



T.C.

ONDOKUZ MAYIS ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME ÇALIŞMASI

SOSYAL MEDYA ÜZERİNDE DUYGU ANALİZİ
Tweety

Hazırlayan

14060304 - Cafer Ümit Salman

14060292 - Tunahan Yetimoğlu

16061624 - Erdem Şahin Uslu

Danışman

Dr.Öğr.Üyesi – İsmail İŞERİ

Mayıs/2019

SAMSUN

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ.....	iii
ÇİZELGE LİSTESİ.....	v
KISALTMA LİSTESİ	vi
ÖZET	vii
BÖLÜM BİR - GİRİŞ.....	8
1.1 AMAÇ	8
1.2 LİTERATÜR ARAŞTIRMASI	9
BÖLÜM İKİ - KULLANILAN ARAÇ ve YÖNTEMLER	11
2.1 Veri Seti.....	11
2.2 Yapay Sinir Ağları.....	15
2.2.1 Öğrenme Yöntemleri.....	15
2.2.1.1 Danışmanlı (Supervised) Öğrenme	16
2.2.1.2 Danışmansız (Unsupervised) Öğrenme	16
2.2.1.3 Pekiştirmeli (Reinforcement) Öğrenme	16
2.2.2 RNN Ağları	16
2.2.3 LSTM.....	17
2.3 NLTK	22
2.4 Kök Bulma (Stemming)	22
2.5 Lemmatizasyon	23
2.6 SpellChecker	24
2.7 Veri Ön İşleme	24
2.8 Python Programlama Dili	25
2.9 Keras.....	26
2.10 TensorFlow.....	26
2.11 Docker	26
2.12 IBM POWER9	27
2.13 Plotly	27
2.14 Jupyter	27
2.15 Pandas.....	27
2.16 Numpy	28
2.17 Matplotlib	28
BÖLÜM ÜÇ - PROJENİN GERÇEKLENMESİ	29
3.1 Twitter API Kullanımı için Gereksinimler	29
3.2 python-twitter Kütüphanesi ile Kullanıcı Tweet'lerinin Elde Edilmesi	31
3.3 Elde Edilen tweet'lerin Doğal Dil İşleme Adımlarından Geçirilmesi	33
3.4 Kullanılan Yapay Sinir Ağının Yapısı.....	34

3.4.1 Adam (Adaptive Moment Estimation) Optimizasyon Algoritması	35
3.4.1.1 Adam Algoritmasının Çalışması	36
3.5. Uygulama Arayüzü.....	37
BÖLÜM DÖRT - DEĞERLENDİRME ve SONUÇ	39
4.1 Yapay Sinir Ağı İyileştirmeleri	39
4.2 Sonuç	40
KAYNAKÇA.....	45
ÖZGEÇMİŞLER	46

ŞEKİL LİSTESİ

ŞEKİL 2.1 “Fun” etiketindeki kelimelerin boyut/frekans gösterimi	12
ŞEKİL 2.2 “Sadness” etiketindeki kelimelerin boyut/frekans gösterimi	12
ŞEKİL 2.3 “Happiness” etiketindeki kelimelerin boyut/frekans gösterimi	13
ŞEKİL 2.4 “Neutral” etiketindeki kelimelerin boyut/frekans gösterimi	13
ŞEKİL 2.5 Kelimelerin veri setindeki kullanım sayısı	14
ŞEKİL 2.6 Kelimelerin kullanıldığı tweet sayısı	14
ŞEKİL 2.7 Temel yapay sinir ağı hücresi	15
ŞEKİL 2.8 RNN ağı mimarisi	17
ŞEKİL 2.9 Hiperbolik tanjant fonksiyonu grafiği	18
ŞEKİL 2.10 RNN hücreler zinciri	18
ŞEKİL 2.11 Sigmoid fonksiyonu grafiği	19
ŞEKİL 2.12 LSTM hücreler zinciri	19
ŞEKİL 2.13 Cell State Line	20
ŞEKİL 2.14 LSTM Forget Gate	20
ŞEKİL 2.15.LSTM Input Gate	21
ŞEKİL 2.16 LSTM Cell Gate	21
ŞEKİL 2.17 LSTM Output Gate.....	22
ŞEKİL 2.18 Python programlama dili logosu	25
ŞEKİL 2.19 Keras logosu.....	26
ŞEKİL 2.20 Tensorflow logosu.....	26
ŞEKİL 2.21 Docker logosu.....	26
ŞEKİL 2.22 IBM POWER9 logosu.....	27
ŞEKİL 2.23 Plotly logosu.....	27
ŞEKİL 2.24 Jupyter logosu.....	27
ŞEKİL 2.25 Pandas logosu.....	27
ŞEKİL 2.26 NumPy logosu.....	28
ŞEKİL 2.27 Matplotlib logosu.....	28
ŞEKİL 2.28 Twitter API logosu.....	28

ŞEKİL 3.1 Twitter API uygulama oluşturma ekranı	30
ŞEKİL 3.2 Key ve Token değerlerinin elde edilmesi	31
ŞEKİL 3.3 Sinir ağı modeli	34
ŞEKİL 3.4 Softmax fonksiyonu grafiği	35
ŞEKİL 3.5 Arayüz tasarımı -1	37
ŞEKİL 3.6 Arayüz tasarımı -2	38
ŞEKİL 4.1 5-Fold ile veri setinin bölünmesi	40
ŞEKİL 4.2 Eğitim-test aşaması için 3- Fold	41
ŞEKİL 4.3 Eğitim-test aşaması için 5-Fold	42
ŞEKİL 4.4 Eğitim-test aşaması için 10-Fold	43
ŞEKİL 4.5 K-Fold validasyon başarımlarını karşılaştırması	44
ŞEKİL 4.6 K-Fold başarımlarını karşılaştırması	44

ÇİZELGE LİSTESİ

ÇİZELGE 2.1 Veri setinden örnekler	11
ÇİZELGE 2.2 Stemming’de karşılaşılan hatalara örnek	23
ÇİZELGE 2.3 Lemmatizasyon örneği	23
ÇİZELGE 2.4 SpellChecker örneği	24
ÇİZELGE 2.5 Örnek veri ön işleme.....	25
ÇİZELGE 3.1 Emoji-Regex dönüşüm tablosu	33

KISALTMA LİSTESİ

LSTM: Long Short-Term Memory
YSA: Yapay Sinir Ağı
RNN: Recurrent Neural Network
CNN: Convolutional Neural Network
AI: Artificial Intelligence
API: Application Programming Interface
NLTK: Natural Language Toolkit
CNTK: Cognitive Toolkit
RMSProp: Root Mean Square Propagation
AdaGrad: Adaptive Gradient Algorithm
Adam: Adaptive Moment Estimation
GPL: General Public Licence
NLP: Natural Language Processing
CPU: Central Processing Unit
GPU: Graphics Processing Unit

ÖZET

Duygu analizi, metin madenciliğinin önemli bir alanı ve son yılların önemli araştırma konularından biridir. Günümüzde sosyal medya kullanıcıları belirli bir konudaki görüşlerini, hislerini sosyal medya üzerinden kolay ve hızlı bir biçimde aktarabilmektedir. Bu da kullanıcıların görüşlerini simgeleyen verilerin sayısının bir hayli artmasına sebep olmaktadır ve bu çok sayıdaki verinin işlenmesiyle önemli çıkarımlar yapmak mümkündür. Fakat böyle büyük çaptaki bir veri kümesinin insan eliyle incelenip bir çıkarım elde edilmesi çok zahmetlidir. Bu durumda duygu analizi kapsamında kullanılacak materyal ve metotlarla bu çıkarımların çok daha kolay ve hızlı bir şekilde yapılması mümkün olmuştur.

Duygu analizi ile elde edilebilecek çıkarımların çeşitliliğine paralel olarak bazı çalışma alanları doğmuştur. Bunlara örnek olarak ticari kuruluşların müşterilerinin satın alma potansiyeli hakkında bilgi edinebilmesi, sosyal bilimcilerin bir toplumun yaşam kalitesi hakkında bilgi edinebilmesi, politik gönderilerden yola çıkarak seçim sonuçlarının önceden tahmin edilebilmesi, sağlık çalışanlarının hastalarının duygu durumu hakkında bilgi edinebilmesi gösterilebilir.

Anahtar kelimeler: NLP (Natural Language Processing), LSTM (Long-Short Term Memory), RNN(Recurrent Neural Network) Tokenization, Lemmatization, Stemming, Twitter, Sentiment Analysis (Duygu Analizi)

BÖLÜM BİR - GİRİŞ

İnternet ağının hızla büyümesi ile beraber yeni iletişim araçları ortaya çıkmıştır ve literatüre bu internet platformlarını tanımlayan "sosyal medya" kavramı katılmıştır. Bu iletişim araçlarından biri olan Twitter ise, kişilerin duygu ve düşüncelerini açıkça ve kolay biçimde paylaşabildiği en önemli platform haline gelmiştir. Böylece Twitter platformu, duygu analizi alanında çalışmak isteyenler için biçilmiş kaftan haline almıştır. Bu durum, proje kapsamında üzerinde çalışacağımız platform olarak Twitter'ı seçmemizdeki en önemli sebeptir. Twitter'ın kullanıcılara sunduğu kullanışlı API servisi sayesinde, yazılım yardımıyla analiz için gerek duyacağımız paylaşımları elde etmek kolay ve hızlı bir hal almıştır. Tüm bunların yanı sıra doğal dil işleme kütüphanelerinin gelişmesi ve sınıflandırma işlemleri için tasarlanmış yapay sinir ağı modellerinin varlığı da duygu analizinin gerçekleşmesi konusunda geliştiricilere kolaylıklar sağlamıştır. Ayrıca sinir ağı modellerinin eğitimi için ihtiyaç duyulacak yüksek işlem gücüne sahip bilgisayarların da ücretsiz olarak geliştiricilerin kullanımına sunulması işleri bir hayli kolaylaştırmaktadır.

1.1 AMAÇ

Bu çalışmada Twitter kullanıcılarının paylaşımları analiz edilecektir ve belirlenen 5 duygu sınıfı için (korku, mutluluk, nötr, üzüntü, eğlence) sınıflandırılacaktır. İlk olarak, Twitter platformunda yapılan metinsel paylaşımlar API servisi yardımıyla elde edilecektir. Bu metinsel ifadeler Doğal Dil İşleme(DDİ) adımlarından geçirilerek öznitelik çıkarımı yapılacaktır. Öznitelik çıkarımı yapılan ifadeler, eğitilmiş yapay sinir ağı modeli ile duygusal olarak sınıflandırılacaktır.

Yapay sinir ağı modelini oluşturmak için farklı sinir ağı modelleri denenecek ve elde edilen başarımlar oranları karşılaştırılarak optimize model seçilecektir. Modellerin parametreleri deneme-yanılma yöntemiyle belirlenecek olup, optimize sonuç elde edilen

parametre deęerleri saklanacak ve bu deęerler ile oluřturulacak olan aę modeli kaydedilecektir. Eęitimlerin hızlı bir bięimde tamamlanması iin iřlem gc yksek bir makineye ihtiya duyulmaktadır. Kullanılan makinenin zellikleri ilerleyen blmlerde anlatılmaktadır.

Modelin eęitimi iin gerekli olan veri seti, istedięimiz beř duygu sınıfını ierecek řekilde dzenlenerek sinir aęının eęitimi iin kullanılacaktır. Son olarak sinir aęından elde edilen analiz sonuları, web uygulaması aracısıyla kullanıcıya grafiksel olarak sunulacaktır.

1.2 LİTERATR ARAřTIRMASI

Ladislav Lenc ve Tomáš Hercig, duygu analizinde İngilizce’de ok iyi sonular veren sinir aęı alıřmalarından ilham alarak slav dillerinden biri olan eke’ye uyarladılar ve bu alanda ilk alıřmalardan birini gerekleřtirdiler. İngilizce duygu analizindeki mevcut en modern yntemleri tespit edip karřılařtırma yapmak iin İki ingiliz řirket ile alıřtılar. Bu alıřmalarında sadece Ynelime Dayalı Duygu Analizi (YDDA) hedefinden ziyade, duygu analizinin genel amacı olan eřitli seviyelerde (metinler, cmlerler, bakıř aısı) kutupluluk tespitine odaklandılar. alıřmalarını Terim Ekstraksiyonu (TE), Terim Polaritesi (TP), Kategori Ekstraksiyonu (KE) ve Kategori Polaritesi (KP) gibi alt grevler řeklinde blmlendirdiler. alıřmalarında eke ve İngilizce iin belge seviyesinde duygu polaritesine (TP, KP) odaklandılar ve nceki sonular ile sinir aęı mimarisi kullanımını karřılařtırarak aradaki farkı lmeyi amaladılar. ncelikli olarak eke zerinde dil bilgisi arařtırmalarında bulundular. Ladislav ve Tomáš, duygu analizinde Yarı Denetimli ęrenme aęlarından olan Aktif Derin Aęları kullanımını ermektedir. Bu aę Kısıtlı Boltzmann Makinesine dayanmaktadır. Sinir aęı mimarisinin implementasyonu iin Theano derin ęrenme ktphanesi tabanlı Keras ktphanesini kullandılar ve makul hesaplama zamanları iin GPU gcnden faydalandılar. Veri olarak Facebook gnderileri ve ekoslovakya Film Veritabanı kaynaklarını kullandılar. Veri setini (belge, cmlerler) nce NLP (Doęal Dil İřleme) iřlemine tabii tuttular ve dzenlediler. Sonrasında Yoon Kim tarafından nerilen Konvansiyonel Sinir Aęı mimarisi oluřturdular. Aęlarını eęitmek ve doęrulamak amacıyla 10 Katlamalı apraz Doęrulama ile eęitim/test veri setini

ayırdılar. Bu çalışma sonucunda yapılan testlerde %70-80 oranlarında başarımları elde ettiler (Lenc ve Hercig, 2016).

