir. Büyük veri setlerinde daha iyi performans göstermektedir ve

CBOW modeline göre hesaplama gücü ihtiyacı daha fazladır. Skip-Gram modelinin arkasındaki temel fikir tanımlanmış pencere sayısına göre kelimeleri tek tek alır ve pencerenin etrafındaki diğer pencerelerdeki verileri tahmin etmeye çalışır. Girdi parametresi olarak aldığı pencere ile etrafındaki pencereyi tahmin etmeye çalışır. Pencereler kelimeleri temsil etmektedir. Şekil 3.1'de Skip-Gram modelinin pencere sayısı 1 olan örneği görüntülenmektedir. Şekildeki yeşil etiketli kutular girdi değerlerini, turuncu olan kutular ise çıktı değerlerini temsil etmektedir.



Şekil 3.1: Skip-Gram model örneği[33]

Skip-Gram modeli üç katmandan oluşmaktadır. Bu katmanlar girdi katmanı, gizli katman ve çıktı katmanı olarak bilinmektedir. Gizli katman ağırlık matrisi ve girdi vektörü arasında nokta çarpımı işlemini gerçekleştiren katmandır. Gizli katmanda aktivasyon fonksiyonu kullanılmamaktadır. Nokta çarpımından elde edilen sonuç çıktı katmanına aktarılmaktadır. Çıktı katmanı gizli katmandan gelen veri ile çıktı katmanındaki ağırlık matrisinin nokta çarpımı işlemini gerçekleştirir. Nokta çarpımından elde edilen sonuca softmax aktivasyon fonksiyonu uygulanarak girdi olarak verilen kelimelerin olasılığı hesaplanmaktadır. Şekil 3.2'de Skip-Gram modelinin mimari yapısı görülmektedir.



Şekil 3.2: Skip-Gram modeli[34]

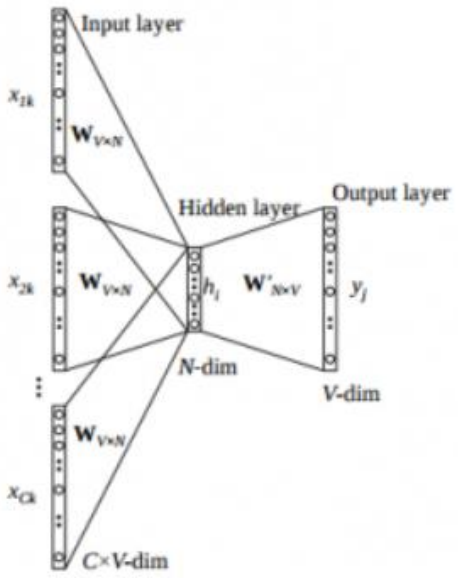
3.3.2. Continous Bag of words (CBOW)

CBOW modeli merkez penceredeki kelimeyi tahmin etmek için etrafındaki pencerelerdeki değerleri kullanarak merkez penceredeki değeri tahmin etmeye çalışmaktadır. Skip-Gram modelinin çalışma mantığının tam tersi bir mantıkla çalışmaktadır. Genellikle daha küçük veri setlerinde performansı daha yüksektir. Skip-Gram modeline göre daha az hesaplama gücüne ihtiyaç duymaktadır. Şekil 3.3'te CBOW modelinin pencere sayısı 1 olan örneği görüntülenmektedir. Şekildeki yeşil etiketli kutular girdi değerlerini, turuncu olan kutular ise çıktı değerlerini temsil etmektedir.

	CBOW WORD2VEC WITH WINDOW_SİZE =1						Training Samples	
							Input	Output
1)	bir	yandan	da	hiç	konuşmak	istemiyor	yandan	bir
2)	bir	yandan	da	hiç	konuşmak	istemiyor	bir	yandan
							da	yandan
3)	bir	yandan	da	hiç	konuşmak	istemiyor	yandan	da
							hiç	da
4)	bir	yandan	da	hiç	konuşmak	istemiyor	da	hiç
							konuşmak	hiç
5)	bir	yandan	da	hiç	konuşmak	istemiyor	hiç	konuşmak
							istemiyor	konuşmak
6)	bir	yandan	da	hiç	konuşmak	istemiyor	konuşmak	istemiyor

Şekil 3.3: Continous Bag of Words (CBOW) model örneği[33]

CBOW modeli de üç katmandan oluşmaktadır. Bu katmanlar girdi katmanı, gizli katman ve çıktı katmanı olarak bilinmektedir. Skip-Gram modelindeki gibi CBOW modelinde de katmanlar aynı işlemleri yapmaktadırlar. Şekil 3.4'de CBOW modelinin mimari yapısı görülmektedir.



