

SIRADAN SIRAYA LSTM MODELİ İLE CHATBOT TASARIMI

Yazar 180202114 .

Gönderim Tarihi: 28-May-2022 11:38PM (UTC+0300)

Gönderim Numarası: 1846044652

Dosya adı: 180202114_180202061_180202023.pdf (1.09M)

Kelime sayısı: 7098

Karakter sayısı: 49497

²¹
KOCAELİ ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

LİSANS TEZİ

SIRADAN SIRAYA LSTM MODELİ İLE CHATBOT TASARIMI

**OĞUZHAN TAYLAN
İBRAHİM PAMUK
ZİYA ÇETİN**

KOCAELİ 2022

KOCAELİ ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

SIRADAN SIRAYA LSTM MODELİ İLE CHATBOT TASARIMI

OĞUZHAN TAYLAN
İBRAHİM PAMUK
ZİYA ÇETİN

14. of.Dr. Sevinç İLHAN OMURCA
Danışman, Kocaeli Univ.

.....

Dr. Öğr. Üyesi ALEV MUTLU
Jüri Üyesi, Kocaeli Univ.
Arş Gör. Gamze KORKMAZ
ERDEM
Jüri Üyesi, Kocaeli Univ.

.....

.....

Tezin Savunulduğu Tarih: 20.05.2022

ŞEKİLLER DİZİNİ

Şekil.1 Yapay Zeka Dalları.....	4
Şekil.2 LSTM Mimarisi.....	8
Şekil.3 Alma Tabanlı Sohbet Robotları İçin Süreç Mimamisi Modeli.....	10
Şekil.4 Doğrulama setindeki doğrulama puanı arasındaki korelasyon.....	12
Şekil.5 İleri Ve Geri Bilgi Akışını Tanımlamak İçin Gizli Katman İle BRNN'Ye Kaynak Girişi Gösterimi.....	14
Şekil.6 Erişim Tabanlı Sohbet Robotları İçin Süreç Mimarisi Modeli.....	16
Şekil.7 Sinir Ağı.....	17
Şekil.8 RNN'in "Yuvarlanmış" Gösterimi.....	19
Şekil.9 Bir Girdinin Bir Çıkıtıya Eşlenmesi.....	20
Şekil.10 Bir Girdinin Birden Çok Çıkıtıya Eşlenmesi.....	21
Şekil.11 Birden Fazla Girdinin Bir Çıkıtıya Eşlenmesi.....	21
Şekil.12 Birden Fazla Girdinin Birden Fazla Çıktıyla Eşlenmesi.....	21
Şekil.13 Birden Fazla Girdinin Birden Fazla Çıktıyla Eşlenmesi.....	22
Şekil.14 Sigmoid: $g(x) = 1/(1 + e^{-x})$ formülüyle temsil edilir.....	23
Şekil.15 Tanh: $g(x) = (e^{-x} - e^{+x})/(e^{-x} + e^{+x})$ formülüyle temsil edilir.....	23
Şekil.16 Hazırlık Aşaması.....	27
Şekil.17 Fonksiyona giren string yapısını istediğimiz biçimde çeviren kod parçası.....	28
Şekil.18 Fonksiyon Çalışmadan Önceki Data.....	28
Şekil.19 Foksiyon Çalıştıktan Sonraki Data.....	28
Şekil.20 Sözlüğümüzde olmayan kullanıcı girdisine tagi ekleyen kod parçası.....	29
Şekil.21 Cevap Datamızın Başlangıcını Ve Sonunu Etiketleyen Kod Parçası.....	29
Şekil.22 Özel keyler barındıran sözlük datamızı oluşturmamıza yardımcı kod parçası.....	30

Şekil.23 Girdilerimizi Input hale getiren Input katmanımızın kod parçası.....	31
Şekil.24 Padding katmanı, encoder-decoder input oluşturma kod parçası.....	31
Şekil.25 dec_lstm ve enc_lstm oluşturma kod parçası, hücre sayısı 400.....	32
Şekil.26 Dense katmanı için “dense” tanımlandı, “dec_op” parametre olarak verilerek katmanımız oluşturuldu.....	33
Şekil.27 Decoder inputunun oluşturulduğu, bu inputlar ile output statelerinin oluşturulmasıyla decode modelin tamamlandığı kod parçası.....	34
Şekil.28 Input katmanında oluşturduğumuz encoder input datamızı embed edip, olusturdugumuz LSTM modele yolluyoruz. Gerekli encoder output stateleri elde ediyoruz.....	34

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler

<EOS> : End Of Sequence(Sıranın Sonu)

<PAD>: Pad Sequence(Pad Sırası)

<OUT>: Vocab Datasında Olmayan Kelime Tagı

<SOS>: Star Of Sequence(Sıranın Başı)

Kısaltmalar

LSTM: Long Short-Serm Memory (Uzun Kısa Süreli Bellek)

NLP: Natural Language Processing (Doğal dil işleme)

RNN: Recurrent Neural Network (Yinelemeli Sinir Ağı)

BRNN: Bidirectional Recurrent Neural Networks(Çift Yönlü Tekrarlayan Sinir Ağları)

23

CBOW: Continuous Bag of Words(Sürekli Sözcük Torbası)

18

TF-IDF: Term Frequency — Inverse Document Frequency(Terim Frekansı ve Ters Belge Frekansı) Tagging: Part-of-Speech Tagging(Metin Parçası Etiketleme)

NER: Named-Entity Recognition(Adlandırılmış Varlık Tanıma)

NLU: Natural Language Understanding(Doğal Dil Anlama)

OCR: Optical Character Recognition(Optik Karakter Tanıma)

AIML: Artificial Intelligence Markup Language(Yapay Zeka İşaretleme Dili)

IVR: İnteractive Voice Response(İnteraktif Sesli Yanıt)

HCI: Human Computer İnteraction(İnsan-Bilgisayar Etkileşimi)

TPU: Tensor Processing Unit(Tensör İşleme Birimi)

SIRADAN SIRAYA LSTM MODELİ İLE CHATBOT TASARIMI

ÖZET

Bu çalışmanın amacı kullanıcı tarafından 7/24 kullanılabilecek yapay zekâ ile güçlendirilmiş sohbet tasarımlı yapmaktadır.

1 Sohbet botu (Chatbot), işitsel veya metinsel yöntemlerle kullanıcı ile sohbet etmeye yarayan bir yazılımdır. Gelişmiş sohbet botları, ilgili alanda sorulan sorulara cevap vererek veya sohbet ederek insan davranışında bulunabilirler. Bu yakınsama önemlidir çünkü botumuzun en önemli görevi karşısındaki kullanıcıya olabildiğince insan-insan ilişkisine yakın bir hissiyat vermektedir. **Bu kapsamda**, çalışmamızda **LSTM (Uzun Kısa Süreli Bellek)** ve **seq2seq modeli** kullanılarak bir **bot uygulaması** gerçekleştirdik. Çalışmada Movie Dialog Corpus veri kümesi kullanılmıştır. Kullanıcı etkileşimi için flask bağlantısı ile telegram ile web arayüz oluşturulmuştur. **Çalışmadaki kayıp oranı her adımda azalarak 50 adım sonunda 0.5621'e düşmüş ve yüzde 85 doğruluk oranı elde edilmiştir.** **10** Çalışmada, açık kaynak kodlu ve ücretsiz yazılımlar kullanılmıştır. Sunulan çalışma, güncel teknolojilerin kullanıldığı literatürde öne çıkan çalışmaların özelliklerinin bir araya getirilmesini sağlamıştır.

Anahtar kelimeler: CHATBOT, LSTM, seq2seq, NLP

CHATBOT DESIGN WITH SEQUENCE 2 SEQUENCE LSTM

ABSTRACT

¹⁹
The aim of this study is to design a chat powered by artificial intelligence that can be used 24/7 by the user. Chatbot (Chatbot) is a software that is used to chat with the user through auditory or textual methods. Advanced chatbots can engage in human behavior by chatting or answering questions asked in the relevant area. This convergence is important because the most important task of our bot is to give the opposite user a feeling as close to the human-human relationship as possible. In this context, we implemented a bot application using LSTM (Long Short Term Memory) and seq2seq model in our study. The Movie Dialog Corpus dataset was used in the study. For user interaction, a web interface was created with flask connection and telegram. The loss rate in the study decreased with each step and decreased to 0.5621 at the end of 50 steps and an accuracy rate of 85% was obtained. The presented study has brought together the features of prominent studies in the literature using up-to-date technologies.

Keywords: CHATBOT, LSTM, seq2seq, NLP

GİRİŞ

¹⁵ Sohbet botu, kullanıcı ile genellikle metin, bazı durumlarda ise konuşma yoluyla diyalog kurarak bilgi veren veya bir işlemi gerçekleştiren bir yazılımdır. Sohbet botu ³ terimi ilk olarak “chatter bot”[24] olarak Michael L. Mauldin tarafından 1994 yılında kullanılmıştır.