Samir Tartir ve Ibrahim Abdul-Nabi, 422 milyon kişi (lehçeleriyle beraber) tarafından konuşulan Arapça dili ve çeşitli lehçeleri üzerinde Twitter gönderileri ile çalışırılığını test etmek üzere arapça sosyal medya paylaşımlarında duygu analizi için Arapça Duygu Ontolojisi üzerinde çalıştılar. Veri için Twitter sosyal paylaşım ağını kullanmaya karar verdiler ve veri seti oluşturmak amacıyla Twitter API'sini kullanarak ayrıca bir crawler geliştirmek yerine farklı kullanıcı ve hashtag gönderileri için Tweet Archivist platformunu kullandılar. Daha sonra spam ve ilgisiz gönderileri filtrelediler. Bir sonraki aşamada ise sosyal medya platformlarında kullanılan gayri resmi ve dil bilgisi kurallarından büyük ölçüde yoksun yazım tarzının, sınıflandırmayı minimum düzeyde etkilemesi için farklı arapça lehçelerini de destekleyen mo3jam (sözlük, arap dil platformu) veritabanını kullandılar. Gönderiler üzerindeki anlamlı bilgi taşıyan arapça kelimeleri -5/5 arasında ağırlıklandırdılar. Gönderi duyarlılığı için karar ağacı yapısı altında pozitif, negatif ve nötr şeklinde üçlü bir sınıflandırma kullandılar. %70-80 oranlarında başarımları elde ettiler (Tartir ve Abdul-Nabi, 2016).

Marouane Birjalia, Abderrahim Beni-Hssanea ve Mohammed Erritali çalışmalarında duygu analizini taban alarak intihara eğilimli bireylerin Twitter gönderileri ile tespit edilmesini amaçlamıştır. Birçok intihara eğilimli birey, bu girişimden önce sosyal paylaşım platformlarındaki gönderilerine duygusal kişiliklerini yansıtır ve otomatik tespit edilmeleri bu çalışma ile mümkündür. Çalışmalarında veri seti (twitter gönderileri) için Twitter4J kütüphanesini ve Twitter API'sini kullanarak bir crawler geliştirdiler WordNet ingilizce sözlük veritabanını kullanarak Leacock and Chodorow tabanlı semantik analiz gerçekleştirdiler. İntihar riskine sahip olan ve olmayan şeklinde ikili sınıflandırma kullandılar. Weka makine öğrenmesi aracını kullanarak farklı algoritmalar üzerinde başarımları testi yaptılar. Bu algoritmalar IB1, J48, CART, SMO ve Naive Bayes'dir. İlgili algoritmaların uygulanması sonucunda elde edilen başarımları yaklaşık olarak %75-85'dir (Birjalia, vd., 2017).

BÖLÜM İKİ - KULLANILAN ARAÇ ve YÖNTEMLER

Bu bölümde proje kapsamında kullanılan veri seti ve yapay sinir ağı modeli, özellikleri ve detayları ile birlikte açıklanmaktadır. Ayrıca kullanılan programlama dili ve bu dile ait ihtiyaç duyulan kütüphaneler, araçlar; model eğitiminin gerçekleştirildiği yüksek işlem kapasiteli bilgisayar ve diğer kullanılan teknolojiler hakkında bilgiler yer almaktadır.

2.1 Veri Seti

Projede kullanılmakta olan veri seti “Emotion in Text” adıyla ücretsiz olarak paylaşılmaktadır (Eight, 2016). Yaklaşık 40,000 satır veriden oluşmaktadır. Veri setinde 4 adet sütun bulunmaktadır ve bunlar “tweet_id”, “sentiment”, “author”, “content” şeklindedir. Çizelge 2.1’de veri setinde bulunan bazı tweet’ler gösterilmektedir:

Çizelge 2.1 Veri setinden örnekler

Tweet_id	Sentiment	Author	content
1956969456	Neutral	feinyheiny	cant fall asleep
1956970860	Surprise	Okiepeanut93	Got the news
1956976371	worry	babyxj	I need skott right now
1957005279	Sadness	PYTDavis	I miss my puppy

Veri setinde 13 farklı duygu(sentiment) sınıfı bulunmaktadır. Proje kapsamında ihtiyaç duyulan 5 duygu sınıfını barındıracak şekilde düzenlenerek sinir ağıının eğitilmesinde kullanılmaktadır.

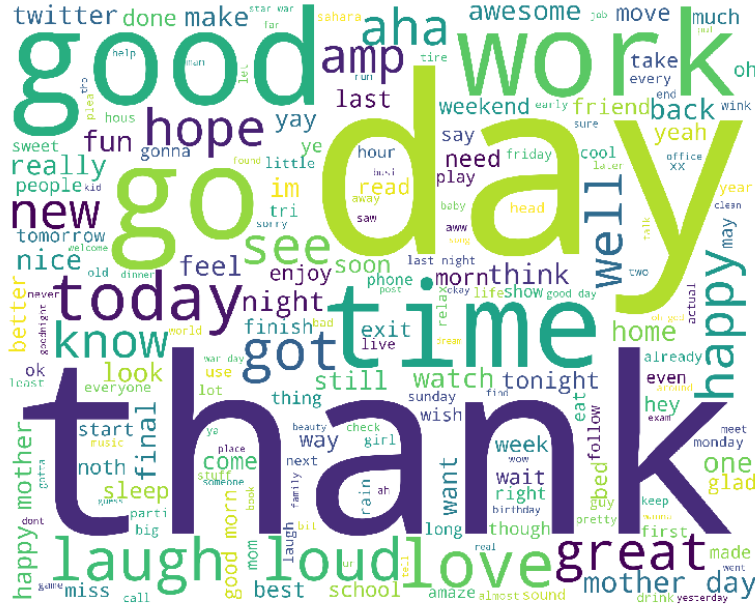
[illegible]

Şekil 2.2’de veri setindeki “Sadness” etiketine sahip tweetlerin içerisindeki kelimelerin boyut/frekans oranı gösterilmektedir. “work, miss, sad, hate” kelimeleri ön plana çıkmaktadır.



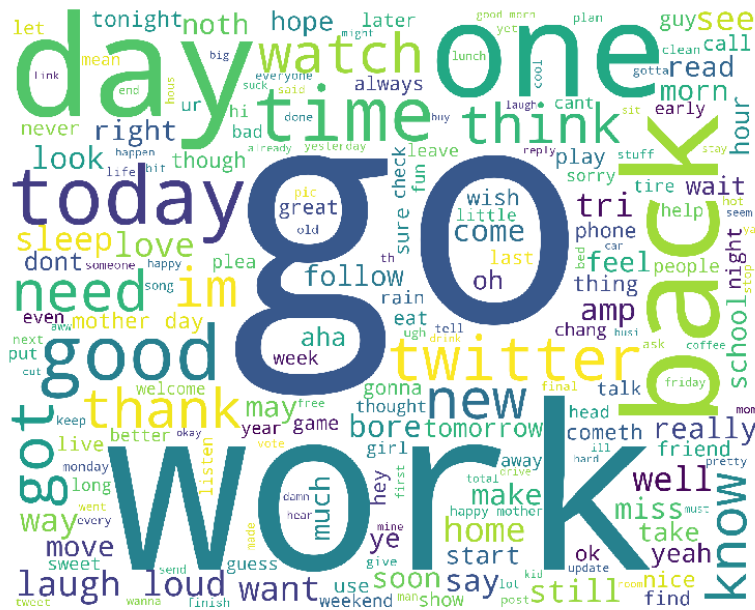
12

Şekil 2.3'te veri setindeki "Happiness" etiketine sahip tweetlerin içerisindeki kelimelerin boyut/frekans oranı gösterilmektedir. "laugh, love, good" kelimeleri ön plana çıkmaktadır.



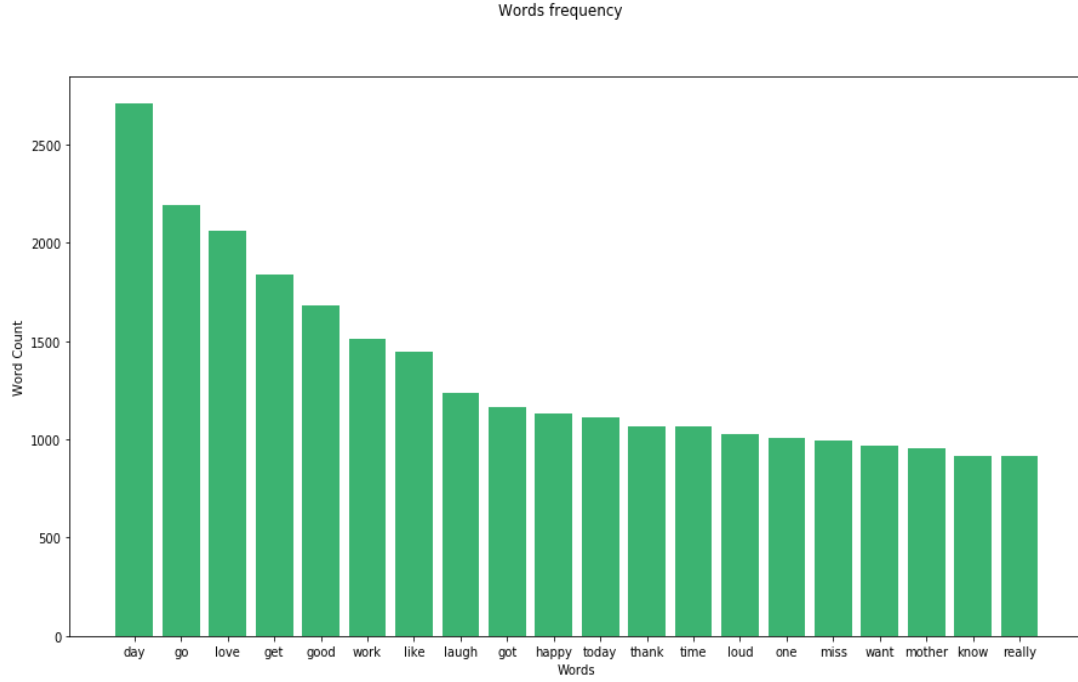
Şekil 2.3 "Happiness" etiketindeki kelimelerin boyut/frekans gösterimi

Şekil 2.4'te veri setindeki "Neutral" etiketine sahip tweetlerin içerisindeki kelimelerin boyut/frekans oranı gösterilmektedir. Genel olarak nötr duygular ön plana çıkmaktadır.



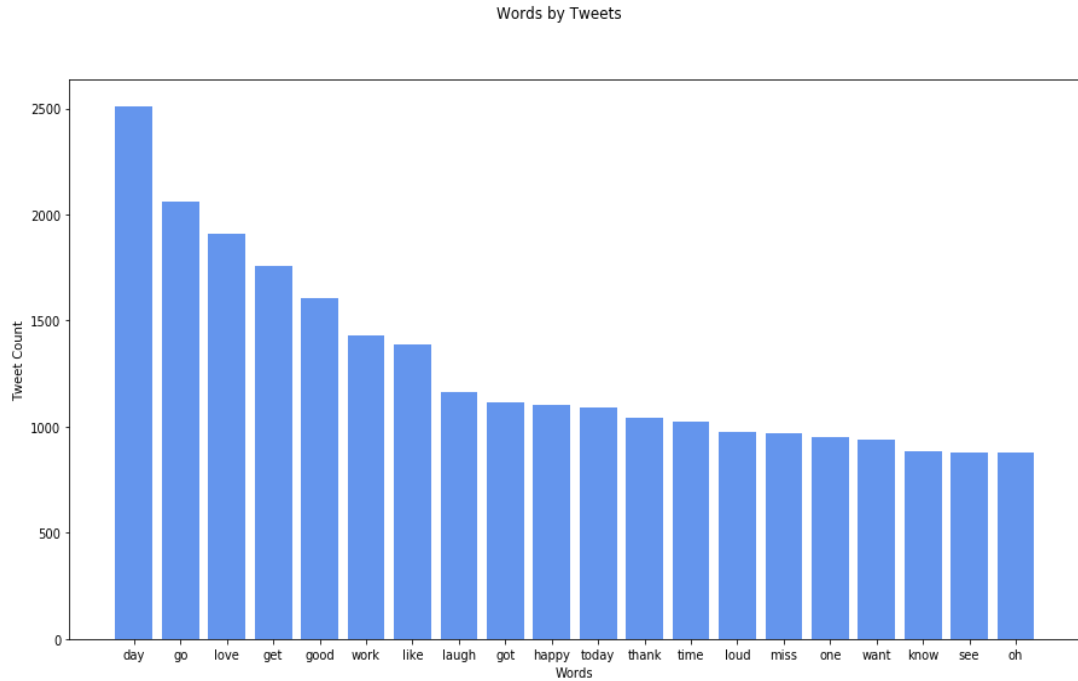
Şekil 2.4 "Neutral" etiketindeki kelimelerin boyut/frekans gösterimi

Şekil 2.5'te veri setinde en çok bulunan 20 kelimenin tweetler içerisinde kaç defa tekrar ettiğini gösteren tablo yer almaktadır.