Şekil 3.4: Continous Bag of Words (CBOW) modeli[34]

4. BULGULAR

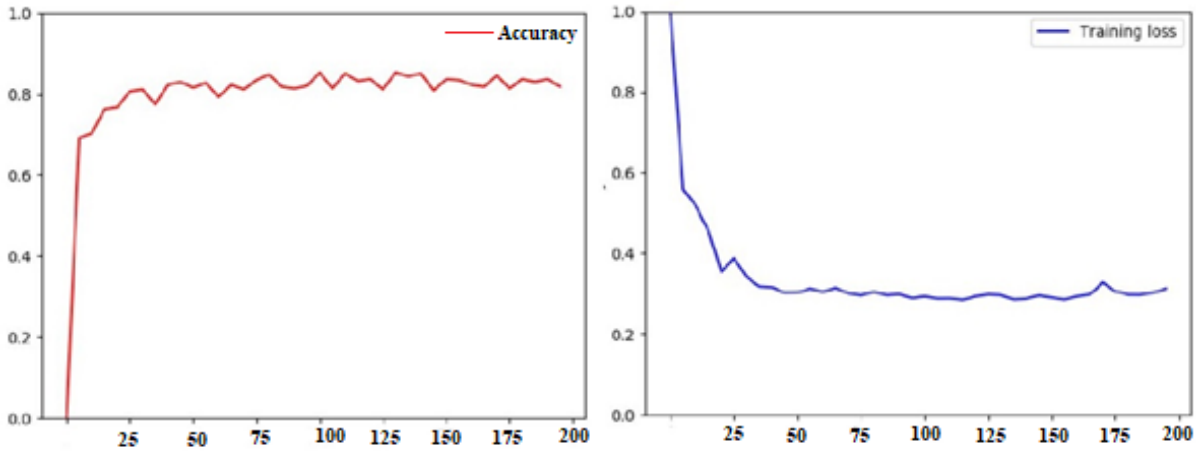
Bu tez çalışmasında derin öğrenme ve makine öğrenmesi algoritmalarının cümle sınıflandırma modeli oluşturulmuştur. Sınıflandırma çok sınıflı ve ikili sınıflandırma modeli olarak ele alınmıştır. İki ayrı veri seti kullanılarak CNN algoritmasının sınıflandırma başarımı diğer algoritmalar ile karşılaştırılmıştır. Aynı zamanda model eğitiminde harcanan sürenin algoritmalar arasındaki farkı da karşılaştırılmıştır. Model eğitimi için oluşturulan veri setinin %80 eğitim için kullanılırken %20'lik kısmı modelin başarımının ölçülmesinde kullanılmıştır.

Tablo 4.1: Sınıflandırma modellerinin karşılaştırılması

Veri Kümesi	Model	Süre	Başarı Oranı
I	SA-LSTM	23 (dk)	% 63.9
I	LM-LSTM	25 (dk)	% 68.8
I	RCNN	45 (dk)	% 71.8
I	WORD CNN	38 (dk)	% 47.2
I	MNB	18.5 (ms)	% 35.2
I	SVM	20.1 (ms)	% 59.9
I	Lojistik Regresyon	17 (ms)	% 60.3
I	Keras ile BOW	8 (sn)	% 46.2
I	CNN	77 (dk)	% 85.1
II	CNN	65 (dk)	% 92.4

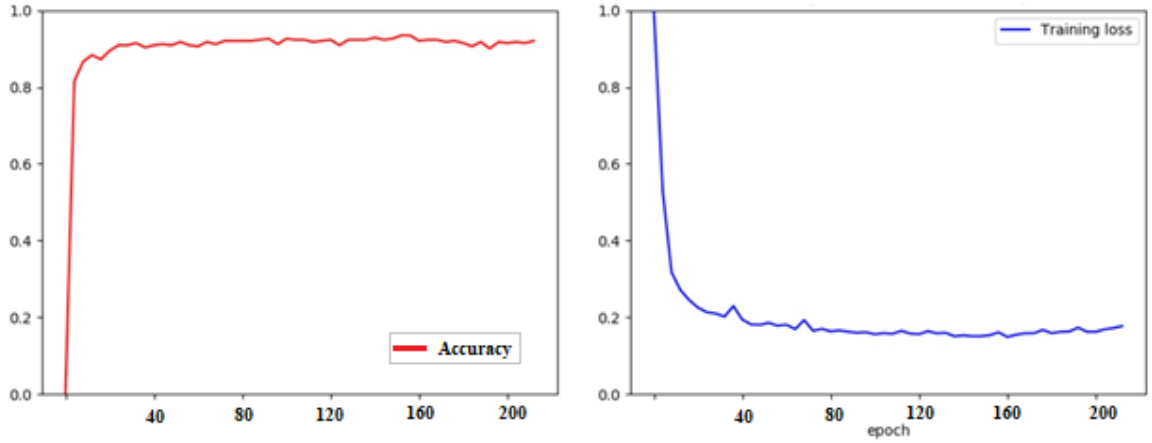
Çalışmada elde edilen sonuçlar Tablo 4.1'de gösterilmiştir. Elde edilen sonuçlara göre CNN algoritması diğer algoritmalara kıyasen oluşturulan veri seti üzerinde daha başarılı sonuçlar göstermektedir. Aynı zamanda makine öğrenmesi ve derin öğrenme algoritmaları kullanılarak eğitilen modellerde eğitimde harcanan sürede ciddi farklılıklar gözlemlenmektedir. Makine öğrenmesi algoritmalarının model eğitiminde harcanan süre milisaniye cinsinden olup derin öğrenme algoritmalarında harcanan süre saatleri geçmektedir. Model eğitiminde harcanan süre her ne kadar donanım ile bağlantılı olsa da yapılan eğitimler aynı donanım üzerinde gerçekleştirilmiştir. Bu kapsamda makine öğrenmesi algoritmalarının eğitim süresinin çok daha kısa olduğunu gözlemlemekteyiz.