¹⁷ Sohbet botları, AI sistemlerinin tipik bir örneğidir ve akıllı İnsan-Bilgisayar Etkileşiminin (HCI) en temel ve yaygın örneklerinden biridir [5]. Sohbet botları, çeşitli alanlarda hedef kullanıcılarla verimli bir şekilde diyalog sağlamak amacıyla anlık mesajlaşma frameworklerini kullanan yazılımlardır. Sohbet botları, herhangi bir çevrimiçi olma durumu veya son görülmeye bilgileri olmadığı için insan kullanıcılarından tamamen farklıdır [6]. Bu sohbet botlarından bazıları Gelişmiş Doğal Dil İşleme Sistemlerini (NLPS) kullanır fakat bir kısmı da sadece giriş metinindeki anahtar kelimeleri tarayıp onlara odaklanarak cevap üretir. Sohbet botları halihazırda gelişim sürecinde olan bir teknoloji olduğu için diyalog kalitelerinde büyük farklılıklar gözlemlenmektedir [7]. İnsanlar gibi sohbet botları da kendi bilgilerini ortama dayalı olarak, anlamak için kullanabilirler. Böyle bir sistem, bilgisayarların hesaplama gücü avantajına sahip olarak, bazı görevlerde bir insandan bile daha başarılı olabilir [8].

³ Sohbet botları, müşterilere hızlı ve doğal bir etkileşim imkânı sunmalarını sağlama³ ve aynı zamanda verimliliği artırmaları ile şirketlerin de ilgisini çekmektedir. Günümüz müsterisi alınan hizmeti kesintisiz ve hızlı bir biçimde almak istemektedir. Sohbet botları ³ bunu sağlama vaadi ile şirketler tarafından denenmeye ve kullanılmaya başlanmıştır. Şirketlerinin gösterdiği ilginin bir diğer sebebi de Sohbet botlarının şirket verimliliğini artırma vaadidir. Günümüzde birçok sektörde daralan kar marjları ve yeni teknolojilerin sunduğu imkânlar şirketleri verimlilik arayışına itmiştir [12].

Sohbet botu tasarlarken duyulan kaygılarından birisi kullanıcılar tarafından bota duyulacak güvenin beklenenden az olmasıdır. Kullanıcının güven derecesini etkileyen faktörlere baktığımızda ise kullanıcının yaşadığı deneyimler, tasarımcıyla ilgili bilgiler ve güvenlik sorunlarından bahsedebiliriz. Bu güveni artırmak istersek de botumuzun arayüzü, insansı isim belirleme, botumuza katacağımız kişilik özelliklerini, insan dilini işleme becerisi gibi konulara göz atmak gerekir. Bunların yanında sohbet botlarına duyu katarak bilinçli bir sohbet robottu tasarlamaya yönelik de mevcuttur [9].

1

Sıklıkla kullanılan iki tür sohbet botu bulunmaktadır. İlk sohbet botları, konuyu anlamaya çalışan, aynı konu hakkında her soru için her zaman aynı cevabı veren yapıdaki sohbet botlarıdır. Genel olarak, bu tür sohbet botları cevap için sınıflandırma algoritmalarını kullanmaktadır. Bu tür bot uygulamalarında, N adet konu ve N adet cevabın listesi bulunmaktadır. Uygulamada, sınıflandırılan konunun olasılığı az ise kullanıcidan daha özel ifadeler kullanarak soruyu tekrar etmeleri istenir. İkinci türdeki sohbet botları ise daha gelişmiş, güncel teknolojileri kullanır ve aynı zamanda daha karmaşıktır. Bu botlardaki cevaplar, genellikle RNN (Tekrarlayan Sinir Ağlığı) kullanılarak oluşturulmaktadır. Bu sohbet botları daha kişiselleştirilmiş ve daha uygun cevaplar verebilmektedir. Özel bir RNN türü olan Uzun Kısa Süreli Bellek (LSTM) kullanıcının sorularını daha iyi anlayıp mümkün olan en iyi cevabı üretmektedir. Geleneksel seq2seq sohbet bot modelleri, çıktı cümlelerinin duyarlılığını düşürmeden sadece girdi dizilerinde şartlandırılmış en yüksek olasılıklara sahip cümleleri bulmaya çalışmaktadır [2, 3].

İlgili bilgiler ile ihtiyaç duyulan cevap arasındaki farkın az olduğu durumlarda, RNN'ler geçmişteki bilgileri kullanmayı öğrenebilirler. Tahmin edilmesi gereken kelime sayısı çoğaldıkça, RNN'ler bilgileri birleştirmeyi öğrenemezler. LSTM'ler ise bu soruna bir çözüm olarak bilginin uzun süre hatırlanması için tasarlanmıştır [2]. Çalışmada LSTM ve seq2seq TensorFlow Kütüphanesi kullanılarak bir sohbet botu tasarımı yapılmıştır.

1. GENEL BİLGİLER

Günümüzde teknolojinin gelişimi ile beraber yavaş yavaş yapay zeka kavramını hayatımıza her alanında görmeye başladık. Yapay zekaların artması ile birlikte şirketlerin iş gücünü artırmak, maliyeti azaltabilmek ve zamanından tasarruf edebilmek gibi temel ihtiyaçlarından bazıları kolayca halledilebilmekte. ¹³ Yapay zeka, görevleri yerine getirmek için insan zekasını taklit eden ve topladığı bilgilerle kendisini seviyeli olarak ilerletebilen sistemler veya makineler anlamına gelir. Yapay zekanın amacı insanın yerini almak değil insanların yeteneklerini geliştirmek ve katkıda bulunmaktadır. Yapay zekanın birçok kullanım alanı vardır. Örneğin;

- Sohbet robotları
- Akıllı asistanlar
- Öneri robotları vb.

Yukarıda belirtilen ve daha birçok farklı alanda karşımıza çıkan yapay zeka ticaret içinde oldukça değerli bir varlıktır.



Şekil.1 Yapay Zeka Dalları [1]

1.1. Sohbet Botu Teknolojisi

³

Sohbet botları yaklaşık 2016'dan bu yana yoğun şekilde konuşulmaktadır. Ancak yapay zeka konusundaki bilimsel araştırmalar 1950'lere kadar gitmektedir. 1950 yılında Alan Turing tarafından Mind dergisinde bir seminer raporu yayınlanmış, yazında bugün yapay zekanın ilk örneği olarak kabul edilen Turing testinden söz edilmiştir.[2] Turing testi bir bilgisayar programının, insan ve yazılım ile iletişime geçmesini ve hangisinin insan, hangisinin yazılım olduğunu ayırt etmesine dayanan bir testtir. Eğer denenler, insan ve yazılımı birbirinden ayıramazsa yazılım testi geçmiş sayılmaktadır.

İlk sohbet botu olarak ise 1966 yılında bir MIT profesörü olan Joseph Weizenbaum tarafından geliştirilen “ELIZA” olarak kabul görmektedir. ELIZA bir psikolog olarak yaratılarak insanı diyaloglar kurması amaçlanmıştır. Fakat makine öğrenmesi ve doğal dil işleme teknolojileri henüz yeterince ilerlememişti için, o dönemde sadece kelime eşleştirerek yanıt oluşturabilmiştir. Chatbot terimi ilk olarak “chatter bot”[3] olarak Michael L. Mauldin tarafından 1994 yılında kullanılmıştır.

Sohbet botları dijital ortamda kullanıcıya bilgi vermek, bir görevi yerine getirmek, çeşitli konularda işlem yapabilmek gibi kullanıcıya yardım etmek amacıyla ile çalışan yapay zekalardır. Amaçları doğrultusunda zaman ve enerji tasarrufu sağlamaktadırlar.

Sohbet botlarını kullanım amaçlarına göre sınıflara ayırmak mümkündür. Örneğin hangi platformda yayınlanacağı, hangi amaç doğrultusunda geliştirildiği veya hangi teknoloji ile geliştirildiği konusunda sınıflandırabiliriz.

1.1.1 Kullanılan Sohbet Bot Türleri

Kelime bazlı sohbet botları: Bu tür sohbet botlarda, bir yapay zeka teknolojisinden söz etmek mümkün değil. Kullanıcı mevcut menü ve butonlardan seçimler yaparak veya bazı ifadeler yazarak ilerliyor. Ön yüzde menü ve butonlar, arka tarafta da daha çok IVR sistemlerinde gördüğümüz karar ağaçları kullanılıyor. Sistem, yazılan ifadelerdeki bazı kelimeleri tanııp, uygun bir cevap ile eşleştirmeye çalışıyor. Bu tür sohbet botları, kısıtlı sayıda ve türde soruyu yanıtlayabiliyor. Ancak kelime bazlı sohbet botları, yazılan ifadeyi bir bütün olarak anlamlandırmada yetersiz kalıyor. Kelimelerin yanlış yazımında kullanıcıyı doğru yönlendirebilmek için olası tüm yanlış yazımların sisteme tanıtılması gerekiyor. Çok basit soru-cevap sohbet botlarında kullanılabilirler ancak karmaşık durumlarda çalışmazlar. Bir örnekle açıklık getirmek istenirse, yazılımda İngilizce-Türkçe dönüşümü tanımlanmış olduğunu varsayıyalım. Kelime bazlı bir sohbet bota kullanıcı “Hi Türkcede ne demek?” sorusunu sordduğunda, sohbet bot “hi” kelimesini tanyarak Türkçe karşılığını verir, ifadenin gerçekten ne sorduğunu anlayamaz.