Şekil 2.5 Kelimelerin veri setindeki kullanım sayısı

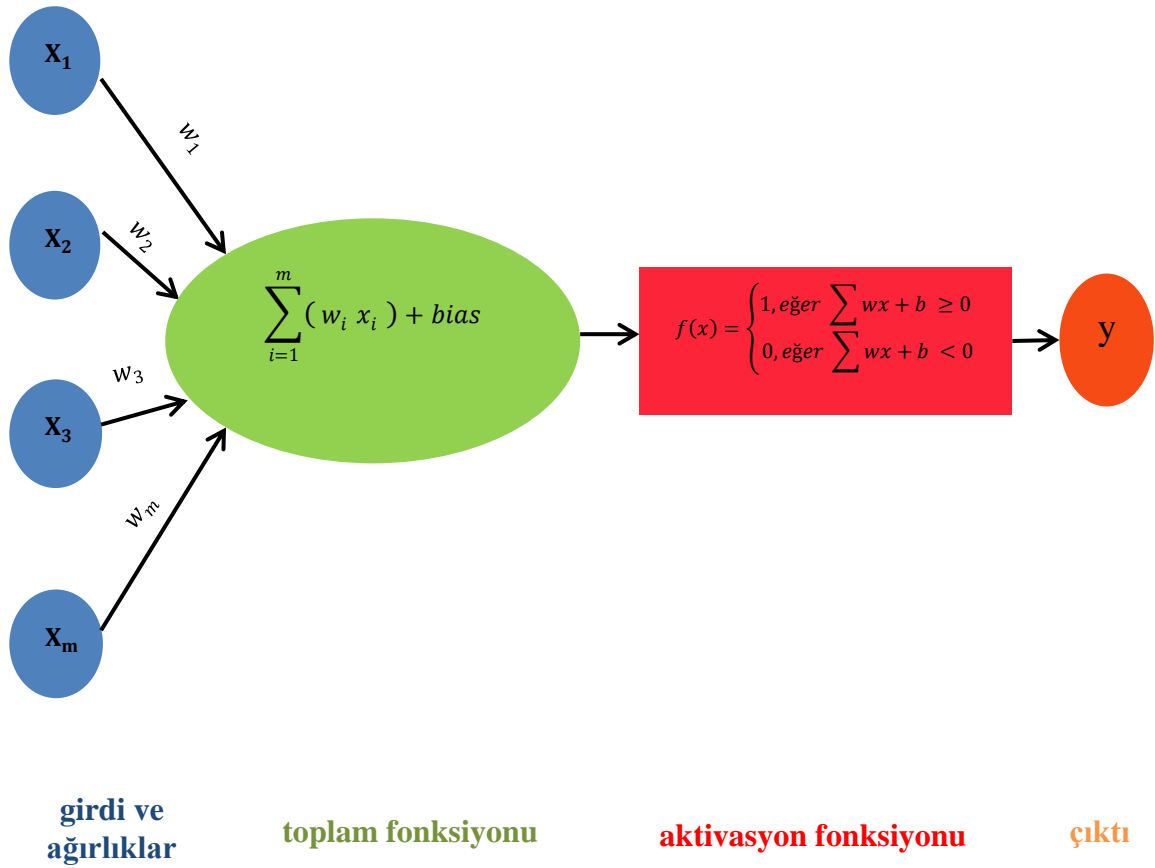
Şekil 2.6'da veri setindeki kelimelerin kaç tweet içerisinde kullanıldığını gösteren şekil yer almaktadır. Şekilde gösterilen veriler en sık kullanılan ilk 20 kelimeyi içermektedir.



Şekil 2.6 Kelimelerin kullanıldığı tweet sayısı

2.2 Yapay Sinir Ağları

Yapay sinir ağları (YSA), insan beyninin bilgi işleme tekniğinden esinlenerek geliştirilmiş bir bilgi işlem teknolojisidir. YSA ile basit biyolojik sinir sisteminin çalışma şekli taklit edilir. Taklit edilen sinir hücreleri nöronlar içerirler ve bu nöronlar çeşitli şekillerde birbirlerine bağlanarak ağı oluştururlar. Bu ağlar öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. Diğer bir ifadeyle, yapay sinir ağları, normalde bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerini gerektiren problemlere çözüm üretmektedir (Toprak, 2005). Şekil 2.7’ de temel yapay sinir ağı hücresi gösterilmektedir.



Şekil 2.7 Temel yapay sinir ağı hücresi

2.2.1 Öğrenme Yöntemleri

Yapay sinir ağları, öğrenme yöntemlerine göre birbirinden ayrılmaktadır. Bu bölümde en temel üç öğrenme yönteminden bahsedilmektedir.

2.2.1.1 Danışmanlı (Supervised) Öğrenme

Bu öğrenme yönteminde sinir ağına verilen girdilere karşılık gelen çıktı değerleri de ağ tarafından bilinmektedir. Sinir ağı, bu girdilere karşılık verilen çıktıları üretecek biçimde öğrenmektedir.

2.2.1.2 Danışmansız (Unsupervised) Öğrenme

Bu yöntemde girdilere karşılık gelen çıktı değerleri sisteme verilmemektedir. Sisteme sadece girdi değerleri gösterilmektedir. Örneklerdeki parametreler arasındaki ilişkileri sistemin kendi kendine öğrenmesi beklenmektedir. Sınıflandırma problemleri için kullanılan bir stratejidir.

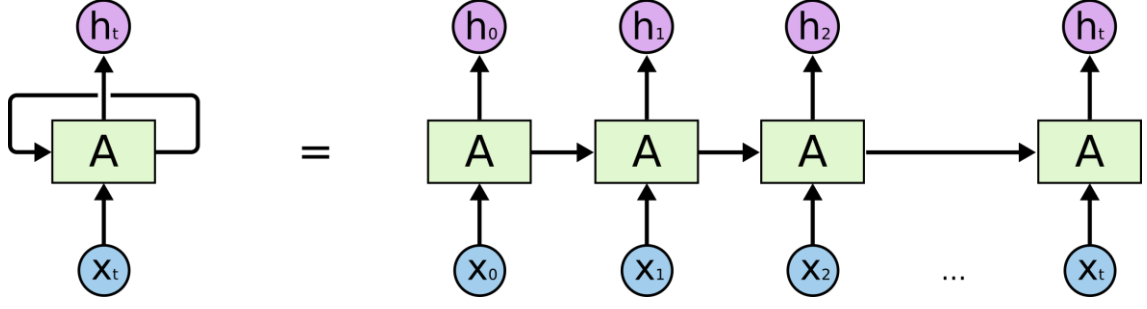
2.2.1.3 Pekiştirmeli (Reinforcement) Öğrenme

Bu öğrenme yönteminde sisteme bir öğretmen yardımcı olmaktadır. Yalnız her girdi setine karşılık gelen çıktı setini sisteme göstermek yerine sistemin verilen girdilere karşılık beklenen çıktıları üretmesini bekler. Sistemin ürettiği çıktı beklenen çıktıyla eşit olana kadar sistem öğrenmeye devam eder.

2.2.2 RNN Ağları

Öğrenme, her ne kadar sürekli gelişmeye dayalı bir süreç olsa da daha önce öğrendiklerimizin mantıksal kullanımı ile daha önce görülmemiş, yeni karşılaşılan bilgi de yorumlanarak anlaşılabilir. Burada karşılaşılan problem ise geçmişte öğrenilen bilginin hatırlanması ve eskinin devamı niteliğinde karşılaşılan bilginin buna göre yorumlanmasıdır. Diğer sinir ağlarının aksine Tekrarlayan Sinir Ağları (RNN) bu probleme bir çözüm getirmektedir.

RNN ağının genel mimarisi, birbirine sıkıca bağlı bir zincir gibi sıra gelen benzer hücrelerin birbirine bağlanması gibidir. Bilgi, her bir sinir hücresinden diğerine bir döngü vasıtası ile gidebilir. RNN'ler metin sınıflandırma, konuşma tanıma, dil modelleme, çeviri gibi problemler için kullanılabilir. Şekil 2.8'de RNN ağı mimarisi gösterilmektedir.



Şekil 2.8 RNN ağı mimarisi

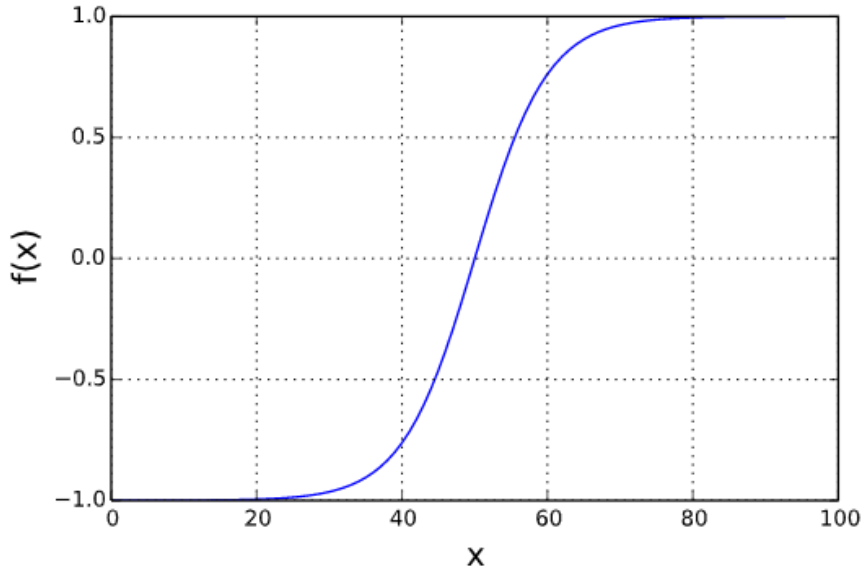
RNN ağları her ne kadar önceki öğrenilen bilginin mantıksal olarak yorumlanması ile karar verebiliyor olsa da yeni karşılaşılan bilgi ile daha önce öğrenilen bilgi arasındaki mantıksal bağlantı, hücreler arası ilerleme ve öğrenme ile doğru orantılı olarak gittikçe uzar ve kopar. Bu hafıza boşluğu problemine çözüm olarak Uzun Kısa Vadeli Hafıza Ağları geliştirilmiştir.

2.2.3 LSTM

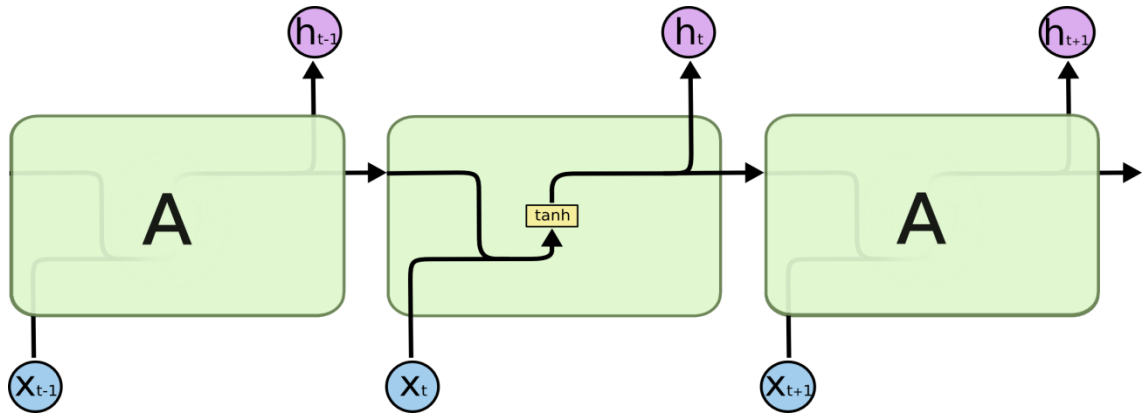
Uzun Kısa Vadeli Hafıza Ağları (LSTM), uzun vadeli hafıza ile öğrenebilen RNN'in uzun süreli hafıza problemine karşı tasarlanmış özel bir türüdür. Sepp Hochreiter ve Jürgen Schmidhuber tarafından 1997 yılında geliştirilmiştir ve günümüze kadar farklı birçok varyasyonu geliştirilmiştir. Geliştirilen tüm RNN ağları Şekil 2.10'da gösterildiği üzere yinelenen hücreler zinciri yapısındadır ve standart bir RNN ağındaki bir hücre, hiperbolik tanjant aktivasyonuna sahip tek bir kapıdan oluşur (Hochreiter & Schmidhuber, 1997). Aşağıda hiperbolik tanjant fonksiyonunun denklemleri ve Şekil 2.9'da grafiği gösterilmektedir.