Şekil 4.1'de I numaralı veri seti kullanılarak eğitilen CNN modelinin doğruluk(accuracy) ve kayıp(loss) fonksiyonlarının ürettiği çıktı grafikleri gözlemlenmektedir. Grafik üzerinde eğitim onaylama (train validation) noktaları ele alınmıştır. Grafik üzerindeki Y eksenindeki değer doğruluk(accuracy) ve kayıp(loss) değerlerini ifade ederken X eksenindeki değerler devir(epoch) değerini temsil etmektedir.



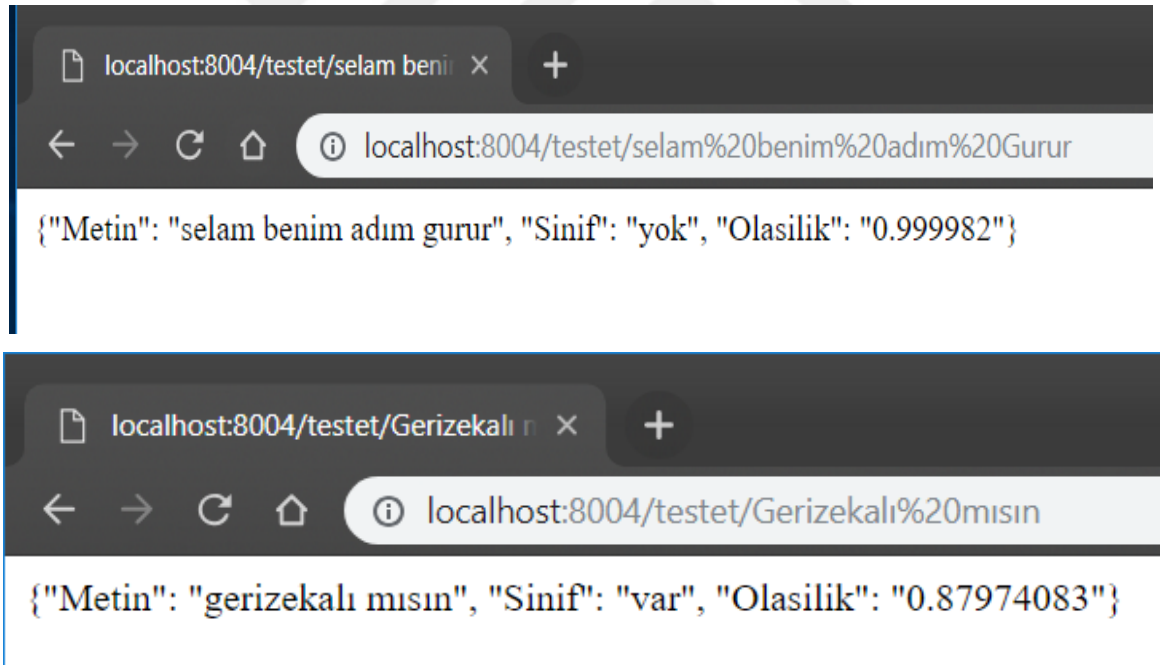
Şekil 4.1: I numaralı veri seti kullanılarak eğitim doğruluk(accuracy) ve kayıp(loss) çıktısı.

Şekil 4.2'de CNN algoritması kullanılarak II numaralı veri seti ile eğitim verilen modelin doğruluk(accuracy) ve kayıp(loss) fonksiyonlarının üretmiş olduğu grafikler gözlemlenmektedir. Grafik üzerinde eğitim onaylama (train validation) noktaları ele alınmıştır. Grafik üzerindeki Y eksenindeki değer doğruluk(accuracy) ve kayıp(loss) değerlerini ifade ederken X eksenindeki değerler devir(epoch) değerini temsil etmektedir.