İfade bazlı sohbet botları: Bu tür sohbet botlarda bir NLP ve makine öğrenmesi teknolojisinden söz etmek mümkün. Sohbet bot, kullanıcının yazdığı ifadeyi bir bütün olarak tamıyor ve anlamlandırmıyor. Ancak karşılıklı konuşma devam ettiğinde, diyaloga bağlı kalamıyor. Dil dönüşümleri ile ilgili örnek soru üzerinden devam edecek olursak, bu tür bir sohbet bot “Hi Türkcede ne demek?” sorusunu bir ifade olarak anlıyor, ancak ardından sorulan “Peki, Almancada ne demek?” sorusunu anlamada yetersiz kalıyor. Çünkü bu noktada diyalogu başından itibaren izleme ve bağlama uygun şekilde sürdürme yetkinliği devreye giriyor.

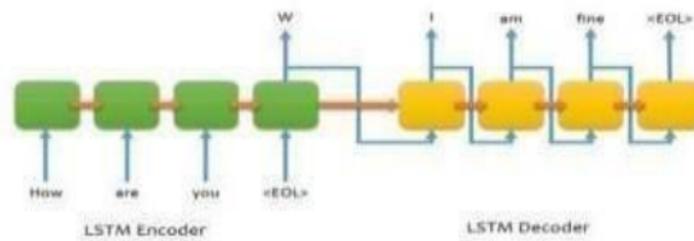
3. Diyalog bazlı sohbet botları: Bu tür sohbet botları **da yapay zeka tabanlı** sohbet botları **ve NLP** (Doğal dil işleme) ve makine öğrenmesi / derin öğrenme teknolojilerini kullanarak geliştiriliyorlar. Diyalog bazlı sohbet botları **kelime veya ifade tabanlı** bir anlayışla sınırlı kalmıyor, yazılan ifadeyi anlamlandırbildikleri gibi, konuşmanın akışını takip edip, **bu** akışına göre **cevaplar da** üretebiliyor. Aynı örneği ² bir de diyalog bazlı sohbet bot için verirsek, “Hi Türkçede ne demek?” sorusunu ifade olarak anlayıp cevap verir, ardından sorulan “Peki Almancada ne demek?” sorusunda neden söz edildiğini, nasıl bir dönüştürme yapacağını anlar ve doğru sonucu çıkartır.

Bu 3 türden hangisi daha iyi diye sorulursa öncelikle kullanım amacına bakmak gerekmektedir. Örneğin bir inşaat şirketine etkileşim alanı sınırlı olacağı için ifade bazlı bir sohbet bot **yeterli olabilir**. Sohbet bot **türüne karar vermek** için kullanıcı için yaratacağı değere bakmak da önemli. Eğer kullanıcı için yaratılacak değer, kelime bazlı veya ifade bazlı bir sohbet bot **ile sağlanıyorsa**, diyalog bazlı bir sohbet bot geliştirmek anlamlı olmayabilir. Bir diğer parametre, hedef kullanıcıların tercihleri. Hedef kitleyi çok iyi tanımak ve onların bekłentilerine uygun bir sohbet bot geliştirmek önemli. Bazı kullanıcı tipleri menülerle ilerlemek isteyebilir, bazı gruplar bir diyalogu serbest şekilde sürdürmeyi tercih edebilir. Bu 3 etmene dikkat ederek sohbet bot **türünü belirlemek en uygun çözüm**.

² Sonuç olarak **bu 3** amaca en iyi şekilde hizmet verecek sohbet botu **bulmak** için önce kendi stratejiniz **içinde** sohbet botlarının **yerinin belirlenmesi**, sohbet bottan **nasıl bir soruna çözüm** getireceğinin tanımlanması **ve buna göre nasıl bir sohbet bot istendiğinin ortaya konması** gerekiyor. Ayrıca botun gelişmişlik seviyesi de önemlidir. Botun çok daha efektif kullanılması için nasıl bir seviye istendiği de belirlenmelidir.

2. LİTERATÜR

5 Derin öğrenme, yapay zekâ ve makine öğrenmesi alanlarında çok önemli bir konu haline gelmiştir. Son yıllarda, farklı derin öğrenme yöntemlerini önermekte olan çalışmaların ve mevcut yöntemleri değişik problemler üzerinde uygulayan çalışmaların sayısı hızla artmaktadır. 7 Derin Öğrenme Yöntemleri; Derin Sinir Ağları (Deep Neural Networks), Derin Oto-Kodlayıcılar (Deep Autoencoders), Derin İnanç Ağları (Deep Belief Networks), Derin Boltzmann Makinesi (Deep Boltzmann Machine), Yinelenen Sinir Ağları (Recurrent Neural Networks) , Evrişimsel Sinir Ağları (Convolutional Neural Networks)ndan 20 olmaktadır. Otomatik soru cevaplama sistemi için Evrişimsel Sinir Ağları (Convolutional Neural Networks) ve 5 LSTM (Long Short-Term Memory) kullanılmaktadır. Evrişimsel sinir ağları, çok boyutlu girdiler için ve özellikle iki boyutlu görsel veriler için önerilmiş bir derin öğrenme yöntemidir. Otomatik Soru Cevaplama Sistemi; doğal dilde ifade edilen sorunun cevabını doğal dildeki büyük veri kümeleri içerisinde arayıp bularak yine doğal dilde otomatik olarak cevaplandırmayı planlayan sistemdir [1].



Şekil.2 LSTM Mimarisi [1]

¹ Chatbot, işitsel veya metinsel olarak kullanıcı ile sohbet edebilen bir yazılımdır. Gelişmiş chatbotlar, ilgili konularda gerekli cevapları verebilmektedir. Chatbotta yapay zekâ kullanılması verimliliği yükseltmektedir. Sıradan Sıraya Modeli (Sequence to Sequence Model): RNN mimarisine dayanmaktadır. İki adet RNN den oluşmaktadır. Bu iki RNN kodlayıcı ve kod çözücüdür. Kodlayıcı girdiyi kontrol etmekte olup girişte bir dizi alıp her adımda bir simbol işlemektedir. Bundaki amaç gereksiz bilgileri kaybedip her adımda önemli bilgileri kodlayan bir simbol dizisini sabit boyutlu bir özellik vektörüne dönüştürmektedir. • Kelime Gömme (Word Embedding): Düşük boyutlu bir vektör uzayda kelimelerin yoğun temsilini öğrenmek için uygulanan tekniktir. • LSTM Modeli (Long Short-Term Memory): Basit RNN' nin giriş ve çıkış kapılarına ek olarak, özel unutkan kapıları olan özel bir RNN çeşididir. Bu kapsamında, yapılan çalışmada LSTM (Uzun Kısa Süreli Bellek) ve seq2seq modeli ile telegram bot uygulaması geliştirilmiştir. Çalışmada LSTM kullanılması bir sonraki konuşma eylemini tahmin etmek için konuşma geçmişine geri dönülebilmesini sağlamıştır. [26]

Akıllı sohbet botlarının insan hayatına adapte oluşu yaygınlaşıkça, insanların bu bottardan bekłentileri değişimektedir. Görev gerçekleştirmeye amacı güden sohbet robotlarına, bu görevlerin üstüne günlük diyalog ve arkadaşlık gibi görevler de eklenebilmektedir. Kullanıcı ihtiyaçlarını karşılamak ve kullanıcılarla insan-insan benzeri bağ kurmak için, sohbet robotlarının daha insancıl ve gelişmiş dil özelliklerini içermesi beklenmektedir. Bu dil özelliklerinden birisi de metafor kullanımıdır. Bu konuda yapılan bir çalışmada, konuya duyarlı ve konu hakkında metafor üretebilen sohbet robotu tasarlamak hedeflenmiştir. Yine bu çalışmada insan gözlemcilere, üretilen metaforların yeniliğini ve uygunluğunu sorgulatılmıştır. Bu çalışmada deneyler, kurulan sistemin sohbet robotumuzla iletişim kurma konusunda kullanıcıların ilgisini etkin bir şekilde uyandırdığını ve bunun sonucunda önemli ölçüde daha uzun insan-chatbot sohbetleri sağladığını gösteriyor [15].