$$a_j^i = f(x_j^i) = \tanh(x_j^i)$$

$$a_j^i = f(x_j^i) = \frac{2}{1 + \exp(-2x_j^i)} - 1$$



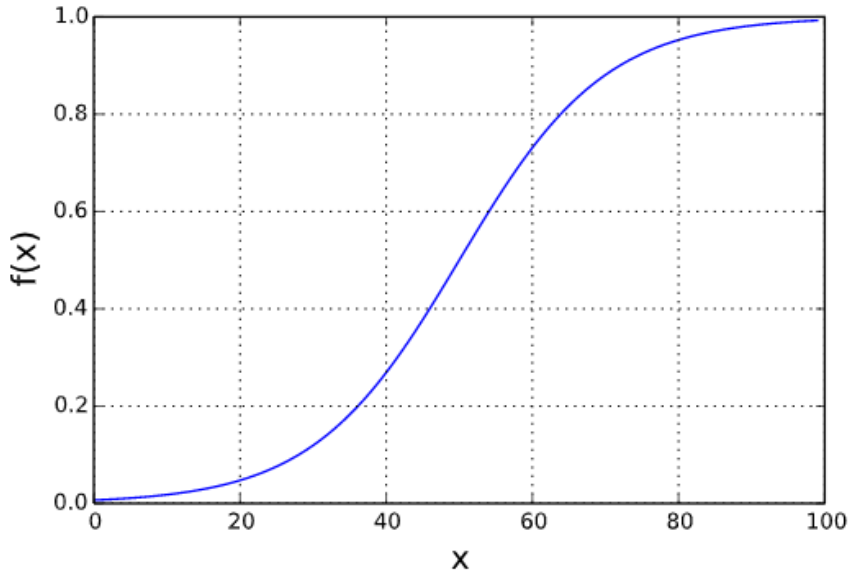
Şekil 2.9. Hiperbolik tanjant fonksiyonu grafiği



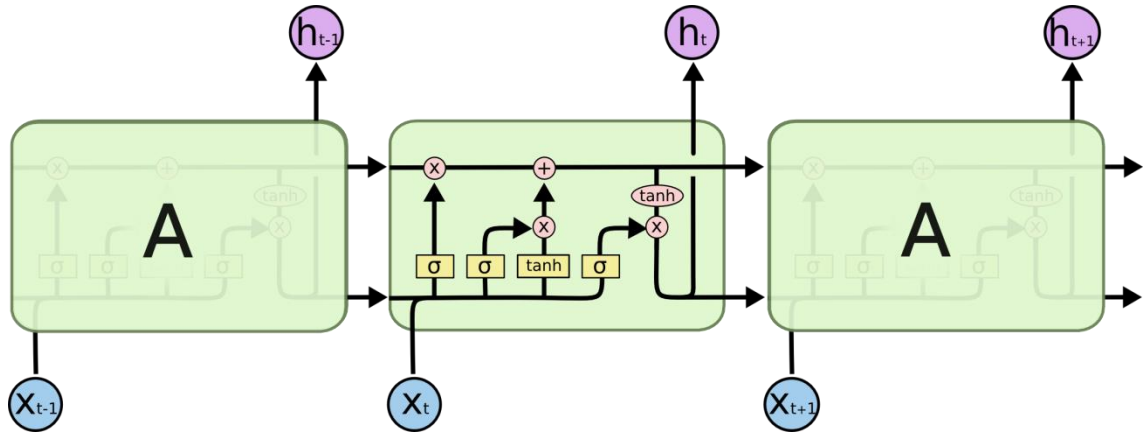
Şekil 2.10 RNN hücreler zinciri

LSTM ağırları ise standart RNN'den farklı olarak 4 kapıdan oluşan bir hücre yapısına sahiptir. Bunlar Giriş(Input), Çıkış(Output), Hücre(Cell) ve Unutma(Forget) kapılarıdır. Bu kapılar hücre durumuna bilgi ekleme ve çıkarma için kullanılır. Hücre yapısındaki bu kapılarda hiperbolik tanjant aktivasyonu ile beraber sigmoid aktivasyonu da kullanılır. Aşağıda sigmoid fonksiyonu denklemi ve Şekil 2.11'de grafiği gösterilmektedir. Şekil 2.12'de ise LSTM hücreler zinciri ve yapısı gösterilmektedir.

$$a_j^i = f(x_j^i) = \frac{1}{1 + \exp(-x_j^i)}$$

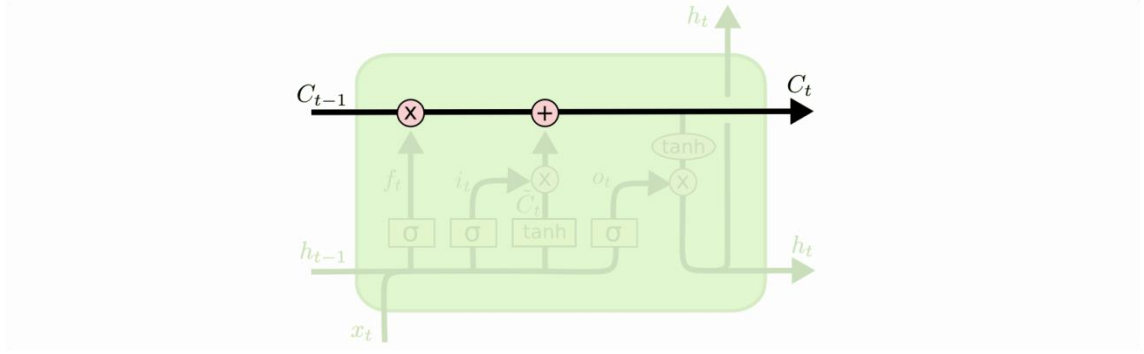


Şekil 2.11 Sigmoid fonksiyonu grafiği



Şekil 2.12 LSTM hücreler zinciri

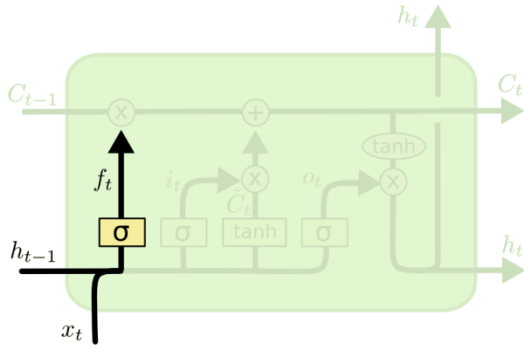
LSTM hücresinin üstündeki tekli ve sadece doğrusal değişimlere sahip olan yol hücre durumu olarak adlandırılır (Cell State). Şekil 2.13’de bu durum gösterilmektedir. Bu bilginin değişmeden taşındığı bir bağlantıdır. Cell State ile bilgi, birbirine bağlı hücreler zincirinde hiç değişmeden ilerleyebilir.



Şekil 2.13 Cell State Line

LSTM'deki ilk adımda hangi bilgilerin hücre durumundan çıkarılacağına karar vermektir. Buna sigmoid aktivasyonunu kullanan Unutma (Forget) Kapısı denilir. Sigmoid fonksiyonunun çıktı değerine göre bilgi tamamen tutulur veya unutulur. Aşağıda unutma kapısının kullandığı sigmoid aktivasyon fonksiyonunu ve Şekil 2.14'te LSTM hücresindeki yeri görülmektedir.

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

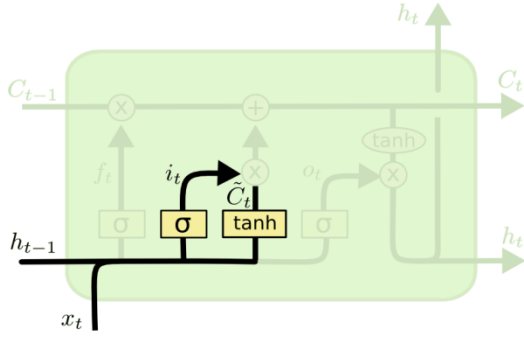


Şekil 2.14 LSTM Forget Gate

Bir sonraki adımda ise hücre durumuna hangi yeni bilgilerin eklenmesi gerektiğine karar verilir. Buradaki sigmoid aktivasyonu Giriş (Input) Kapısı olarak adlandırılır. Bu kapıda hangi bilgilerin güncellenmesi gerektiğine karar verilir. Yeni aday bilgilerin vektörü ise hiperbolik tanjant aktivasyonu ile oluşturulur. Aşağıda burada kullanılan sigmoid aktivasyonu ve hiperbolik tanjant aktivasyon formüllerini görmekteyiz. Şekil 2.15'te giriş kapısının LSTM hücresindeki yeri görülmektedir.

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

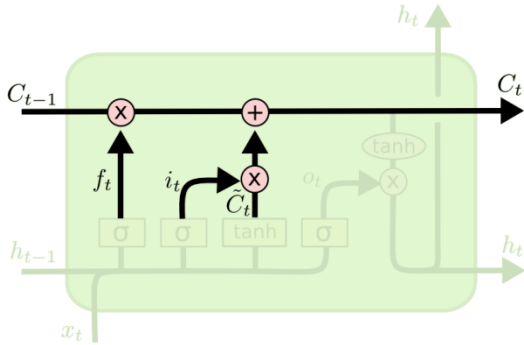
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$



Şekil 2.15 LSTM Input Gate

Önceki adımda güncellenmesine (eklenmesine veya unutulmasına) karar verilen bilgilerin her bir aday durum değeri için ne kadar güncellemek gerektiğine bu adımda karar verilir ve bilgi buna göre işlenir. Bu aşamaya Hücre (Cell) kapısı adı verilir. Şekil 2.16’da hücre kapısının LSTM hücresindeki yeri görülmektedir.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

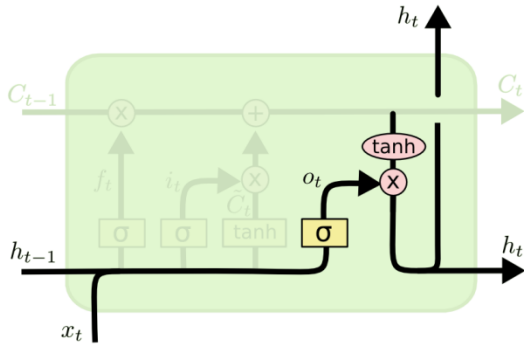


Şekil 2.16 LSTM Cell Gate

Son aşamada hücre çıktısı oluşturulur. Bu aşama Çıkış (Output) kapısı olarak adlandırılır. Bu çıktı hücre durumunun filtrelenmiş halidir. Hücre durumundan hangi bilgilerin çıkarılacağına, hücre durumunun hiperbolik tanjant aktivasyonu ile vektörel olarak ifade edilmesi ve vektörün sigmoid aktivasyonu ile çarpılmasıyla karar verilir. Aşağıda hiperbolik tanjant aktivasyon formülü ve Şekil 2.17’de çıkış kapısının LSTM hücresindeki yeri görülmektedir (Olah, 2015).

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

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



Şekil 2.17 LSTM Output Gate

2.3 NLTK

NLTK, 2001 yılında Pennsylvania Üniversitesi'nde hesaplamalı bir ilbilim dersi ile birlikte geliştirilmiştir. Python dili ile yazılmış olan bu kütüphane, GPL ile açık kaynak olarak lisanslanmış ve kullanıma sunulmuştur. Projemizde NLTK içerisinde yer alan “WordNetLemmatizer”, “PorterStemmer” metodlarından faydalanılmıştır. “WordNetLemmatizer”, kelimeleri köklerine ayırma işlemi için, “PorterStemmer”, İngilizce’de en yaygın olarak kullanılandır. İkisinin de kullanmasının sebebi ise, kök bulma (stemming) işleminin bazı dezavantajlar vardır. Bu dezavantajları lemmatizasyon aşaması ile giderilmesi ve gürültünün azaltılması hedeflenmiştir.

2.4 Kök Bulma (Stemming)

Birçok durumda kelimelerin yapısal olarak çeşitleri, benzer anlamlara sahiptir. Bu sebeple bir takım kök bulma algoritmaları geliştirilmiştir (Porter, 1980; Lovins, 1968; Paice, 1990). Bu algoritmalar sonucunda kelimeler köklerine ayrılmaktadırlar. Proje kapsamında PorterStemmer algoritması kullanılmıştır. Kök bulma işlemi sırasında kelimelerin ön ve son ekleri atılmıştır. Bu aşamadaki sorunların temelinde, kelimelerin dilbilimsel olarak analiz ve anlamlarına bakılmamasıdır. Bundan dolayı anlamsal veya kavramsal olarak aynı gruba ait olan kelimeler farklı köklere indirgenebilmektedir. Bu durumun önüne geçmek amacıyla kök bulma işleminden sonra kontrol amaçlı ve gürültülü veri oluşumunu önlemek amacıyla lemmatizasyon aşaması eklenmiştir (Özbek, 2019).