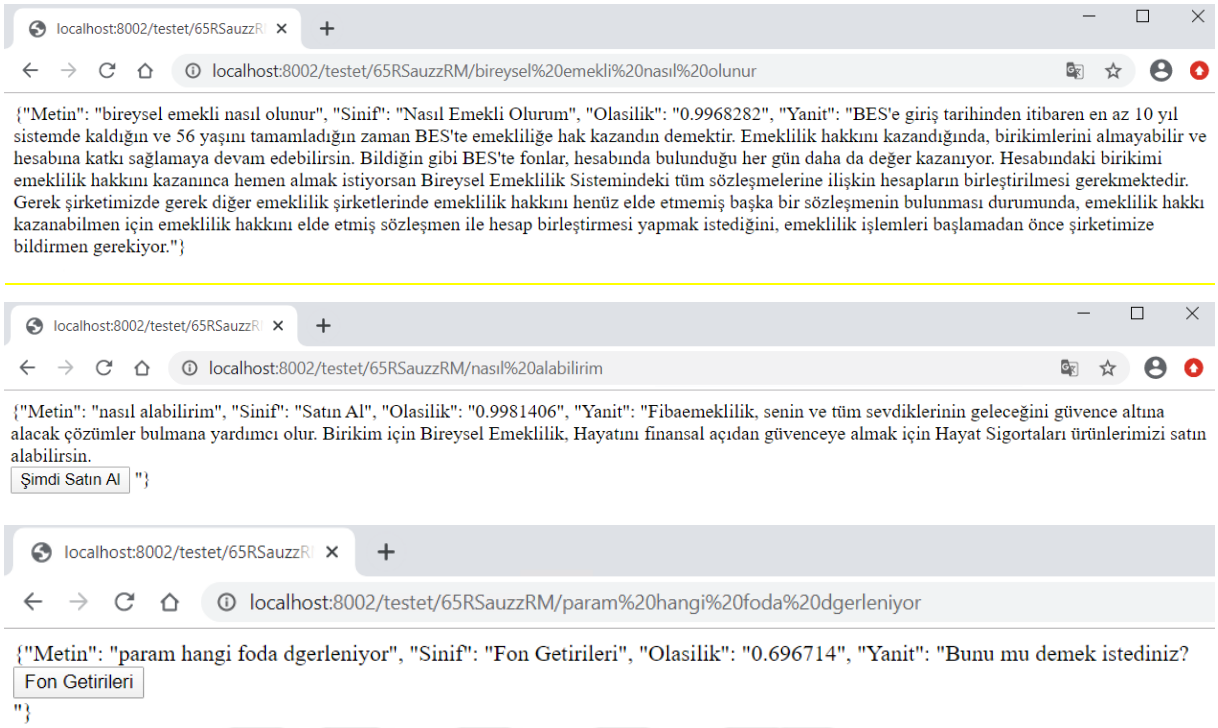
Şekil 4.2: II numaralı veri seti kullanılarak eğitim doğruluk(accuracy) ve kayıp(loss) çıktısı.

I numaralı veri seti ile model eğitiminden sonra Flask kütüphanesi kullanılarak verilen girdi üzerinde modelin oluşturduğu bazı eşleştirmeleri şekil 4.3'te görüntülenmektedir. Bu model yardımı ile girilen cümle üzerinde küfür içerikli veri olup olmadığı kontrol edilmektedir.



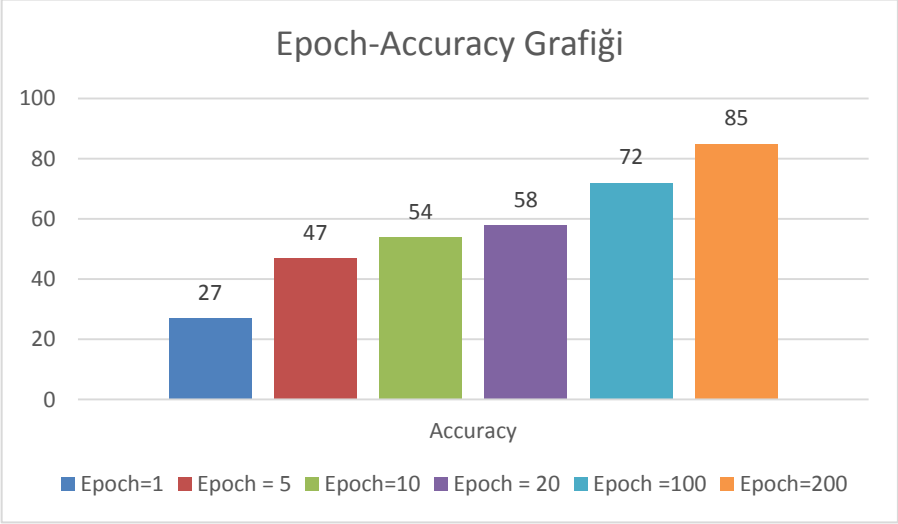
Şekil 4.3: Flask kütüphanesi ile model test edilmesi.