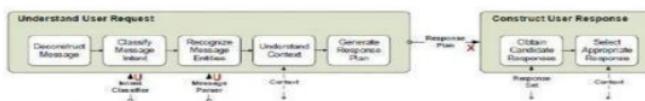
Canlı sohbet arayüzleri aracılığıyla müşterilerle iletişim kurmak, e-ticaret şirketlerinin müşteri hizmeti sağlamak için başvurduğu yöntemlerden biridir. Günümüzde, arka planda müşteri hizmet çalışanının olduğu sohbet hizmeti araclarının yerini, genellikle yapay zekaya dayalı doğal dil aracılığıyla insan kullanıcılarla iletişim kurmak için tasarlanmış yazılımlar olan sohbet botu aracları almaktadır. Maliyet, zaman ve iş gücünden tasarruf sağlayan bu çözüm, yapay zeka tabanlı sohbet robotlarının hızla yayılmasını sağlamış olsa da yine de müşteri beklenilerini karşılamada başarısız olabildiklerini ve müşterilerin sohbet robotu tarafından yapılan taleplere uymaya daha az meyilli olabildiklerini gözlemlenebilmektedir. Bağlılık-tutarlılık teorisinden yardımıyla, sözlü antropomorfik tasarım ipuçlarının ve foot-in tekniğinin kullanıcı isteği uyumluluğunu nasıl etkilediğini çevrimiçi bir deney aracılığıyla incelemiştir. Alınan sonuçlara göre , hem antropomorfizmin hem de tutarlı kalma ihtiyacının, müşterilerin bir sohbet robotunun hizmet geri bildirim talebine uyma olasılığını kayda değer ölçüde artırdığını göstermiştir. [4]

Derin Öğrenme, Makine öğrenimini orijinal ve ana hedeflerinden birine, yani Yapay zekâya yaklaşımak amacıyla tanıtılan bir Makine Öğrenimi ve araştırma alanıdır. Otomatik öğrenme algoritmasından bahsedersek, bunlar doğrusal olma eğilimindedir, DL (Derin Öğrenme) algoritmaları karmaşıklığı ve soyutlamayı artıracak şekilde yapılandırılmıştır. Derinlemesine öğrenmek için ilk kelimesi kedi olan 5 yaşında bir çocuk düşünüm. Çocuk, nesneleri göstererek ve kedi kelimesini söyleyerek kedinin ne olduğunu öğrenmeye devam eder. Anne, "Evet, o bir kedi" veya "Hayır, o bir kedi değil" der. Çocuk nesneleri işaret etmeye devam ettikçe, tüm kedilerin özelliklerinin ve özelliklerinin daha çok farkına varır. [3]

Yapılan bir çalışmada, lokanta ortamında müşterilerden gelen soruları okuyan, her bir kelimeyi analiz eden ve ilgili anahtar kelimelere göre cevabı oluşturmaya çalışan bir sohbet robottu tasarlamak amaçlanmıştır [14].

Yapılan bu çalışmaya göre sohbet robottu, siparişleri almaya ve restoran hakkında daha fazla bilgi vermeye hazır garsonu temsil etmektedir. Kullanıcı; Menüden bir veya daha fazla yemek siparişi verebilir, restoranın günlük açılış ve kapanış saatleri hakkında bilgi alabilir, restoranın sipariş alma ile ilgili bilgilerini alabilir, siparişlerini içeren tam listeyi alabilir. Bu çalışmadan yola çıkarak da sohbet robotlarının kullanım alanlarının ne denli geniş olduğu sonucuna varabiliziz.[14]

Bir sohbet-bot geliştirme çalışmasında genellikle son aşama, botu istenilen ortama entegre etmektir. Bu aşama temel olarak, botu entegre ettiğiniz ortamda kullanıcıların botla iletişim kurmaya başlamasına izin vermenizdir. [13]. Botunuzu oluştururken halihazırda hedef ortamınızın belli olması gerekmektedir, çünkü farklı ortamlar farklı tipte diyalogları gerçekleştirebilir ve farklı özellikleri destekleyebilir. Örneğin, Microsoft Teams, Uyarlanabilir Kartlar gibi farklı içerikleri işleyebilirken Twilio metin mesajları için kullanılır. Bu verilen örnekte, metin mesajları Uyarlanabilir Kartlar'ı görüntüleyemediğinden, Twilio ortamında Uyarlanabilir Kartları kullanamazsınız. Bu nedenle, sohbet-botu'nuzun ilerleyen güncellemelerinde büyük değişiklik taleplerinden kaçınmak için geliştirme aşamasında mutlaka hedef kanalları düşünmeniz gereklidir. [13]



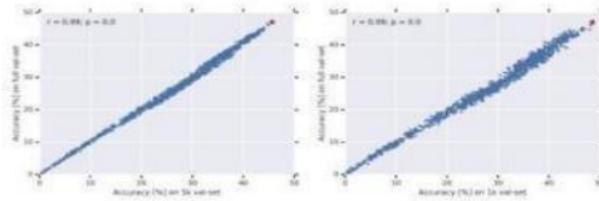
Şekil.3 Alma Tabanlı Sohbet Robotları için Süreç Mimarisi Modeli [11]

Bu çalışmada görüntü sınıflandırıcıların yarı denetimli öğrenme problemi ele alınmıştır. Ana hedef, yarı denetimli öğrenme alanının, hızla gelişen kendi kendini denetleyen görsel temsili öğrenme alanından faydalansabileceğidir. Bu iki yaklaşımı birleştirerek, kendi kendine denetimli, yarı denetimli öğrenme çerçevesi önerilmiştir ve bunu iki yeni yarı denetimli görüntü sınıflandırma yöntemini türetmek için kullanılır. [16]

NBA ve EPL (İngiltere Premier Ligi) dahil olmak üzere büyük spor ligleri, oyun bilgileri sunmak ve taraftarlarının içeriklere ulaşmalarını kolaylaştırmak için yenilikçi bir çıkış noktası olarak sohbet botlarını benimsiyor. Bununla birlikte, mevcut spor sohbet robotları, istenen istatistiksel verileri karşılamakta genellikle yetersiz olan skor ve maçın öne çıkan videoları sağlayabilmektedir. Yapılan bir çalışmada, spor hayranlarının daha fazla oyun istatistik verilerini keşfetmeleri için etkileşimli bir sohbet robotu olan GameBot adında sohbet robotu tasarılmıştır. [22]

Yinelemeli bir geliştirme süreciyle bir sohbet robotu oluşturmak, sohbet robotu geliştiricisi için belirli zorluklar doğurur. Geliştiriciden beklenen şey, kısa bir döngü sonunda botun hizmete hazır olan ilk sürümü üretmesidir. Bahsedilen dönemin her yinelenmesi, sohbet robotunun kapasitesini aşamalı olarak ilerletmeli ve bir öncelik listesine dayalı olarak genel kullanıcı öykülerinin bir alt kümesini uygulamalıdır. Bu bağlamda, ticari sohbet robotu oluşturma platformları, geliştiricinin bu kullanıcı hikayelerini belirli bir biçimde eşleyebilmesi koşuluyla, sohbet robotu geliştiricisine birçok avantaj sağlar. Bunu yapmak için, sohbet robotunun yanıtlaması beklenen her soru için geliştiricinin kullanıcının isteklerini değerlendirmesi gereklidir. Amaca bağlı olarak, sorgunun amaca hizmet edecek işlenmesi gereklidir, bu da bazı iş mantığının yürütülmesini gerektirebilir. Ek olarak, sorgunun işlenmesi, kullanıcının sohbet robotu ile konuşmanın bir parçası olarak sağlaması gereken belirli veri öğelerini gerektirebilir. Böylece sohbet botu, karşılaşabileceği bir "bağlam" ve bağlam gözlemlendiğinde gerçekleşmesi gereken "tepki" sağlanarak tanımlanır. [23]

Modern bilgisayarlı görüş sistemleri, görüntü tanıma, nesne algılama, anlamsal görüntü bölütleme vb. gibi çeşitli zorlu bilgisayarlı görüş kısaslarında olağanüstü performans sergilediği görülmektedir. Beraberinde getirdiği sonuçlar, büyük miktarda zaman alan ve elde edilmesi pahalı olan açıklamalı veriler. Birçok gerçek dünya bilgisayarlı görüş uygulaması, standart kıyaslama veri kümelerinde olmayan görsel kategorilerle veya görsel kategorilerin veya görünümlerinin zaman içinde değişimini dinamik nitelikteki uygulamalar ilgili olduğunu göstermektedir. Fakat, tüm bu senaryolar için büyük etiketli veri kümeleri oluşturmak pratik olarak mümkün değildir. Dolayısıyla, yalnızca az miktarda etiketlenmiş örnekten yararlanarak yeni kavramları tanımayı başarılı bir şekilde öğrenebilen bir öğrenme yaklaşımı tasarlamak önemli bir araştırma zorluğu olarak tanımlanmış, insanların sadece birkaç örnek gördükten sonra yeni kavramları çabucak anlamaları ve bu amaca prensipte ulaşabileceğinden bahsedilir [16].



Şekil.4: Doğrulama setindeki doğrulama puanı arasındaki korelasyon
[16]

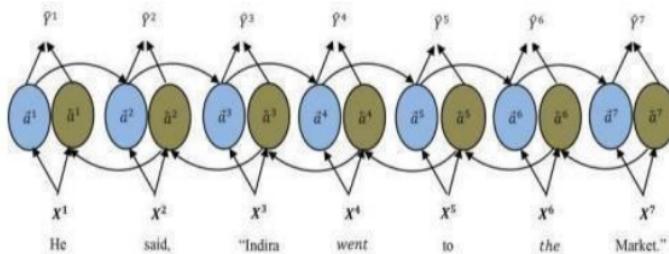
Akademik danışmanlık, öğrencilerin akademik hayatında önemli rol oynar; ancak, zaman alıcı ve sürdürülmesi zordur. Yapılan bir çalışmada, hızlı, ölçeklenebilir ve uygun maliyetli bir sohbet robottu kullanarak etkili çözüm üretme amaçlanmıştır. Bu çalışmada tasarlanan bot, Microsoft Teams ile entegre edilebilen bir bottur. Natural Language Processing'den (NLP) gelişmiş semantik analiz tekniklerini uygular ve öğrencinin sorularının bağlamını analiz eder ve buna göre yanıt verir [21].