Çizelge 2.2 Stemming’de Karşılaşılan Hatalara Örnek

Biçim	Ek	Stem
studies	-es	Studi
Studying	-ing	study

Çizelge 2.2’deki örnekte görüldüğü üzere kök bulma işlemi bazı kelimeler için stabil bir ayrıştırma gerçekleştirememektedir. Bu gibi durumları önlemek amacıyla bu aşamaya ek olarak lemmatizasyon aşaması da gerçekleştirilmektedir.

2.5 Lemmatizasyon

Dilbiliminde, anlamsal olarak köke inerek bir kelimenin farklı ekler almış hallerinin tek bir kelime olarak analiz edilmesidir. Bir kelimenin en yalın halinin belirlenmesidir. Örnek olarak “yaz” kelimesi; “yazmak”, ”yazıyor”, ”yazacak”, gibi birden fazla şekilde kullanılabilir. Lemmatizasyon aşamasında bu kelimeler tek bir kelime olacak şekilde köklerine indirgenmiştir. Projede “WordNet” veritabanında yer alan kelimelerden faydalanılmıştır. Kök bulma algoritmalarından farklı olarak kelimenin anlamına bakılmaktadır. Örnek olarak İngilizce’de geçen “worst” kelimesi ve “bad” kelimesi anlamsal olarak aynı grupta yer alsalar da kök bulma aşamasında bunlar farklı kelimeler olarak gruplanır. Bu aşamada bunun önüne geçilerek tek bir gruba indirgenmesi hedeflenmiştir (Press, 2008).

Çizelge 2.3 Lemmatizasyon Örneği

Biçim	Morfolojik Bilgi	Lemma
Studies	Fiilin şimdiki zaman hali	study
studying	İsim-fiil	study

Kök bulmadan aşamasından farklı olarak lemmatizasyon aşamasında kelimenin morfolojik bilgisine bakılır ve kelimenin köküne ulaşılmaya çalışır. “Studies” kelimesi kök bulma aşamasından sonra “studi” olarak güncellerinken lemmatizasyon aşamasında

Çizelge 2.3'te görüldüğü üzere “study” olarak güncellenmiştir.

2.6 SpellChecker

SpellChecker, python dili ile kodlanmış bir kütüphanedir. Yanlış yazılmış kelimeleri kullanım sıklıklarına göre tahmin ederek en yüksek sıklığa sahip olan kelimeyi tahmin etmeye çalışır. Proje kapsamında Twitter kullancılarının tweetlerinden duygu analizi yapılmıştır. Gündelik hayatta birçok kelime yazılırken harfler eksik veya fazla yazılabilmektedir. İnsanlar, bu kelimeleri tahmine dayalı olarak anlayabilir fakat makine dili ortamında bu kelimelerin hepsi anlamsız olarak nitelendirilmektedir ve bu durum gürültü oluşumuna sebep olmaktadır. Bu gürültülü veri oluşumunun önüne geçebilmek amacıyla her kelime analiz edilmeden önce “SpellChecker” aşamasından geçirilir ve anlamlı birer kelimeye evrilmesi beklenir (Norvig, 2016).

Çizelge 2.4 SpellChecker Örneği

Kelimenin Yanlış Hali	Kelimenin Doğru Hali
Goingi	Go
Helpa	Help
Yestrday	Yesterday

Çizelge 2.4'de spellchecker'ın çalışma mantığı gösterilmektedir. Yazım hatası olan kelime yerine o kelimeye en yakın kelime ile değiştirilmiştir.

2.7 Veri Ön İşleme

Veri ön işleme aşaması, ham verileri temizleyip anlaşılır bir şekile dönüştürmeyi amaçlayan veri madenciliği tekniğidir. Gündelik hayatta kullanılan kelimeler, cümleler, noktalama işaretleri gibi veriler çoğunlukla eksik, hatalı veya belirsizdir. Veri ön işleme aşamasında bu tür problemlerin önüne geçmek hedeflenmektedir. Proje kapsamında Twitterdan çekilen tweetler üzerinde duygu analizi yapılmaktadır. Bu tweetler yapay sinir ağına verilmeden önce veri ön işleme aşamasından geçirilerek temizlenir. Çizelge 2.5'te örnek bir tweet'in veri ön işleme aşamasından nasıl geçirildiği görülmektedir.

Çizelge 2.5 Örnek Veri Ön İşleme

Tweet'in ilk hali	I am goingi afk from https://www.twitter.com #HashTagControl #erdemsahin #morehashtags :D to 47, :?! @tunahan sorry my love @umitsalman <3 at least Friday?
Link Temizleme	I am goingi afk from #HashTagControl #erdemsahin #morehashtags :D to 47,:?! @tunahan sorry my love @umitsalman <3 at least Friday?
Mention Temizleme	I am goingi afk from #HashTagControl #erdemsahin #morehashtags :D to 47,:?! _mention_ sorry my love _mention_ <3 at least Friday?
Hashtag Temizleme	I am goingi afk from Hash Tag Control :D to 47,:?! _mention_ sorry my love _mention_ <3 at least Friday?
Emoji Temizleme	I am goingi afk from Hash Tag Control laugh to 47,:?! _mention_ sorry my love _mention_ love at least Friday?
Kısaltma Temizleme	I am goingi Away From Keyboard from Hash Tag Control laugh to 47,:?! _mention_ sorry my love _mention_ love at least Friday?
Noktalama Temizleme	i am goingi away from keyboard from hash tag control laugh to mention sorry my love mention love at least friday
Kelime Düzeltme (Son)	go away keyboard hash tag control laugh mention sorry love mention love least friday

2.8 Python Programlama Dili



Şekil 2.18 Python logosu

Python, nesne yönelimli, yorumlamalı, modüler ve etkileşimli yüksek seviyeli bir programlama dilidir. Girintilere dayalı basit sözdizimi, dilin öğrenilmesini ve akılda kalmasını kolaylaştırır. Bu da ona söz diziminin ayrıntıları ile vakit yitirmeden programlama yapılmaya başlanabilen bir dil olma özelliği kazandırır. Hemen hemen her türlü platformda (Unix, Linux, Mac,

Windows, Amiga, Symbian) çalışabilir. Python ile sistem programlama, kullanıcı arabirimi programlama, ağ programlama, uygulama ve veritabanı yazılımı programlama gibi birçok alanda yazılım geliştirilebilir.

2.9 Keras



Şekil 2.19 Keras logosu

Keras, Python dili ile yazılmıştır ve TensorFlow, CNTK veya Theano'nun üstünde çalışabilen üst düzey sinir ağları API'sidir. CPU ve GPU üzerinde sorunsuz çalışır. RNN ve CNN sinir ağlarını oluşturmak için kullanılabilir.

2.10 TensorFlow



Şekil 2.20 Tensorflow logosu

TensorFlow, yüksek performans gerektiren sayısal hesaplamalar için oluşturulmuş açık kaynaklı bir yazılım kütüphanesidir. Esnek mimarisi sayesinde kullanımı çeşitli platformlara yayılmıştır. Aslen Google'ın AI organizasyonunda yer alan Google beyin takımındaki araştırmacılar ve mühendisler tarafından geliştirilen TensorFlow, makine öğrenimi ve derin öğrenme alanlarında sıklıkla kullanılmaktadır. Ayrıca esnek sayısal hesaplama çekirdeği sayesinde diğer bilimsel alanlarda da kullanılmaktadır.

2.11 Docker



Şekil 2.21 Docker logosu

Docker, en çok kullanılan yazılım konteynerleştirme platformlarından bir tanesidir. Uygulama geliştirmeyi ve çalıştırmayı kolaylaştırmaktadır. Sanal makinalara kıyasla daha esnek bir yapıya sahiptir.

2.12 IBM POWER9



Nvlink mimarisi ile bütünleştirilmiş 128 Core Power9 CPU , 512GB RAM ve 4 adet Nvidia Tesla V100 GPU'lu hesaplama sunucusudur.

Şekil 2.22 IBM Power9 logosu

2.13 Plotly



Plotly, veri grafiği oluşturmada ve analiz etmede kolaylık sağlayan bir araçtır. Projenin web arayüzünü oluşturmak amacıyla kullanılmaktadır.

Şekil 2.23 Plotly logosu

2.14 Jupyter



Jupyter, çeşitli programlama dilleri ile kullanılabilen, etkileşimli ortam sağlayan bir araçtır. Tarayıcı tabanlı olarak da çalışabilmektedir.

Şekil 2.24 Jupyter logosu

2.15 Pandas



Python programlama dili için yüksek performanslı, kullanımı kolay veri yapıları ve veri analizi araçları sunan bir veri analizi kütüphanesidir.

Şekil 2.25 Pandas logosu

2.16 Numpy



Şekil 2.26 NumPy logosu

Python ile bilimsel hesaplamalar yapılabilmesini sağlayan bir kütüphanedir. Çok boyutlu diziler, çeşitli türetilmiş nesneler ve diğer birçok veri yapıları üzerinde hızlı ve kolay bir şekilde işlem yapmayı sağlamaktadır.

2.17 Matplotlib



Şekil 2.27 Matplotlib logosu

Matplotlib, Python programlama dili için hazırlanmış, hem iki boyutlu hem de üç boyutlu grafikler oluşturmak için kullanılan bir kütüphanedir. Verilerin etkileşimli bir biçimde ve yüksek kalitede sunulabilmesini sağlamaktadır.

2.18 Twitter API



Şekil 2.28 Twitter API logosu

Analiz edilmesi için ihtiyaç duyulan tweet'leri elde etmek amacıyla kullanılan API servsidir. Twitter API'sinin kullanımı kolaylaştıran "python-twitter" kütüphanesi ile birlikte projede kullanılmaktadır.

BÖLÜM ÜÇ - PROJENİN GERÇEKLENMESİ

Bu bölümde, ikinci bölümde açıklanan araç ve yöntemlerin uygulamada nasıl kullanıldığı anlatılmaktadır. Kaynak koddan bazı önemli kesitler ve ekran görüntüleri de paylaşılmaktadır.

3.1 Twitter API Kullanımı için Gereksinimler

Twitter API servisinin kullanılabilmesi için öncelikle onaylanmış bir geliştirici hesabına ihtiyaç duyulmaktadır. Hesap elde edildikten sonra API özelliklerini kullanabilmek için öncelikle bir uygulama oluşturmak gerekmektedir. Uygulama oluşturmak için Twitter tarafından istenen bilgiler Şekil 3.1’de gösterilmektedir.

App details

The following app details will be visible to app users and are required to generate the API keys needed to authenticate Twitter developer products.

App name (required) ?

Maximum characters: **32**

Application description (required)

Share a description of your app. This description will be visible to users so this is a good place to tell them what your app does.

Please be detailed.

Between 10 and 200 characters

Şekil 3.1 Twitter API Uygulama Oluşturma Ekranı

Şekil 3.1’de istenilen bilgiler girildikten sonra bu uygulama için “access token” ve “token secret” anahtar bilgileri elde edilmelidir.

Keys and tokens

Keys, secret keys and access tokens management.

Consumer API keys

0jw56xW

Bg4Dh (API key)

JJmNB4FAfVRvuP144tk

EKiX7oXgan2Vjh9d (API secret key)

Regenerate

Access token & access token secret

809713505721024!

IZm5ii3ZCG2V9ioIAkYcqN (Access token)

aJqEGFni79kE2iCadma

IJR8iv8AaH4Q (Access token secret)

Read and write (Access level)

Revoke

Regenerate

Şekil 3.2 Key ve Token Değerlerinin Elde Edilmesi

Elde edilen anahtar bilgileri ve Şekil 3.2’de görülen “consumer API keys” anahtar bilgileriyle birlikte toplam dört adet anahtar elde edilir. Bu anahtarlar python-twitter kütüphanesi aracılığıyla API hizmeti alınabilmesi için gerekmektedir.

3.2 python-twitter Kütüphanesi ile Kullanıcı Tweet’lerinin Elde Edilmesi

Anahtar bilgileri elde edildikten sonra API vasıtasıyla tweetleri elde edebilmek için Python programlama dili için oluşturulmuş “python-twitter” kütüphanesi kullanılmaktadır. Kütüphanenin projeye dahil edilmesi, anahtar bilgilerinin girilmesi ve örnek bir kullanıcının tweet’lerinin çekilmesi için aşağıdaki kod parçasığı kullanılmaktadır:


```

tweets = []
CONSUMER_KEY = 'VqZGnJ4n4TQVdQqCUPLnZU6cE'
CONSUMER_SECRET = 'koOA2ICE3KW3xvJN9XFSI15N4kmBehIyMX3WTJBTwptF95xtlIF'
ACCESS_TOKEN = '809713505721024512-GNM9zHuSdvOe8gNY4JuPHT0Jos9TTjH'
ACCESS_TOKEN_SECRET = 'USG95IQStCAsPWFnlq7RapRKu2rKK0r8tLjNufYhbApvJ'

api = twitter.Api(consumer_key=CONSUMER_KEY,
                  consumer_secret=CONSUMER_SECRET,
                  access_token_key=ACCESS_TOKEN,
                  access_token_secret=ACCESS_TOKEN_SECRET)

statuses = api.GetUserTimeline(screen_name=value, count=6, exclude_replies=False)

for s in statuses:
    tweets.append(s.text)

```

Yukarıdaki kod bloğunda görüldüğü üzere, “screen_name” parametresine “realDonaldTrump” ve “count” parametresine “6” değeri verilmiştir. Dolayısıyla bu kullanıcı adına sahip kullanıcının son 6 tweet’i elde edilmektedir. Kodun çıktısı ise aşağıdaki gibidir:

Tweet: The Dems are getting NOTHING done in Congress! They only want a Do-Over on Mueller!