II numaralı veri seti ile verilen girdi üzerinde modelin oluşturduğu çıktıları Şekil 4.4'te verilmiştir. Bu model yardımı ile verilen girdinin emeklilik sınıflarından hangisi ile eşleştiğini, eşleşme olasılığı ve aynı zamanda kullanıcıya oluşturulan cevap görüntülenmektedir.



Şekil 4.4: II numaralı veri seti ile oluşturulan modelin flask ile test edilmesi

Dönem(Epoch) algoritmanın tüm veri kümesindeki örneklerin kaç kez görüntüleneceğini temsil eden sayıdır. Bir dönem veri setinde bulunan bütün örneklerin görüldüğü bir öğrenme döngüsüdür. Şekil 4.5'te verilen grafik üzerinde görüldüğü gibi epoch sayısı arttırıldığı zaman model başarımı yükselmektedir. Bunun sayesinde hiper parametrelerin model başarımına olan etkisi net bir şekilde görülmektedir.



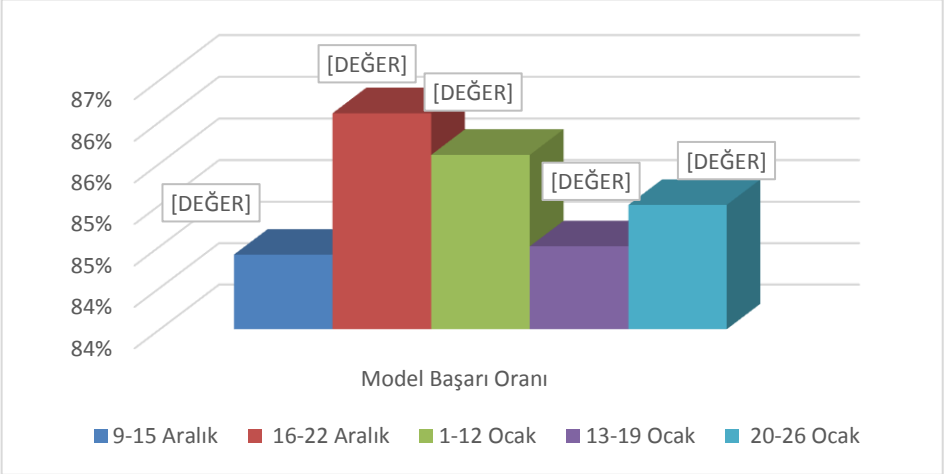
Şekil 4.5: Farklı epoch değerlerinin model üzerindeki etkisi

En uygun dönem(epoch) sayısını bulmak için doğrulama ve eğitim hatası göz önünde bulundurulmalıdır. Doğrulama ve eğitim hatası yükselmeye başladığı noktada modelin aşırı öğrenme probleminin göstergesi olabilir. Dönem(epoch) sayısı yüksek tutulmalı fakat hata değerleri göz önünde bulundurularak en uygun değer seçilmelidir.

Farklı zaman dilimlerinde eğitilen model üzerine gelen cümlelerin sınıflandırılmasında elde etmiş olduğumuz başarı oranları Tablo 4.1 ve Şekil 4.6' da görüntülenmektedir. Bu işlem ile gerçek kullanıcılardan almış olduğumuz girdiler ile modelin başarımını ve kullanım için uygunluğunu gözlemlemekteyiz.

Tablo 4.2: Zaman aralıklı model başarı tablosu

Tarih Aralığı	Toplam Soru Sayısı	Doğru	Yanlış	Başarı Oranı
9-15 Aralık 2019	257	217	40	%84.4
16-22 Aralık	181	156	25	%86.1
1-12 Ocak 2020	237	203	34	%85.6
2019 13-19 Ocak 2020	110	93	17	%84.5
20-26 Ocak 2020	134	114	20	%85



Şekil 4.6: Zaman aralıklı model başarı grafiği.

Modelin başarımını temsil eden faktörlerden birisi veri seti üzerindeki örneklerin sayısıyla ilişkilidir. Bu çalışmada veri seti üzerindeki örneklerin arttırılmasıyla modelin başarı oranının arttığını gözlemledik. Aynı zamanda örneklerin artmasıyla model eğitiminde harcanan sürenin de artışını gözlemledik. Kaynaklar incelenirken farklı dillerde gerçekleştirilen benzer çalışmalarda daha yüksek başarım oranları elde edildiği gözlemlenmiştir. Fakat bu çalışmada Türkçe metinleri ele aldığımız için veri seti eksikliğinden ve veri seti örnek sayısından kaynaklı olarak farklı dillerdeki çalışmalar ile karşılaştırma gerçekleştirilememektedir.