Başlıca dephinilen konular:

- Kendi kendini denetleyen öğrenmedeki son gelişmelerden yararlanan doğal görüntülerle yan denetimli öğrenme için yeni bir teknikler ailesi önerilir.
- Kendi kendini denetleyen yarı denetimli tekniklerin, etiketlenmemiş veriler olmadan eğitilmiş dikkatle ayarlanmış temel çizgilerden daha iyi performans gösterdiğini ve önceden önerilen yarı denetimli öğrenme teknikleriyle rekabet edebilir performans ettiği gözlemlenmiştir.

Bir gün içerisinde çok fazla miktarda veri işlenmekte ve veri alışverişi yapılmakta ve veri tabanı uzun süredir verileri düzenlemek ve işlemek için kullanılan ve aktif olarak üzerine çalışılan bir araştırma konusudur. Veritabanı konusu birçok bilgisayar sisteminde büyük role sahiptir. Konuya ilgili kişi ya da kişilerce her zaman yapılan işlemleri kolaylaştırma ve harcanan vakti azaltma amacıyla tercih sebebidir. Bir veritabanıyla doğrudan yolla etkileşim kurmak için “doğal dil” i kullanmak, güzel ve kullanıcıya kolaylık sağlayan bir çözümüdür. İnsan ve bilgisayar arasındaki bu tarz bir iletişimini sağlayabilmek için bilgisayarın kendinden ne istendiğini anlaması ve istenilen doğrultusunda doğru cevap veriyor olabilmeliyiz. Doğal dil işleme (NLP), Doğal Dil anlama veya Doğal Dil oluşturma veya hatta ikisini birlikte gerektiren birçok uygulamaya sahiptir. Örnek olarak, metni bir insan dilinden diğerine kendiliğinden çevirmeye odaklılan Makine Çevirisi. NLIDB sistemler fikri, veritabanını sorgulamak için veritabanı dili yerine Doğal Dil sorgularını kullanan veritabanı soru yönteminden gelir. Kullanıcı, çok basit bir şekilde veritabanıyla etkileşim kurmak için Doğal Dil’i kullanabilir hatta kullanıcının bununla ilgili herhangi bir eğitime sahip olup olmaması sorun olmayacağındır. Ayrıca birden çok veritabanı tablosu içeren sorgularda Doğal Dil’i kullanmak daha kolaydır [17].

Chatbot için derin öğrenme hesaplamasındaki modellemeyi ve performansı gösterir. Neural Machine Translation için Tensorflow yazılım kitaplığının kullanımı Modelleme için data elde etmek en önemli işlemlerden biridir ve onu işleyebilmek oldukça zordur. Dikkat katmanlarını içeren Çift Yönlü Tekrarlayan Sinir Ağları (BRNN) kullanılır, böylece çok sayıda zarf içeren giriş cümlesi daha uygun bir şekilde cevaplanabilir. İnsanoğlu gerekli cevapları alabilmek için sorduğu sorularda çok detay verir. Verilen bu detaylar içerisinde chatbotların en doğru cevapları verebilmesi için BRNN ve Attention modelinin birleşimini mükemmel yapan sebep budur. BRNN yapısı şekil.2 te gösterilmektedir. Sinyaller ileri ve geri olarak 2 tiptedir. $A \rightarrow$ ileri yinelenen bileşendir ve $a \leftarrow$ geriye doğru yinelenen bileşendir [2]. Birçok aktivasyon fonksiyonu türü vardır, ancak BRNN ile en çok sigmoid ve lojistik aktivasyon fonksiyonları kullanılır. Örneğin, Şekil.2'te gösterilen ağda, birkaç ifadeden oluşan bir girdi verilmiştir. Açıklama 1- He said, "Indira went to the market." ve açıklama 2- He said, "Indira Gandhi was a great Prime Minister." Bu durumda, deyim 2 ağa eklendiğinde, "Indira" kelimesinin eklendiği adım 3'te, Y^3 hücre tahmini veya çıkış sinyali kontrol edilir ve çift yönlü olması nedeniyle, ileri bilgi 1. hücreden akar ve 2. veya 3. hücrenin yanı sıra, 9. hücreden 3. hücreye doğru geriye bilgi akışı ile 2. ifadedeki 'İndira'nın Başkan olduğunu ve kendisine giden 'İndira' olmadığını tahmin etmesine yardımcı olur [2].



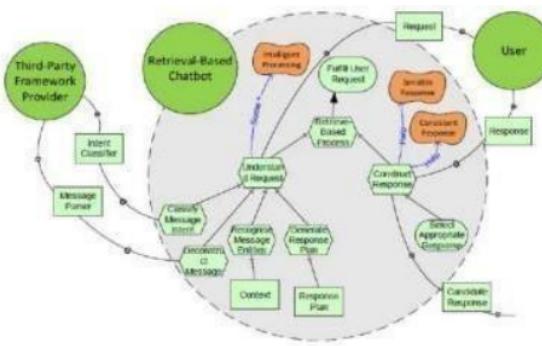
Şekil.5 İleri ve geri bilgi akışını tanımlamak için gizli katman ile BRNN'ye kaynak girişi gösterimi [2]

Bilgi iletişimindeki dönüşüm yirminci yüzyılın son yirmi yılında gerçekleştiğinde medya endüstrisinde de önemli değişiklikler meydana geldi. Gazetecilik yöntemleri multimedya ve etkileşimli özelliklerle zenginleştirilmiştir, ancak iletişim temel medyası (metin, fotoğraflar ve videolar) sabit kalır. Son 10 yılda sohbet robotlarının kullanıma sunulması, doğal dil (Natural Language) biçiminde bile gerçekleşebilen etkileşimli sohbetleri kullanarak gazeteciliği dönüştürmek için yeni imkanlar sağlamıştır. [18]

Bilgi portalları genellikle göçmenlerin ev sahibi ülkeye uyum sağlama süreçlerini desteklemek için oluşturulur. Bununla birlikte bilgi arama süreci, doküman listelerinden istenen bilgileri almanın uzun sürebileceği için göçmenler için yorucu ve karıştırıcı olabilir. Sohbet robotlarının kullanımını kolay, doğal ve sezgiseldir ve bu nedenle bilgi aramayı destekleyebilir. Sohbet botu geliştirmede ortak tasarım katılımcıları olarak göçmenleri ve diğer paydaşları dahil eden ve güçlendiren araştırma eksikliği olduğu düşünülmektedir ve göçmenlerin sosyal entegrasyonlarını destekleyen bir sohbet robottu tasarlamayı hedefleyen çalışma mevcuttur. Bu çalışmada ortak tasarım yaklaşımı kullanarak, göçmenler çevrimiçi anketler, empati araştırmaları, anketler ve ortak tasarım atölyeleri gibi etkinlikler yürütülmüştür. [19]

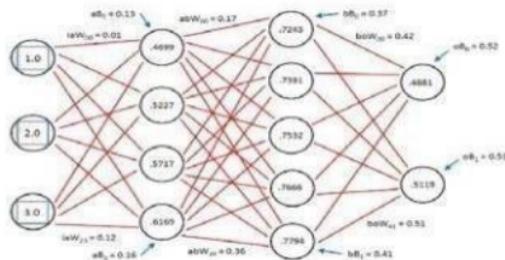
Yapılan bir çalışmada Hindistan'da ülkenin tamamına etki eden covid-19 sorunun üstesinden gelinmesine yardımcı olmayı amaçlayan sohbet robottu geliştirilmiştir. Hastalar, hastanelerdeki yatak gibi kritik kaynakları elde etmek için hastanelerin dışında uzun kuyruklarda beklemekte fakat zaman zaman yeterli yatak olmaması nedeniyle elleri boş yere geri dönebilmektedirler. Aynı çalışmada önerilen sohbet botu, kullanıcıların seyahat etmelerine gerek kalmadan tüm hastanelerdeki mevcut yatakları ve gerekli tıbbi yardımın bulunup bulunmadığını sorgulayabilmeleri hedeflenmiştir. Çalışmada bahsedilen deneysel sonuçlara göre önerilen sohbet botunun, yataklar ve oksijen tüplerinin mevcudiyeti hakkında hastane bilgileriyle ilgili tüm kullanıcıların sorularına başarıyla yanıt verdiği gösteriyor. [20]

Bu araştırmanın amacı, kendisine verilen önermeye dayalı olarak kişiye tavsiye vererek yanıt veren bir hizmet olarak bir danışman Chatbot oluşturmaktır. Xen, kullanıcılarla yaptığı konuşmalara dayanarak kullanıcının ne iletmeye çalıştığını öğrenir ve anlar. Üretken sohbet robotları, veriler üzerinde eğitim alır ve verileri anlamak için kurallar oluşturarak içerik sınıflandırması veya içerik sınıflandırması temelinde çalışır ve böylece tüm senaryolar için çalışabilir. Bağlam sınıflandırmasında hangisinin ne anlama geldiğini ve ne anlama gelebileceğini anlamak için birçok bilginin analiz edilmesi gereklidir. Bağlam, muğlaklık, eksik ve yanlış verilerle birlikte birçok anlam taşıyabilir. İnsanlar tarafından sohbet etmek veya sorunlarını anlamak ya da sadece bir düşünce akışına ihtiyaç duyuklarında konuşmak için kullanılabilir. Önerilen sohbet robotları, kullanıcıyla etkileşim kurarak yeterli veriyi topladıktan sonra, K-Means kullanarak kendi kendine öğrenmeyi kullandığı Alım, AIML (Yapay Zeka İşaretleme Dili) ve Üretken sohbet botudur. [10] Üretken sohbet robotu, yanıtlarını iyileştirebilmesi ve kullandığı yanıtları genişletebilmesi için bağlam sınıflandırmasını kullanarak kendini eğitir. Önerilen sohbet robotu Xen, verildiği bağlama göre uygun yanıtlar verir ve uygun şekilde yanıt verir ve tasvir edilen duyguya iyi anlar. Yaptığı konuşmalar üzerinde eğitim alır ve bir dahaki sefere benzer bir bağlamda daha uygun yanıtlar verir ve ayrıca daha fazla duruma yanıt verebilir. [10]