Tweet: Impeach for what, having created perhaps the greatest Economy in our Country’s history, rebuilding our Military, ta... <https://t.co/IUJWC45OeY>

Tweet: @ianbremmer now admits that he MADE UP “a completely ludicrous quote,” attributing it to me. This is what’s going... <https://t.co/EOx6VPgKia>

Tweet: Great to get out there and take a few cuts at the plate yesterday—I had a blast with all these extraordinary young... <https://t.co/TQ7O6BVMSJ>

Tweet: RT @ObamaFoundation: Congratulations on the inauguration of @ChicagosMayor Lori Lightfoot, Treasurer Melissa Conyears-Ervin, @ChiCityClerk...

Tweet: Here’s a great story: While we’ve still got a lot of work to do to make college affordable for everyone, that didn’t... <https://t.co/eW9usqYzu1>

3.3 Elde Edilen tweet'lerin Doğal Dil İşleme Adımlarından Geçirilmesi

Elde edilen bu tweetlerin eğitilmiş yapay sinir ağı modeline verilmeden önce bir takım veri ön işleme aşamalarından geçmesi gerekmektedir. Veri ön işleme aşamasındaki adımlar sırasıyla açıklanmaktadır:

1. Link temizleme aşamasında tweet'te bulunan link'ler temizlenir.
2. Alınana tweet başka bir kullanıcıya atılan bir tweet ise “@mention” ifadesi temizlenir.
3. Tweet içerisinde “#hashtag” varsa, hashtag içeriğine bakılır ve anlamsız harfler veya kelimelerden oluşuyorsa ise temizlenir. Anlamlı kelimeler varsa “#” karakteri temizlenir ve kelime elde edilir.
4. Çizelge 3.1'deki regex ifadelerinden yararlanılarak tweet'ler içerisinde kullanılan emojiler yerine o emojileri ifade eden uygun kelimeler yerleştirilir.

Çizelge 3.1 Emoji-Regex Dönüşüm tablosu

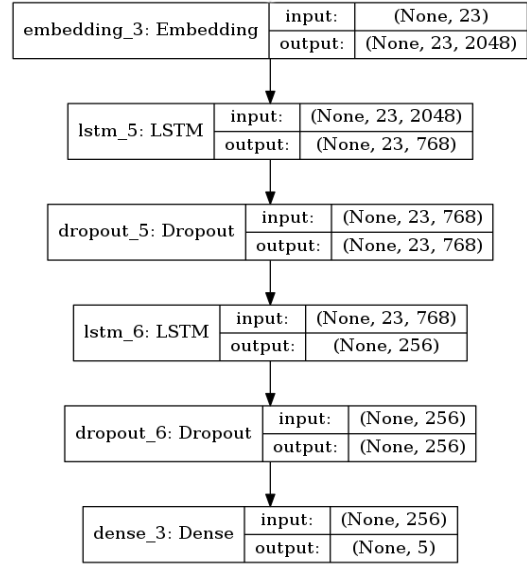
Emoticon	Type	Regex
, :) , (: (: , (-: , :') , :D , : D , :-D , xD , XD , X -D	Laugh	<code>r'(:\s?D :-D x-?D X-?D)', 'laugh'</code>
<3 , :*	Love	<code>r'(<3 : *)', 'love'</code>
:- (: (,) : ,) -:	Sad	<code>r'(:\s?(\(:-\(\) \s?:)\-:)', 'sad'</code>
: , (, : ' (, : ' (Cry	<code>r'(: , \(: :\'((: \' O)', 'cry'</code>

5. Gündelik hayatta sıkça kullanılan kısaltmalar tespit edilir ve yerine bu kısaltmaların açılımları yerleştirilir.
6. Kelime yazım hataları “spellchecker” yardımıyla tespit edilir ve tahmin yolu ile hatalı kelime yerine en uygun kelime yazılır.

Bu aşamalardan sonra yapay sinir ağı modeline verilmeden önce vektör formatına çevirilmeye hazır temiz veriler elde edilmektedir.

3.4. Kullanılan Yapay Sinir Ağının Yapısı

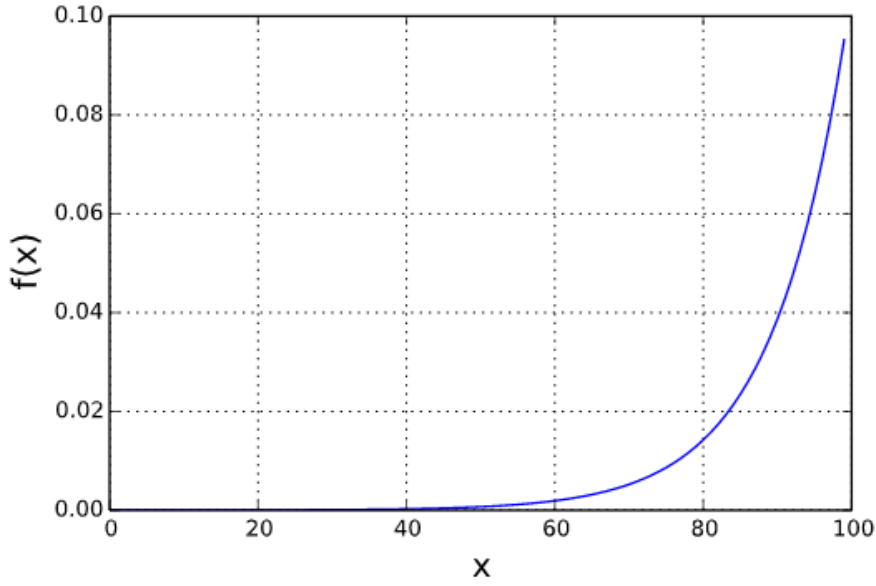
Yapay sinir ağının katmanlı yapısı, Keras kütüphanesinin Sequential model sınıfı kullanılarak oluşturulmuştur. Sinir ağımız giriş verisi olarak doğal dil işleme aşamaları ile temizlenmiş metinde kullanılan kelimeleri temsil eden, eğitim veri setindeki yaklaşık 12.061 farklı kelimenin indeks numarası ile numaralandırılmış olan ve uzunluğu sabit olarak 23 (metindeki maksimum kelime sayısı) belirlenmiş bir dizidir. Çıktı ise one hot encoding ile kodlanmış ve her bir indekzin bir duygu sınıfını temsil ettiği uzunluğu 5 olan bir dizidir. Şekil 3.3'de YSA



Şekil 3.3 Sinir Ağı Modeli

modelimiz görülmektedir. İlk katmanımız yukarıdaki diziyi giriş olarak alacak Embedding katmanıdır. Giriş boyutu 11.000 kelime olarak belirlenmiştir. Giriş dizisi uzunluğu 23 dür. Sonraki iki katman ise sırasıyla 768 ve 256 hücreli RNN tabanlı LSTM katmanlarıdır. Aktivasyon fonksiyonu olarak Keras varsayılanı olan hiperbolik tanjant kullanılmıştır. LSTM katmanları sonrası ağın ezberlemesini önlemek amacı ile birer Dropout katmanı eklenmiştir ve drop out katsayısı 0.2'dir. Tekrarlama adımındaki drop out katsayısı ise 0.3 olarak belirlenmiştir. İkincil kullanılan LSTM katmanının ilk LSTM katmanındaki gizli çıktıları kullanması amacıyla birincil LSTM katmanında geri dönüş dizileri (return_sequences) aktifleştirilmiştir. Çıktı katmanı aktivasyon fonksiyonu olarak softmax fonksiyonunu kullanan ve çıktı boyutsallığı 5 (duygu sayısı) olan bir Dense katmanıdır. Optimizasyon olarak Adam optimizasyon algoritması kullanılmıştır. Adam algoritmasının β_1 ve β_2 katsayı değerleri sırasıyla 0.9 ve 0.999'dur. Öğrenme katsayısı 0.001 olarak düzenlenmiştir. Ağ eğitimindeki tur sayısı (epoch) 4'tür ve sinir ağına bir anda verilecek veri sayısı 64'tür.

$$a^i_j = f(x^i_j) = \frac{\exp(z^i_j)}{\sum_k \exp(z^i_k)}$$



Şekil 3.4 Softmax fonksiyonu grafiği

3.4.1 Adam (Adaptive Moment Estimation) Optimizasyon Algoritması

Derin öğrenme modelleri için optimizasyon algoritmasının seçimi, istenilen sonuçlara ne kadar hızlı ulaşılabilceği konusunda önemli etkiye sahiptir. Modelimize en uygun optimizasyon algoritmasının kullanılması, geliştiricilere bazen günler kazandırabilmektedir.

Adam algoritması, eğitim aşamasında modelin ağırlıklarını güncellemek amacıyla, klasik stokastik gradyan iniş (stochastic gradient descent) prosedürünün yerine kullanılabilen bir optimizasyon algoritmasıdır. Adam algoritmasının bazı önemli faydaları, ilk sunulduğu zamanlarda aşağıdaki gibi belirtilmiştir:

- Uygulanması basit
- Hesaplama açısından verimli
- Düşük hafıza gereksinimi
- Büyük boyutta veri veya parametre içeren problemler için uygun
- Gürültülü veya seyrek gradyan problemleri için uygun
- Hiper parametreler çok az ayar gerektirir

3.4.1.1 Adam Algoritmasının Çalışması

Adam algoritması, klasik stokastik gradyan iniş yönteminden biraz farklıdır. Stokastik gradyan iniş yönteminde her ağırlık güncellemesi için tek bir öğrenme oranı (learning rate) kullanılmaktadır ve eğitim sırasında öğrenme oranı değişmemektedir. Ağırlık her ağırlığı için bir öğrenme oranı sağlanır ve ayrı ayrı öğrenme çıktısı olarak uyarlanır. Adam algoritması, stokastik gradyan iniş yönteminin iki faydalı özelliğinin birleşimi olarak tanımlanmaktadır:

- **Adaptive Gradient Algorithm (AdaGrad)**, her parametre için bir öğrenme oranı sağlar. Bu öğrenme oranı seyrek gradyan problemlerinde performansı artışı sağlar (doğal dil, bilgisayarlı görme problemleri vb.).
- **Root Mean Square Propagation (RMSProp)**, her parametre için bir öğrenme oranı sağlar. Bu öğrenme oranı ağırlık için gradyanların en son büyüklüklerinin ortalamasına göre uyarlanmaktadır.

Adam algoritması, AdaGrad ve RMSProp yaklaşımlarının ikisinin de faydalarından yararlanır. Parametre öğrenim oranlarını RMSProp'taki gibi ilk anın ortalamasını temel alarak uyarlamak yerine, Adam algoritması, gradyanların ikinci anının da ortalamasından yararlanır. Aşağıdaki formül adam algoritması adımlarıdır.

foreach (w^j):

$$v_t = \beta_1 * v_{t-1} - (1 - \beta_1) * g_t$$

$$s_t = \beta_2 * s_{t-1} - (1 - \beta_2) * g_t^2$$

$$\Delta\omega_t = -\eta \frac{v_t}{\sqrt{s_t + \epsilon}} * g_t$$

$$\omega_{t+1} = \omega_t + \Delta\omega_t$$

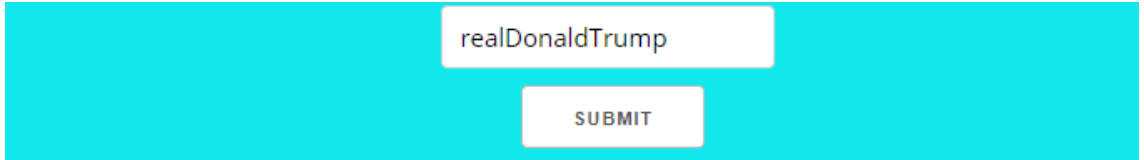
Adam algoritması konfigürasyon parametreleri;

- η , öğrenme oranı veya adım sayısı olarak da belirtilmektedir. Ağırlıkların güncellenme oranını belirtmektedir (örneğin 0,001). Yüksek değerler (örneğin 0,3), başlangıçtaki öğrenme işlemlerinin daha hızlı olmasına sebep olur. Düşük değerler (örneğin 1,0E-5) ise öğrenmeyi yavaşlatır.

- β_1 , ilk an tahminleri için üstel bozulma oranını belirtmektedir (örneğin 0,9).
- β_2 , ikinci an tahminleri için üstel bozulma oranını belirtmektedir (örneğin 0,999). Doğal Dil İşleme veya Bilgisayarlı Görme çalışmalarında bu değer 1,0 değerine yakın bir değer olmalıdır.
- ϵ , uygulamada sıfıra bölme hatasını engellemek amacıyla kullanılan çok küçük bir sayı (örneğin 10E-8).