5. TARTIŞMA VE SONUÇ

Günümüzde yapay zekâ alanındaki çalışmaların artması ile insan hayatını kolaylaştıran uygulamalar üretilmiştir. Yapay zekâ uygulamalarının gelişmesindeki en önemli etken büyük veriler üzerinde işlem yapmasıdır. Büyük veriler üzerinde belirli standartlar kullanılıp bu verilerin içerisinde bulunan bilgiler yardımı ile yapay zekâ uygulamaları günlük yaşantımızda çeşitli alanlarda kolaylık sağlamaktadır. Bilim insanlarının çalışmalarda kullanılmak üzere büyük veri setlerinin kullanıma açık olarak paylaşılması gelecekte oluşturulacak yöntemler için yön vericidir.

Derin öğrenme ve makine öğrenmesi yöntemleri metin sınıflandırma işleminde önemli başarılara yol açmıştır. Metin sınıflandırma işlemi sıklıkla ihtiyaç duyulan problemlerden biridir. Son yıllarda, özellikle bankacılık ve benzer sektörlerde ihtiyaç haline gelen Türkçe otomatik cümle sınıflandırma akıllı sistemleri büyük bir ihtiyaç haline gelecektir [34].Özellikle çağrı merkezi bulunan sektörlerde otomatik cümle sınıflandırma sistemine ihtiyaç duyulmaktadır. Bu sebepten dolayı bankacılık ve benzer sektörlerde sanal asistan sistemlerinin yaygınlaşmasına yol açmıştır. Çağrı merkezi personel maliyeti ve ihtiyacını azaltarak personelin daha verimli işlemler yapmasını sağlamaktadır. Muhtemel olarak gelecekte tamamen otomatik olarak sanal asistan ile işlemlerimizi tamamlayacağız.

Bu tez çalışmasında Türkçe metinler için cümle sınıflandırması temel olarak ele alınmıştır. Çalışmada ele alınan Türkçe metinler bir emeklilik bankasının müşteri ilişkileri personel kayıtlarından oluşturulmuştur. Sınıflandırma yapılan kategoriler veri setine bağlı olarak emeklilik işlemleri ile ilgili kategorilerden oluşmaktadır. Türkçe dilinde hali hazırda sınıflandırma yapabilmek için bir veri seti bulunmadığı için veri seti oluşturulmuştur.

Bu tez çalışması sayesinde Türkçe dilinde oluşturulan veri setinin, derin öğrenme ve makine öğrenmesi algoritmaları arasında hangi algoritmanın en başarılı olacağı incelenmiştir. Veri seti algoritmalarda kullanılmadan önce ön işlemden geçirilmiştir. Elde edilen sonuçların başarı oranları bir biri ile karşılaştırılmıştır. Model eğitiminde kullanılan algoritmaların

harcadıkları süreler karşılaştırılmıştır. Bu nedenle tez çalışmasında önerdiğimiz yöntem, oluşturduğumuz veri setinde en büyük başarı elde eden yöntemdir.

Tez çalışmasında elde edilen sonuçlar doğrultusunda veri setinde bulunan örnek sayısının modelin başarımını etkilediği gözlemlenmiştir. Model başarımını etkileyen bir diğer faktör de kategori sayısıdır. Kategori sayısının az olması, bir kategoriyi temsil eden örnek sayısını etkilediği için model başarımını doğrudan etkilemektedir. Aynı zamanda veri setinde bulunan örneklerin tekrar etmemesi, aynı örneğin birden fazla kategori ile eşleştirilmemesi de önemli unsurlardan biridir.

Bu çalışma üzerinde gelecekte yapılabilecek bir kategori azaltma çalışması mümkün olabilir. Kategori sayısının azaltılması ile başarımın daha da yükselmesi gerektiğini ön görmekteyiz. Kategori azaltması çalışmasının haricinde veri setinin genişletilmesi ve örnek sayısının arttırılması çalışması ile de başarım oranının artacağını ön görmekteyiz. Aynı zamanda elde edilen bulgular ile Türkçe veri seti ile yapılan benzer metin tabanlı çalışmalar ile karşılaştırma yapılabilir.

Hesaplama süresinin azaltılması ve büyük verinin daha az depolama alanı harcaması için dijital veri unutma (digital data forgetting) işlemi gelecekte bu çalışmaya uygulanabilir [35]. Bu işlem yardımı ile daha az depolama alanı kullanmış olup hesaplama işlemlerinde hızlı sonuçlar elde edileceği ön görülmektedir.