Şekil.6 i* Erişim Tabanlı Sohbet Robotları için Stratejik Mantık Modeli [11]

Derin Öğrenme, Makine öğrenimini orijinal ve ana hedeflerinden birine, yani Yapay zekâya yaklaşımak amacıyla tanıtılan bir Makine Öğrenimi ve araştırma alanıdır. Otomatik öğrenme algoritmasından bahsedersek, bunlar doğrusal olma eğilimindedir, DL (Derin Öğrenme) algoritmaları karmaşıklığı ve soyutlamayı artıracak şekilde yapılandırılmıştır. Derinlemesine öğrenmek için ilk kelimesi kedi olan 5 yaşında bir çocuk düşünüm. Çocuk, nesneleri göstererek ve kedi kelimesini söyleyerek kedinin ne olduğunu öğrenmeye devam eder. Anne, "Evet, o bir kedi" veya "Hayır, o bir kedi değil" der. Çocuk nesneleri işaret etmeye devam ettikçe, tüm kedilerin özelliklerinin ve özelliklerinin daha çok farkına varır. Çocuğun ne yaptığı bile bilmeden ne yaptığı çok karmaşık bir soyutlama düzeyine açıklık getirir ve her bir soyutlama düzeyinin, ölçekte bir önceki katmanda edindiği bilgi yardımıyla bir ölçek oluşturur.¹¹ Derin öğrenme kullanan sisteme çalışan yazılımlar da aynı süreçte ilerler. Her algoritma girdisine doğrusal olmayan bir yaklaşım uygulayacak şekilde ögrendiklerini verilen çıktıdan istatistiksel bir model oluşturmak için kullanır. Çıktı kabul edilebilir bir doğruluk noktasına ulaşana kadar tekrar devam eder. [3]



Şekil.7 Sinir Ağı [3]

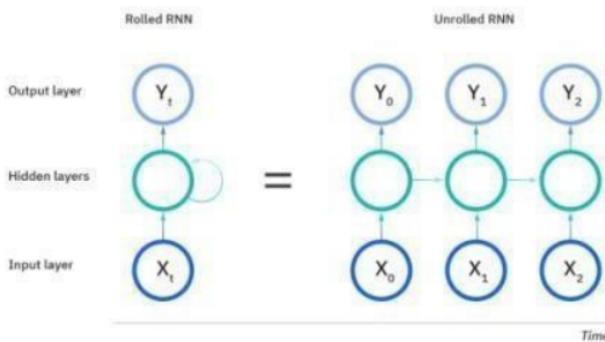
3. EVRİŞİMLİ SINİR AĞLARI

3.1 Yinelemeli Sinir Ağrı (Recurrent Neural Network) Rnn Encoder-Decoder

Yinelemeli Sinir Ağrı (RNN), sıralı verileri veya zaman serisi verilerini kullanan bir tür yapay sinir ağıdır. Bu derin öğrenme algoritmaları, dil çevirisini, doğal dil işleme (NLP), konuşma tanıma ve resim yazısı gibi sıralı veya zamansal sorunlar için yaygın olarak kullanılır; Siri, sesli arama ve Google çeviri gibi popüler uygulamalara dahil edilirler. İleri beslemeli ve Evrişimli Sinir Ağları (CNN'ler) gibi, Yinelemeli Sinir Ağları da öğrenmek için eğitim verilerini kullanır. Mevcut girdi çıktıyı etkilemek için önceki girdilerden bilgi aldıktarı için “hafızaları” ile ayırt edilirler. Geleneksel derin sinir ağları, girdilerin ve çıktıların birbirinden bağımsız olduğunu varsayıken, yinelemeli sinir ağlarının çıktısı, dizi içindeki önceki öğelere bağlıdır. Gelecekteki olaylar, belirli bir dizinin çıktısının belirlenmesinde de yardımcı olurken, tek yönlü tekrarlayan sinir ağları, bu olayları tahminlerinde açıklayamaz. Bir Yinelemeli Sinir Ağrı (RNN), bir simbol dizisini sabit uzunlukta bir vektör temsiline kodlar ve diğerini gösterimi başka bir simbol dizisine çözer. Önerilen modelin kodlayıcısı ve kod çözücü, bir kaynak dizisi verilen bir hedef dizinin koşullu olasılığını en üst düzeye çıkarmak için ortaklaşa eğitilir. İstatistiksel bir makine çevirisini sisteminin performansının, mevcut log-lineer modelde ek bir özellik olarak RNN EncoderDecoder tarafından hesaplanan tümce çiftlerinin koşullu olasılıkları kullanılarak ampirik olarak iyileştirildiği bulunmuştur. Niteliksel olarak, önerilen modelin dilsel ifadelerin anlamsal ve sözdizimsel olarak anlamlı bir temsilini öğrendiği gözlemlenir.

RNN'lerin açıklanmasında bize yardımcı olması için, ingilizcede bulunan “havanın altında hissetmek” deyiminin anlamlı olması için, o belirli sırayla ifade edilmesi gereklidir. Sonuç olarak, yinelemeli ağların deyimdeki her kelimenin konumunu hesaba

katması gereklidir ve bu bilgiyi dizideki bir sonraki kelimeyi tahmin etmek için kullanırlar. Aşağıda verilmiş olan görselde, RNN'nin "yuvarlanmış" görseli, tüm sinir ağını veya daha doğrusu "havanın altında hissetmek" gibi öngörülen tüm ifadeyi temsil ediyor. "Kaydırılmamış" görsel, sinir ağının tek tek katmanlarını veya zaman adımlarını temsil eder. Her katman, "hava durumu" gibi o ifadedeki tek bir kelimeyle eşlenir. "Hissetme" ve "altında" gibi önceki girdiler, "the" dizisindeki çıktıyı tahmin etmek için üçüncü zaman adımda gizli bir durum olarak temsil edilecektir.[25]



Şekil.8 ,RNN'nin “yuvarlanmış” görseli [25]

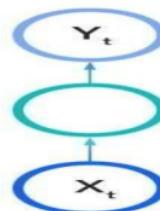
Tekrarlayan ağların bir diğer ayırt edici özelliği, ağın her katmanında parametreleri paylaşmalarıdır. İleri beslemeli ağlar her düğümde farklı ağırlıklara sahipken, tekrarlayan sinir ağları ağın her katmanında aynı ağırlık parametresini paylaşır. Bununla birlikte, bu ağırlıklar, pekiştirmeli öğrenmeyi kolaylaştırmak için geri yayılım ve gradyan iniş süreçlerinde hala ayarlanır. Tekrarlayan sinir ağları, dizi verilerine özgü olduğu için geleneksel geri yayılımdan biraz farklı olan gradyanları belirlemek için zaman içinde geri yayılım (BPTT) algoritmasından yararlanır. BPTT'nin ilkeleri, modelin çıktı katmanından girdi katmanına hataları hesaplayarak kendini eğittiği geleneksel geri yayılımla aynıdır.

Bu hesaplamalar, modelin parametrelerini uygun şekilde ayarlamamızı ve uydurmamızı sağlar. BPTT geleneksel yaklaşımından farklıdır, çünkü BPTT her zaman adımında hataları toplar, ileri beslemeli ağlar ise her katmanda parametreleri paylaşmadıklarından hataları toplamaya ihtiyaç duymazlar. Bu süreç boyunca, RNN'ler, patlayan gradyanlar ve kaybolan gradyanlar olarak bilinen iki probleme karşılaşma eğilimindedir. Bu sorunlar, hata eğrisi boyunca kayıp fonksiyonunun eğimi olan gradyanın boyutu ile tanımlanır. Gradyan çok küçük olduğunda, küçülmeye devam eder, ağırlık parametreleri önemsiz hale gelene kadar güncellenir. Bu gerçekleştiğinde, algoritma artık öğrenmiyor. Patlayan gradyanlar, gradyan çok büyük olduğunda meydana gelir ve kararsız bir model oluşturur. Bu durumda, model ağırlıkları çok büyüyecek ve sonunda NaN olarak temsil edilecektir. Bu sorumlara bir çözüm, sinir ağı içindeki gizli katmanların sayısını azaltarak RNN modelindeki karmaşıklığın bir kısmını ortadan kaldırmaktadır.