3.5. Uygulama Arayüzü

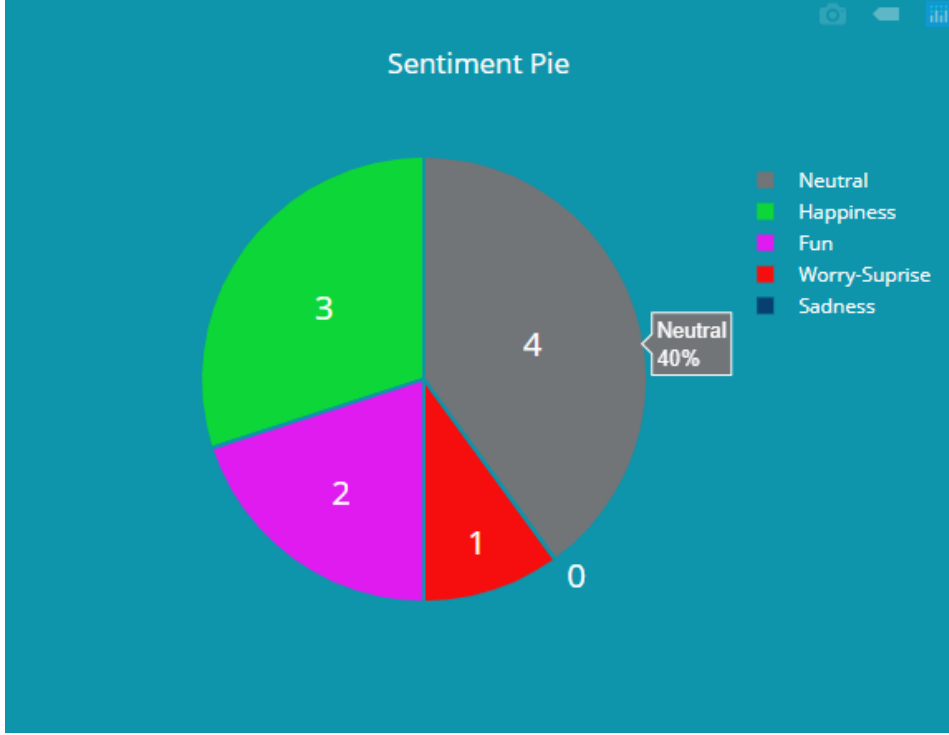
Uygulamamız web tabanlıdır. İlk olarak duygu analizi yapılmak istenen Twitter kullanıcısının kullanıcı adı bilgisinin girilmesini isteyen bir form görüntülenmektedir.



The image shows a web form with a light blue background. There is a white text input field containing the text 'realDonaldTrump'. Below the input field is a white button with the text 'SUBMIT' in blue capital letters.

Şekil 3.5 Arayüz Tasarımı -1

Kullanıcı adı bilgisi girildikten sonra, kullanıcının tweet'lerinin Twitter API vasıtasıyla elde edilmesi, doğal dil işleme adımlarından geçirilmesi, eğitilmiş ağ modeline gönderilerek analiz sonuçlarının elde edilmesi adımları gerçekleştirilir. Bu analiz sonuçları da Şekil 3.4'teki gibi sunulmaktadır.



Şekil 3.6 Arayüz Tasarımı -2

Şekil 3.4'te görülmekte olan, analiz sonucu olarak sunulan pasta grafiğinde, her bir duygu sınıfı için dilimler oluşturulmaktadır. Dilimlerin üzerindeki sayılar o duygu sınıfına ait kaç adet tweet atıldığını belirtmektedir. Analiz son 10 tweet üzerinde gerçekleştirilmektedir. Dolayısıyla tüm dilimlerin üzerindeki sayıların toplamı 10'a eşittir. Ayrıca her duygu sınıfı için ayrı renklendirme yapılmıştır ve bu renklerin hangi duygu sınıfını temsil ettiği sağ tarafta belirtilmiştir.

BÖLÜM DÖRT - DEĞERLENDİRME ve SONUÇ

Bu bölümde projede kullanılan yapay sinir ağı modelinin başarımlar oranları paylaşılmaktadır. K-Fold yöntemi kullanılarak veri seti parçalara bölünmüş olup bu parçaların test sonuçları belirli k değerleri için grafiksel olarak paylaşılmaktadır. Ayrıca sinir ağına başarımlar oranını arttırmak için neler yapılabileceği konusunda değerlendirmeler paylaşılmaktadır.

4.1 Yapay Sinir Ağı İyileştirmeleri

Yapay sinir ağımda ilk yapılabilecek iyileştirme, veri setindeki gürültü oranının azaltılması olacaktır. Burada gürültü ile kastedilen veri setindeki bazı metinlerin yanlış etiketlenmesi, metinlerin anlamsız olması veya birden çok ve zıt duyguyu olasılıksal olarak eşit veya yakın olarak barındırmasıdır. Veri seti düzenlemelerinden sonraki iyileştirme aşaması ise sinir ağına girdi olarak verilecek metinlerdeki kelimelerin, doğru doğal dil aşamalarından geçtiğini doğrulamak ve kontroller sonrasında doğal dil aşamalarında gerekli düzenlemelerin yapılmasıdır. Veri setinin ve girdilerin doğruluğu sağlandıktan sonraki son aşama ise farklı sinir ağlarının başarımlar olarak karşılaştırılması, kullanılan sinir ağı modelinde yapısal (hücre sayısı ve katman düzenlemeleri) değişiklikler yapılarak başarımlar oranlarının karşılaştırılması, sinir ağı katsayılarının düzenlenmesi ve farklı epoch/batch size parametreleri ile ağı eğitilmesidir.

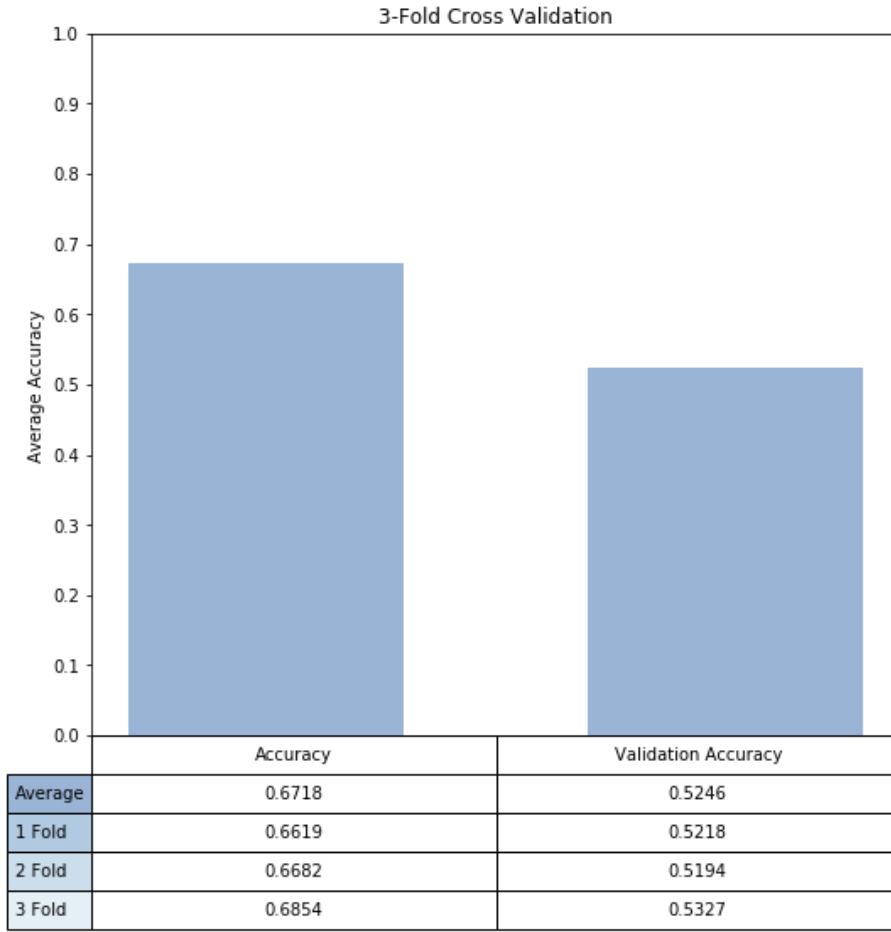
4.2 Sonuç

Bu çalışmada metin sınıflandırma yöntemleriyle Twitter'dan elde edilen veriler üzerinde duygu sınıflandırması yapılmıştır. Veri setinin eğitim ve test aşamasında k-fold yönteminden faydalanılmıştır. Bu yöntemde K tam sayısına göre veri seti parçalara bölünerek test edilmiştir. Şekil 4.1'de k tam sayısı 5 seçildiğinde veri setinin test ve eğitim aşaması sırasındaki parçalanışı görülmektedir.

1. adım	TEST	TRAIN	TRAIN	TRAIN	TRAIN
2. adım	TRAIN	TEST		TRAIN	TRAIN
3. adım	TRAIN	TRAIN	TEST	TRAIN	TRAIN
4. adım	TRAIN	TRAIN	TRAIN	TEST	TRAIN
5. adım	TRAIN	TRAIN	TRAIN	TRAIN	TEST

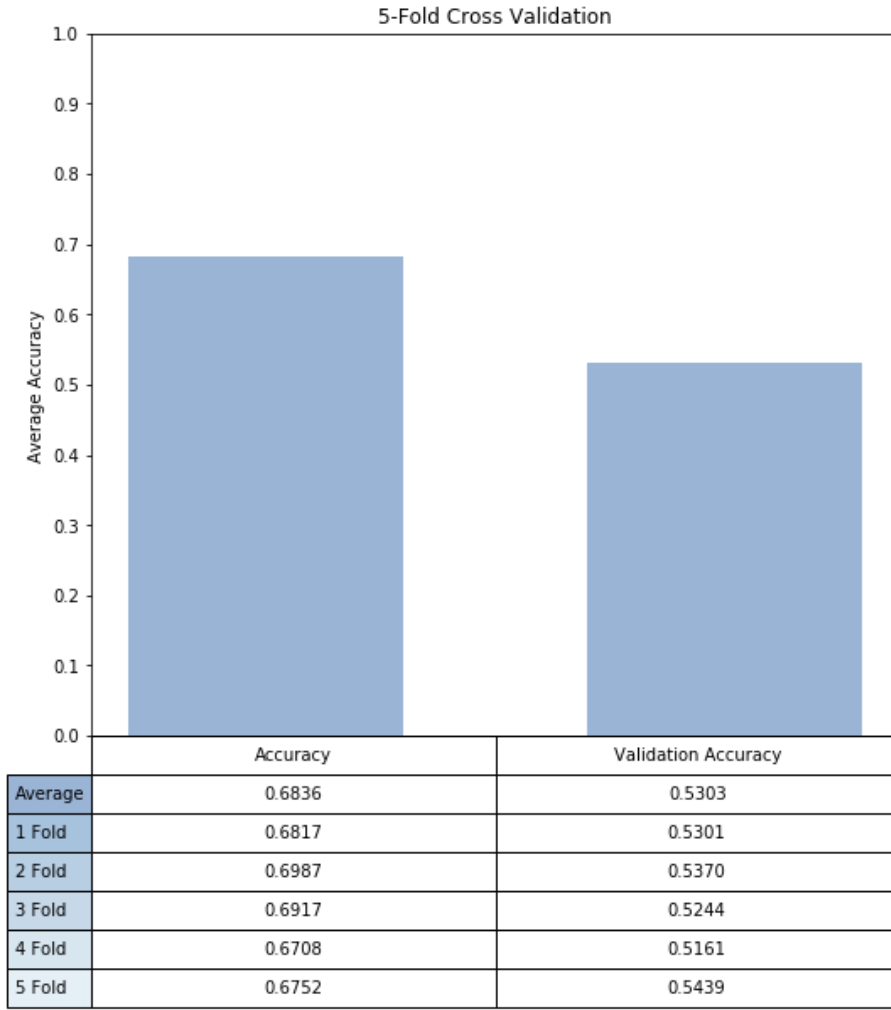
Şekil 4.1 5-Fold ile Veri Setinin Bölünmesi

K değeri 3 alındığında veri seti için elde edilen test sonuçları Şekil 4.2'de gösterilmektedir.