KAYNAKLAR

[1]. LeCun, Y., Bengio, Y. and Hinton, G., 2015 Deep learning. Nature 521, 436–444, doi:10.1038/nature14539

[2]. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, 1998, Gradient-based learning applied to document recognition, Published in Proceedings of the IEEE , vol. 86, no. 11, pp. 2278-2324.

[3]. R. Hecht-Nielsen, 1989, Theory of the backpropagation neural network,International Joint Conference on Neural Networks, IEEE, New York, Vol. 1, 1989, pp. 593-605, doi:10.1109/IJCNN.1989.118638.

[4]. Jiuxiang, G., Zhenhua, W., 2015, Recent Advances in Convolutional Neural Networks, Published in Pattern Recognition 2015, doi:10.1016/j.patcog.2017.10.013.

[5]. Silva, J., Coheur, L., Mendes, A. C. And Wicher, A., 2011, From symbolic to sub-symbolic information in question classification, Artificial Intelligence Review, Volume 35, Issue 2, pp 137–154.

[6]. Wang, S. And Manning, C., 2012, Baselines and Bigrams: Simple, Good Sentiment and Topic Classification, Published in Association for Computational Linguistics (ACL). Pages 90-94.

[7]. Yoon, K., 2014, Convolutional Neural Networks for Sentence Classification, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. Pages 1746–1751, doi: 10.3115/v1/D14-1181.

[8]. Socher, R., Huval, B., Manning, C. And Ng, A., 2012, Semantic compositionality through recursive matrix-vector spaces, Association for Computational Linguistics (ACL), D12-1110

[9]. Johnson, R., Zhang, T., 2016, Supervised and Semi-Supervised Text Categorization using LSTM for Region Embeddings, ICML'16 Proceedings of the 33rd International Conference on Machine Learning - Volume 48, Pages 526-534

[10]. LeCun, Y., Bengio, Y., 1998, Convolutional Networks for Images, Speech and Time-Series, Published in The handbook of brain theory and neural networks, MIT Press Cambridge, MA, USA, ISBN:0-262-51102-9 , Pages 255-258 1998.

[11]. Jacovi, A., Shalom, O. S., Goldberg,Y., 2018 Understanding Convolutional Neural Networks for Text Classification, Published in Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, Pages: 56–65, doi:10.18653/v1/W18-5408.

[12]. Furbush, J., 2018, Machine learning: A quick and simple definition, https://www.oreilly.com/ideas/machine-learning-a-quick-and-simple-definition, [Ziyaret Tarihi: 4 Aralık 2019].

[13]. Heidenreich , H., 2018, An analysis of the types of machine learning written for the average person , https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f , [Ziyaret Tarihi: 4 Aralık 2019].

[14]. McCallum, A., Nigam, K., 1998, A comparison of event models for naive bayes text classification, Published in Association for the Advancement of Artificial Intelligence, Pages 41-48.

[15]. Liu, B., Hao, Z. And Yang, X., 2005, Nesting support vector machinte for muti-classification, International Conference on Machine Learning and Cybernetics 2005, Guangzhou, China, China, ISBN: 0-7803-9091-1, doi: 10.1109/ICMLC.2005.1527678.

[16]. Rifkin, R., Klautau, A., 2004, In Defense of One-Vs-All Classification, Journal of Machine Learning Research, ISBN: 1532-4435, Vol, 5, pp. 101-141.

[17]. Pupale, R., 2018, Support Vector Machines(SVM) — An Overview, https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989,[Ziyaret Tarihi: 8 Aralık 2019].

[18]. Prabhat, A., Khullar, V., 2017, Sentiment classification on big data using Naïve bayes and logistic regression, 2017 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, Electronic ISBN: 978-1-4673-8855-9, doi: 10.1109/ICCCI.2017.8117734

[19]. Ramadhan, W., Novianty, S., Setianingsih, S., 2017, Sentiment analysis using multinomial logistic regression, 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), Yogyakarta, Indonesia, Electronic ISBN: 978-1-5386-1667-3, doi: 10.1109/ICCEREC.2017.8226700

[20]. Behadada, O., Trovati, M., MA, C., Bessis, N., Korkontzelos, Y., 2016, Logistic Regression Multinomial for Arrhythmia Detection, 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, Electronic ISBN: 978-1-5090-3651-6, DOI: 10.1109/FAS-W.2016.39

[21]. Goldberg, Y., 2017, Neural Network Methods for Natural Language Processing , Morgan & Claypool, Synthesis Lectures on Human Language Technologies, Electronic ISBN: 9781627052955, doi: 10.2200/S00762ED1V01Y201703HLT037

[22]. Bai, X., 2018, Text classification based on LSTM and attention, 2018 Thirteenth International Conference on Digital Information Management (ICDIM), Berlin, Germany, Electronic ISBN: 978-1-5386-5244-2, doi: 10.1109/ICDIM.2018.8847061

[23]. Olah, C., 2015, Understanding LSTM Networks, https://colah.github.io/posts/2015-08-Understanding-LSTMs/, [Ziyaret Tarihi: 12 Aralık 2019].