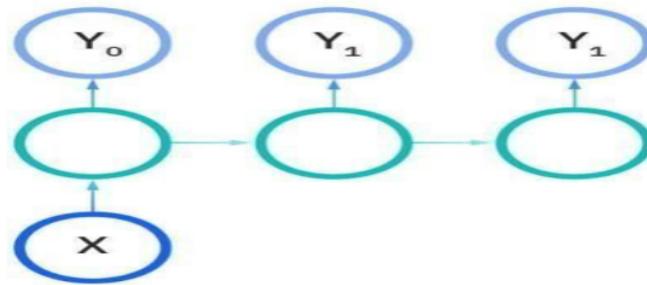
4

3.2 Tekrarlayan Sinir Ağları Türleri

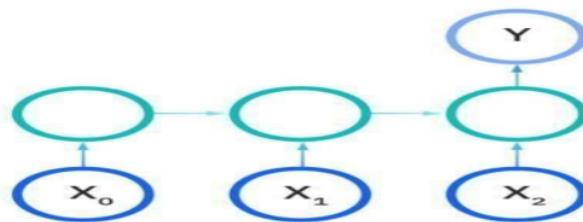
İleri beslemeli ağlar, bir girdiyi bir çıktıya eşler ve yukarıdaki diyagamlarda tekrarlayan sinir ağlarını bu şekilde görselleştirirken, aslında bu kısıtlamaya sahip değildir. Bunun yerine, giriş ve çıkışlarının uzunluğu değişimdir ve müzik oluşturma, duygusal sınıflandırması ve makine çevirisi gibi farklı kullanım durumları için farklı RNN türleri kullanılır. Farklı RNN türleri genellikle aşağıdaki diyagamlar kullanılarak ifade edilir:



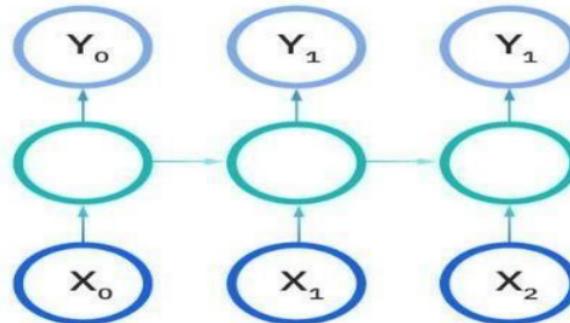
Şekil.9 Bir Girdinin Bir Çıktıya Eşlenmesi [26]



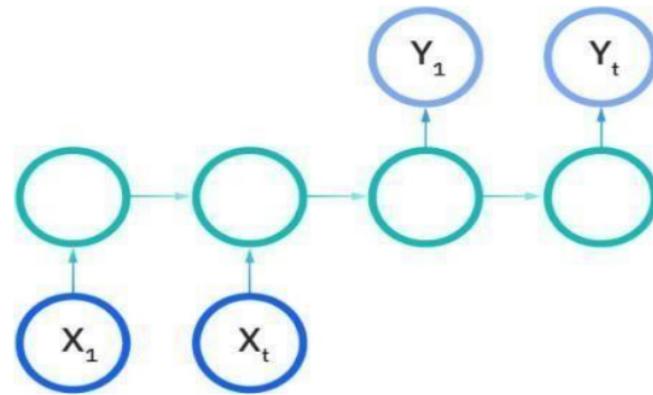
Şekil.10 Bir Girdinin Birden Çok Çıktıya Eşlenmesi [26]



Şekil.11 Birden Fazla Girdinin Bir Çıktıya Eşlenmesi [26]



Şekil.12 Birden Fazla Girdinin Birden Fazla Çıktıyla Eşlenmesi [26]

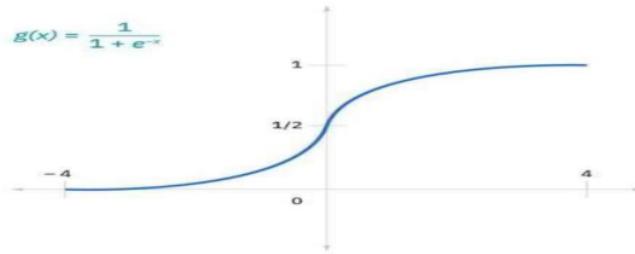


Şekil.13 Birden Fazla Girdinin Birden Fazla Çıktıyla Eşlenmesi
[26]

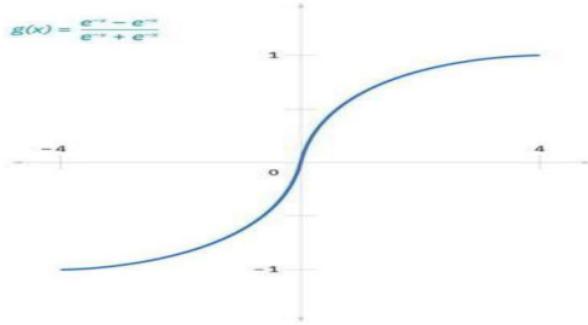
Tekrarlayan bir sinir ağı (RNN), bir katman içindeki ağırlıklı bağlantıları içeren bir sinir ağları sınıfıdır (beslemeyi yalnızca sonraki katmanlara bağlayan geleneksel ileri beslemeli ağlarla karşılaştırıldığında). RNN'ler döngüler içerdiğiinden, yeni girdiyi işlerken bilgi depolayabilirler. Bu bellek, onları önceki girdilerin dikkate alınması gereken (zaman serisi verileri gibi) işleme görevleri için ideal kılardır. Bu nedenle mevcut derin öğrenme ağları RNN'lere dayanmaktadır. Bu öğretici, RNN'lerin arkasındaki fikirleri araştırır ve seri veri tahmini için sıfırdan bir tanesini uygular. Sinir ağları, yüksek düzeyde bağlantılı işlem elemanları (nöronlar) ağına dayalı olarak bir girdiyi bir çıktıya eşleyen hesaplama yapılarıdır.

3.3 Ortak Aktivasyon Fonksiyonları

Sinir Ağları hakkındaki Öğren makalesinde tartışıldığı gibi, bir aktivasyon işlevi, bir nöronun etkinleştirilip etkinleştirilmeyeceğini belirler. Doğrusal olmayan fonksiyonlar tipik olarak belirli bir nöronun çıktısını 0 ile 1 veya -1 ile 1 arasında bir değere dönüştürür. En sık kullanılan fonksiyonlardan bazıları şu şekilde tanımlanır:



Şekil.14 Sigmoid: $g(x) = 1/(1 + e^{-x})$ formülüyle temsil edilir. [26]



Şekil.15 Tanh: $g(x) = (e^{-x} - e^x)/(e^{-x} + e^x)$ formülüyle temsil edilir. [26]

Basitçe söylemek gerekirse, ortak aktivasyon fonksiyonları ağın verilerdeki karmaşık kalıpları öğrenmesine yardımcı olmak için bir yapay sinir ağına eklenen bir işlevdir. Beynimizde bulunan nöron temelli bir modelle karşılaşıldığında, aktivasyon işlevi sonunda bir sonraki nörona neyin ateşleneceğine karar verir. Bir ortak aktivasyon fonksiyonunun ANN'de (Yapay Sinir Ağları) tam olarak budur. Bir önceki hücreden gelen çıkış sinyalini alır ve onu bir sonraki hücreye girdi olarak alınabilecek bir forma dönüştürür.

Şekil.14'de görmüş olduğumuz Sigmoid fonksiyonu Yapay Sinir Ağları'nda en sık kullanılan aktivasyon fonksiyonudur. Sigmoid fonksiyonunun bu kadar sık kullanılmasının ana nedeni (0 ile 1) arasında olmasıdır. Bu nedenle özellikle çıktı olarak olasılığı tahmin etmemiz gereken modeller için kullanılır. Herhangi bir şeyin olasılığı sadece 0 ile 1 aralığında olduğu için sigmoid doğru seçimdir.

Şekil.15'de görmüş olduğumuz Tanh fonksiyonu Yapay Sinir Ağları'nda kullanılan bir başka alternatif aktivasyon fonksiyonudur. Hiperbolik tanjant aktivasyon fonksiyonu aynı zamanda basitçe Tanh (ayrıca "tanh" ve "TanH") fonksiyonu olarak da adlandırılır. Sigmoid aktivasyon fonksiyonuna çok benzer ve hatta aynı S şekline sahiptir. Fonksiyon, herhangi bir gerçek değeri girdi olarak alır ve -1 ile 1 aralığındaki değerleri çıkarır. Girdi ne kadar büyükse (daha pozitif), çıktı değeri 1.0'a o kadar yakın olur, oysa girdi ne kadar küçükse (daha negatif), o kadar yakın olur. Çıktı -1.0 olacaktır.

3.4 Varyant Rnn Mimarileri

Çift yönlü tekrarlayan sinir ağları (BRNN): Bunlar, RNN'lerin değişken bir ağ mimarisidir. Tek yönlü RNN'ler yalnızca mevcut durum hakkında tahminlerde bulunmak için önceki girdilerden alınamasınken, çift yönlü RNN'ler doğruluğunu artırmak için gelecekteki verileri çeker. Bu makalenin başlarındaki "havanın altında hissetmek" örneğine geri dönersek, model dizideki son kelimenin "hava" olduğunu bilseydi, bu ifadedeki ikinci kelimenin "under" olduğunu daha iyi tahmin edebilir.