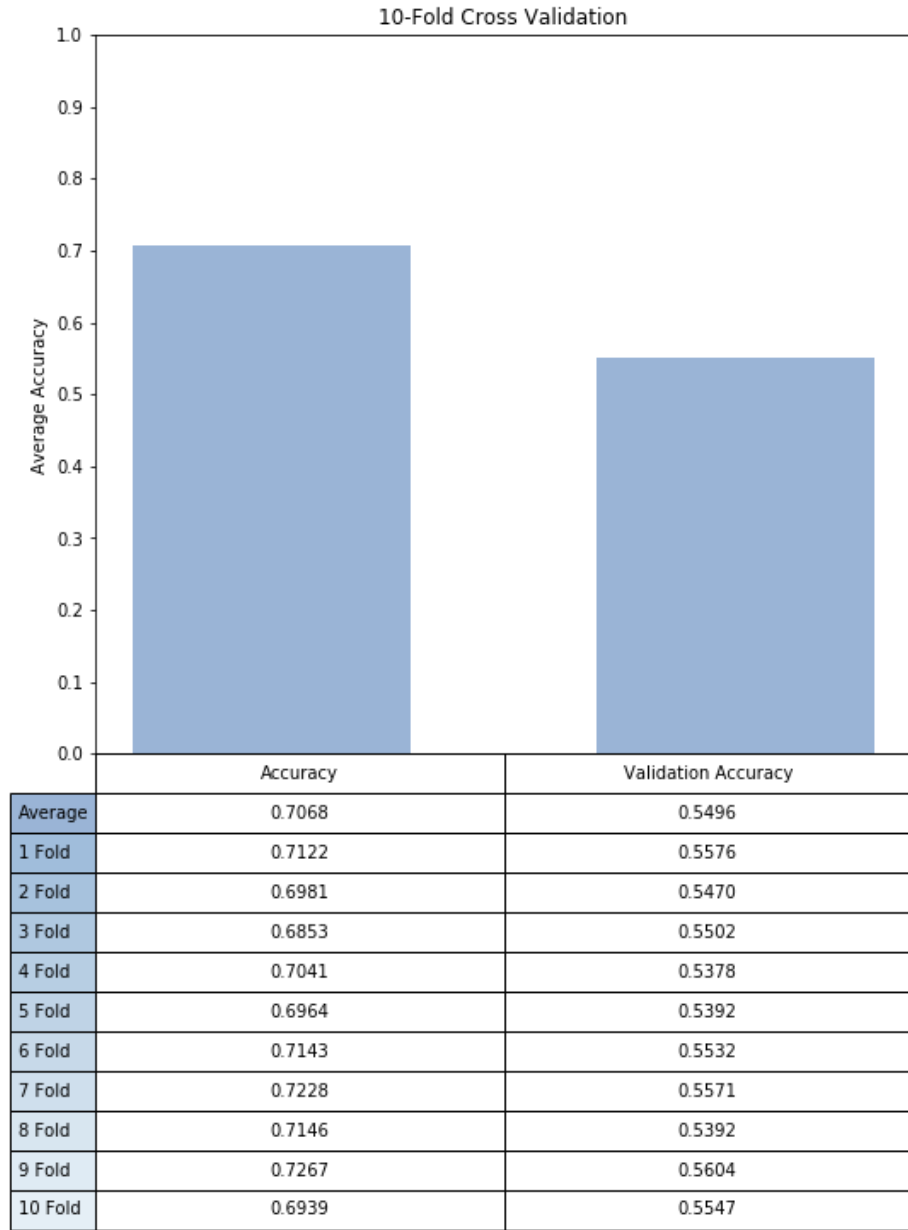
Şekil 4.2 Eğitim-Test aşaması için 3 - Fold

K değeri 5 alındığında veri seti için elde edilen test sonuçları Şekil 4.3'te gösterilmektedir.



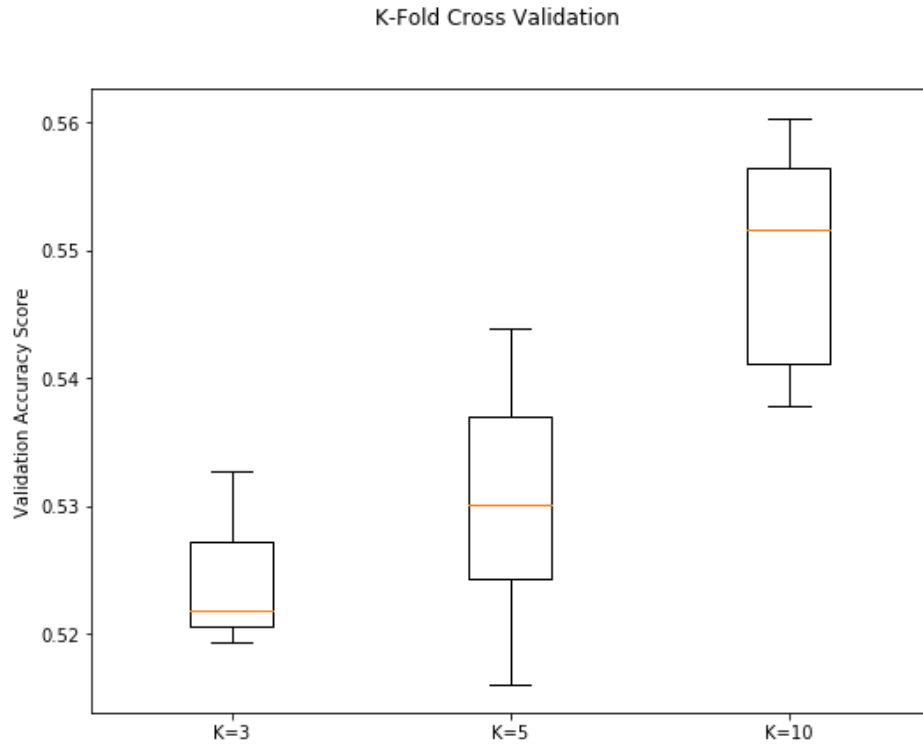
Şekil 4.3 Eğitim-Test aşaması için 5 - Fold

K değeri 10 alındığında veri seti için elde edilen test sonuçları Şekil 4.4'te gösterilmektedir.

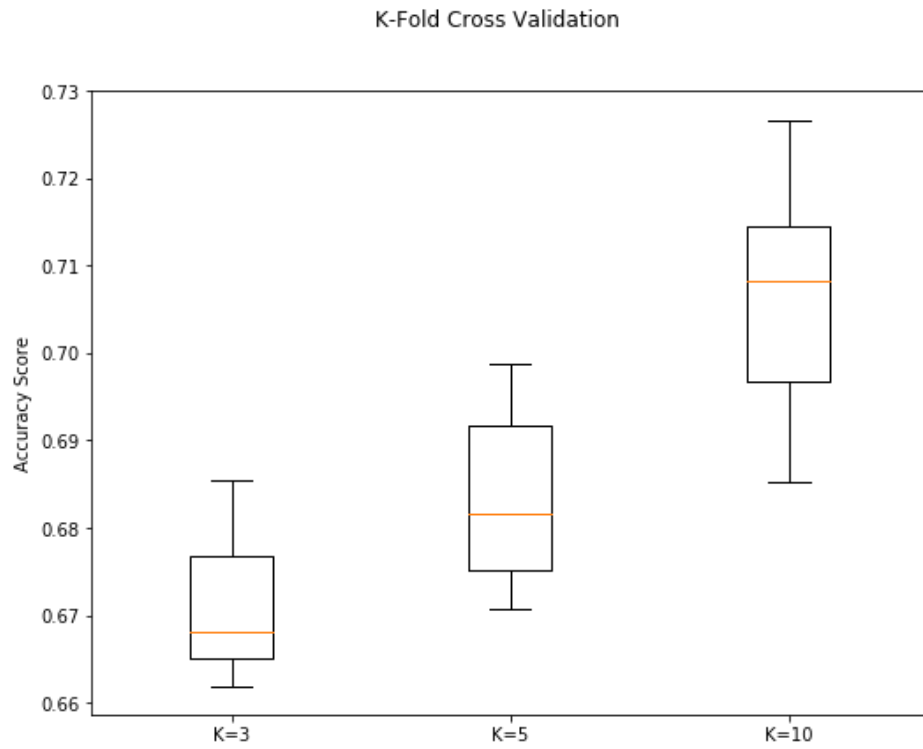


Şekil 4.4 Eğitim-Test aşaması için 10 - Fold

Farklı K değerleri için elde edilen başarımların topluca karşılaştırması Şekil 4.5 ve Şekil 4.6’da gösterilmektedir.



Şekil 4.5 K-Fold Cross Validasyon Başarım Karşılaştırması



Şekil 4.6 K-Fold Başarım Karşılaştırması

KAYNAKÇA

- Eight, F. (2016, Temmuz 15). Emotion in Text. (Erişim Tarihi: 3 Ocak 2019),
<https://www.figure-eight.com/data-for-everyone/>
- Hochreiter, S., ve Schmidhuber, J. (1997). LONG SHORT-TERM MEMORY. (Erişim Tarihi: 12 Mart 2019), <https://www.bioinf.jku.at/publications/older/2604.pdf>
- Lenc, L., ve Hercig, T. (2016, September). Neural Networks for Sentiment Analysis in Czech. IN ITAT (pp. 48-55).
- Birjali, M. Beni-Hssanea, A. Erritali, M. (2017). Machine Learning and Semantic Sentiment Analysis based Algorithms for Suicide Sentiment Prediction in Social Networks. Procedia Computer Science 113, (pp. 65-72).
- Norvig, P. (2016, Ağustos). Spell Correct. (Erişim Tarihi: 23 Şubat 2019),
<https://norvig.com/spell-correct.html>
- Olah, C. (2015). Understanding LSTM, (Erişim Tarihi: 11 Mart 2019),
<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- Özbek, A. (2019, Ocak 12). Stemming ve Lemmatization, (Erişim Tarihi: 18 Şubat 2019),
<https://anilozbek.blogspot.com/2019/01/stemming-ve-lemmatization.html>
- Press, C. U. (2008). Stemming and Lemmatization. (Erişim Tarihi: 19 Şubat 2019),
<https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html>
- Tartir, S. Abdul-Nabi. I. (2017). Semantic Sentiment Analysis in Arabic Social Media. Journal of King Saud University-Computer and Information Sciences, 29(2), (pp. 229,233)
- Toprak, B. (2005, Eylül 12). Yapay sinir ağları. (Erişim Tarihi: 9 Mart 2019),
<http://www.wikizero.biz/index.php?q=aHR0cHM6Ly90ci53aWtpcGVkaWEub3JnL3dpa2kvWWFwYXlfc2luaXJfYcSfbGFyxLE>

ÖZGEÇMİŞLER

04.10.1996 tarihinde Samsun’da doğdu. İlköğrenimi Altınyıldız İlköğretim okulunda tamamladı. Okyanus Koleji bursunu kazandı ve ortaöğrenimi burada tamamladı. Buradaki eğitimini tamamladıktan sonra Lise öğrenimi için Dede Korkut Anadolu Lisesini kazandı. Lise öğreniminin son senesinde Hüseyin Yıldız Anadolu Lisesine geçerek lise eğitimini tamamladı. Lise öğrenim sürecini tamamladıktan sonra Ondokuz Mayıs Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümünü kazandı. Bu süreç boyunca 2 zorunlu staj yaptı. Stajlarının ikisini de Uyumsoft Bilgi Sistemleri ve Teknolojileri Tic. A.Ş’de tamamladı.

Tunahan Yetimoğlu

07.09.1994 tarihinde Trabzon’da doğdu. İlköğrenimini Şehit Yüzbaşı Cengiz Topel İlköğretim Okulunda tamamladı. Lise öğrenimini Trabzon Mesleki ve Teknik Anadolu Lisesinde tamamladı. 2014’te Ondokuz Mayıs Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümünü kazandı. Stajını 2018’de Solus Bilişim Teknoloji Ticaret A.Ş.’de tamamladı.

Cafer Ümit Salman

1995 yılında Ankara'da doğdu. İlk ve ortaöğrenimini 2001-2009 yıllarında Ankara Tevfik İleri İlköğretim Okulunda tamamladı. Ortaöğretiminde PYBS bursu kazandı ve okulunu Tübitak Matematik Olimpiyatlarında temsil etti. 2009-2012 yılları arasında Ankara Yavuz Sultan Selim Anadolu Lisesinde öğrenim gördü. Bu yıllarda okulunda robotik ve satranç kulüplerinde faaliyetlerde bulundu ve görev aldı. Lise öğrenimini 2013 yılında Ankara Kurtuluş Anadolu Lisesinde tamamladı. 2014 yılında Ankara Turgut Özal Üniversitesinde lisans öğrenimine başladı. 2015 yılında Ankara Yıldırım Beyazıt Üniversitesine ve 2016 yılında Samsun Ondokuz Mayıs Üniversitesine geçiş yaptı. Lisans eğitimini 2019 yılında Ondokuz Mayıs Üniversitesinde tamamladı. Üniversite yıllarında Siber Güvenlik, Robotik, Özgür Yazılım topluluklarında yöneticilik yaptı ve kulüpler adına GNU/Linux sistem yönetimi, mobil programlama, web programlama, robotik tasarım, programlama eğitimleri verdi. 2016 yılında Turkcell Geleceği Yazanlar Elçisi seçildi ve 2017 yılında Geleceği Yazan Kadınlar projesinde Android uygulama geliştirme eğitmeni olarak görev aldı. 2016 yılında Arçelik Bulaşık Makinesi Fabrikasında enerji analizör cihazları ve otomasyon sistemleri geliştirme üzerine staj yaptı. 2017 ve 2018 yılında Özgür Yazılım A.Ş.'de şirketin özgür yazılımlarından Tekir Ön muhasebe ve yönetim sistemi ile beraber Telve Framework yazılımlarında stajyer olarak geliştirici oldu. 2018 yılında Solus Bilgi Teknolojileri A.Ş.'de GNU/Linux sistem yöneticisi olarak staj yaptı. Staj sonrası Solus Bilgi Teknolojilerinde IBM sistem çözümleri ve danışmanı olarak işe başladı ve halen aktif olarak burada çalışmaktadır.

Erdem Şahin Uslu