[24]. Luan, Y., Lin, S., 2019, Research on Text Classification Based on CNN and LSTM, 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, Electronic ISBN: 978-1-7281-1223-7,doi: 10.1109/ICAICA.2019.8873454

[25]. Niepert, M., Ahmed, M., Kutzkov, K., 2016, Learning Convolutional Neural Networks for Graphs, ICML'16 Proceedings of the 33rd International Conference on International Conference on Machine Learning, New York, NY, USA, Pages 2014-2023

[26]. Şeker, A., Diri, B., Balık, H. H., 2017, Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme, Gazi Mühendislik Bilimleri Dergisi, vol:3, sayfa: 47-64

[27]. Wallach, I., Dzamba, M., Heifets, A., 2015, AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery, https://arxiv.org/abs/1510.02855

[28]. Bayramlı, B., Evrişimsel Ağlar, 2019, Derin Öğrenim (Convolutional Nets -Convnet-, Deep Learning), https://burakbayramli.github.io/dersblog/algs/convnet/evrisimsel_aglar_derin_ogrenim_ _convolutional_nets_convnet_deep_learning_.html [Ziyaret Tarihi: 15 Aralık 2019].

[29]. Sharma, S., Activation Functions in Neural Networks, 2017, https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 [Ziyaret Tarihi: 16 Aralık 2019].

[30]. Wang, S., Tang C., Sun, J., 2018, Multiple Sclerosis Identification by 14-Layer Convolutional Neural Network With Batch Normalization, Dropout, and Stochastic Pooling, Published in Frontiers in Neuroscience 12: 818, doi:10.3389/fnins.2018.00818

[31]. Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014, Rich feature hierarchies for accurate object detection and semantic segmentation, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, Pages 580-587, ISBN: 978-1-4799-5118-5, doi:10.1109/CVPR.2014.81.

[32]. Mikalov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J., 2013, Distributed representations of words and phrases and their compositionality, NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, Lake Tahoe, Nevada, Pages:3111-3119.

[33]. Buyukkinaci, M., 2018, Word2Vec Nedir, https://medium.com/@muhammedbuyukkinaci/word2vec-nedir-t%C3%BCrk%C3%A7e-f0cfab20d3ae, [Ziyaret Tarihi: 23 Aralık 2019].

[34]. Pirana, G., Sertbaş, A., Ensari, T., 2019, Sentence Classification with Deep Learning Method For Virtual Assistant Applications, IEEE 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, pp. 1-5, doi: 10.1109/ISMSIT.2019.8932888.

[35]. Gunay, M., Yildiz, E., Nalcakan, Y., Asiroglu, B., Zencirli, A., Mete, B. R., Ensari, T.,2018, Digital Data Forgetting: A Machine Learning Approach, IEEE International Symposium on Multidisciplinary Studies and Innovative Technologies, Ankara, Turkey, pp. 1-4, doi: 10.1109/ISMSIT.2018.8567046.

[36]. Amidi, A., Amidi, S., Derin Öğrenme El Kitabı (Stanford CS 230), https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-convolutional-neural-networks, [Ziyaret Tarihi: 04 Şubat 2020].

ÖZGEÇMİŞ

Kişisel Bilgiler	
Adı Soyadı	Gurur PİRANA
Doğum Yeri	PRİZREN/KOSOVA
Doğum Tarihi	02.03.1995
Uyruğu	❒ T.C. ☑ Diğer: Kosova
Telefon	5077193888
E-Posta Adresi	gururpirana@gmail.com
Web Adresi	https://tr.linkedin.com/in/gururpirana

Eğitim Bilgileri	
Lisans	
Üniversite	Trakya Üniversitesi
Fakülte	Mühendislik Fakültesi
Bölümü	Bilgisayar Mühendisliği
Mezuniyet Yılı	2017

Yüksek Lisans	
Üniversite	İstanbul Üniversitesi-Cerrahpaşa
Enstitü Adı	Lisansüstü Eğitim Enstitüsü
Anabilim Dalı	Bilgisayar Mühendisliği Anabilim Dalı
Programı	Bilgisayar Mühendisliği Programı

Makale ve Bildiriler
Pirana, G., Sertbaş, A., Ensari, T., 2019, Sentence Classification with Deep Learning Method For Virtual Assistant Applications, IEEE 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, pp. 1-5