Uzun kısa süreli bellek (LSTM): Bu, Sepp Hochreiter ve Juergen Schmidhuber tarafından kaybolan gradyan sorununa bir çözüm olarak tanıtılan popüler bir RNN mimarisidir. Makalelerinde (PDF, 388 KB) (bağlantı IBM dışındadır), uzun vadeli bağımlılıklar sorununu çözmek için çalışırlar. Yani, mevcut tahmini etkileyen önceki durum yakın geçmişte değilse, RNN modeli mevcut durumu doğru bir şekilde tahmin edemeyebilir. Örnek olarak, aşağıdaki italik kelimeleri tahmin etmek istediğimizi varsayalım, "Alice findıklara alerjisi var. Fıstık ezmesi yiyecek." Bir findık alerjisi bağlamı, yenemeyen yiyeceklerin findık içerdigini tahmin etmemize yardımcı olabilir. Bununla birlikte, bu bağlam birkaç cümle önce olsayıdı, RNN'nin bilgiyi bağlamasını zorlaştırır, hatta imkansız hale getirirdi. Bunu düzeltmek için, LSTM'ler, sinir ağının gizli katmanlarında üç kapıya sahip "hücrelere" sahiptir: ³ bir giriş kapısı, bir çıkış kapısı ve bir unutma kapısı. Bu kapılar, ağıdaki çıktıyı tahmin etmek için gerekli olan bilgi akışını kontrol eder. Örneğin, "she" gibi cinsiyet zamirleri önceki cümlelerde birden çok kez tekrarlandıysa, bunu hücre durumundan hariç tutabilirisiniz. Kapılı yinelenen birimler (GRU'lar): Bu RNN varyantı, RNN modellerinin kısa süreli bellek sorununu da çözmeye çalıştığı için LSTM'lere benzer. Bir "hücre durumu" düzenleme bilgisi kullanmak yerine, gizli durumları kullanır ve üç kapı yerine iki tane vardır: bir sıfırlama kapısı ve bir güncelleme kapısı. LSTM'lerdeki kapılarla benzer şekilde, sıfırlama ve güncelleme kapıları, ne kadar ve hangi bilgilerin tutulacağını kontrol eder.

3.5 Tekrarlayan Sinir Ağları Ve Ibm Cloud

8

IBM, on yıldır, IBM Watson'ın gelişimi ve evrimi ile vurgulanan yapay zeka teknolojilerinin ve sinir ağlarının geliştirilmesinde öncü olmuştur. Watson, artık yapay zekanın benimsenmesi ve uygulanmasına yönelik kantilanmış katmanlı bir yaklaşım kullanarak gelişmiş doğal dil işleme ve derin öğrenme tekniklerini sistemlerine uygulamak isteyen kuruluşlar için güvenilir bir çözümdür. IBM Watson Machine Learning gibi IBM ürünler, yinelenen sinir ağlarında yaygın olarak kullanılan TensorFlow, Keras ve PyTorch gibi popüler Python kitaplıklarını da destekler. Kuruluşunuz, IBM Watson Studio ve Watson Machine Learning gibi araçları kullanarak, modellerinizi herhangi bir bulutta devreye alıp çalıştırırken açık kaynaklı yapay zeka projelerinizi sorunsuz bir şekilde üretime getirebilir.

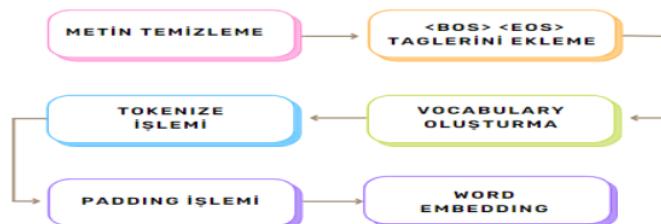
4. YÖNTEM

4.1 Kullanılan Materyaller ve Aşılan Zorluklar

Bu bölümde LSTM katmanı içeren seq2seq modeli uygulayarak sohbet bot uygulaması tasarlanmıştır. Uygulamayı geliştirme aşamasında ise yapay zeka uygulamaları için en uygun dillerden Python kullanılmıştır. Ayrıca, ihtiyaç duyulan kütüphaneler ve frameworkler kullanılarak özellikle, ücretsiz GPU desteği sağlanması sebebi ile, Google’ın hizmeti olan Colab kullanılmıştır. Uygulama geliştirilirken performans düşüklükleri, tutarsız veya anlamsız çıktı gibi sorunlarla karşılaşılmıştır. Karşılaşılan bu sorunlar sonucunda, konfigürasyon işlemleri tekrar düzenlenip gerekli adımlar atılarak yeniden eğitim sağlanmış ve başarı oranı tatmin edici seviyelere getirilmiştir.

4.2 Hazırlık Aşaması

Çalışmaya başlamadan önce sohbet botumuza soracağımız soruları ve bu sorulara karşılık beklediğimiz cevapları düşündük. Bunu düşünme sebebimiz konumuz hakkında çok fazla veri seti mevcut olması ve botumuzdan almayı beklediğimiz cevaplar hem veri setine hem soracağımız sorulara bağlıydı. Veri setimizi seçtikten sonra izlediğimiz ön işleme adımları aşağıdaki gibidir.



Şekil.16 Hazırlık Aşaması

4.3 Metin Temizleme

Bu aşama için kullanılan fonksiyonlar genel olarak birbirleriyle benzerdir. Bizim kullandığımız fonksiyonun kod parçası şekildeki gibidir.

```
def temizle_txt(txt):
    txt = txt.lower()
    txt = re.sub(r"i'm", "i am", txt)
    txt = re.sub(r"he's", "he is", txt)
    txt = re.sub(r"she's", "she is", txt)
    txt = re.sub(r"it's", "it is", txt)
    txt = re.sub(r"that's", "that is", txt)
    txt = re.sub(r"what's", "that is", txt)
    txt = re.sub(r"where's", "where is", txt)
    txt = re.sub(r"how's", "how is", txt)
    txt = re.sub(r"\'ll", " will", txt)
    txt = re.sub(r"\'ve", " have", txt)
    txt = re.sub(r"\'re", " are", txt)
    txt = re.sub(r"\'d", " would", txt)
    txt = re.sub(r"\'ne", " are", txt)
    txt = re.sub(r"\'won't", "will not", txt)
    txt = re.sub(r"\'can't", "cannot", txt)
    txt = re.sub(r"\n't", " not", txt)
    txt = re.sub(r"\n'", "ng", txt)
    txt = re.sub(r"\'bout", "about", txt)
    txt = re.sub(r"\'til", "until", txt)
    txt = re.sub(r"[-()\"#/@;:<>{}^+=~|.!?.,]", "", txt)
    return txt
```

Şekil.17 Fonksiyona giren string yapısını istediğimiz biçimde çeviren kod parçası

```
["Drink up, Charley. We're ahead of you.",
 'Did you change your hair?',
 'I believe I have found a faster way.']}
```

Şekil.18 Fonksiyon çalışmadan önceki data

```
['drink up charley we are ahead of you',
 'did you change your hair',
 'i believe i have found a faster way']
```

Şekil.19 Fonksiyon çalıştırınca sonraki data

4.4 Etiket Ekleme

Ön işleme aşamalarımızdan birisi de decoder, encoder inputlarına etiketler eklemektir. <PAD> etiketini eğitim aşamasında ağa göndereceğimiz batchlerin eşit boyutta olmasını sağlamaktadır. <EOS> etiketimiz cümlenin bitiş noktasını belirtir. <SOS> etiketimiz cümlenin başlangıcını işaretler. Şekil.20'deki döngüde temizle_soru datamızı önce satır satır sonrasında kelime kelime dolaşarak vocab datasında bulunmayan kelimeleri tespit edip yerine <OUT> etiketini ekledik.

```
151 for cümle in temizle_soru:  
152     lst = []  
153     for word in cümle.split():  
154         if word not in vocab:  
155             lst.append(vocab['<OUT>'])  
156         else:  
157             lst.append(vocab[word])  
158
```

Şekil.20 Sözlüğümüzde olmayan kullanıcı girdisine <OUT> tagi ekleyen kod parçası

```
for i in range(len(temizle_cevap)):  
    temizle_cevap[i] = '<SOS> ' + temizle_cevap[i] + ' <EOS>'
```

Şekil.21 Cevap datamızın başlangıcını ve sonunu işaretleyen kod parçası

4.5 Vocabulary Oluşturma

Bu aşamada datamızın sözlüğünü oluşturuyoruz diyebiliriz. Sözlüğümüzde kelimelere atanmış key değerleri olacaktır. İlk önemli adımımız şekildeki fonksiyon ile her kelimenin kaç kez kullanıldığı belirlemektir. Sonrasında belirlediğimiz thresh değerinden fazla kez tekrarlanan kelimeleri sözlüğümüze ekleyerek sözlüğümüzü oluşturmuş olduk. Thresh değerimiz uygulamamızın tasarlama süresince değişiklikler görmüştür.

```
word2count = {}

for line in clean_ques:
    for word in line.split():
        if word not in word2count:
            word2count[word] = 1
        else:
            word2count[word] += 1
for line in clean_ans:
    for word in line.split():
        if word not in word2count:
            word2count[word] = 1
        else:
            word2count[word] += 1
```

Şekil.22 Özel keyler barındıran sözlük datamızı oluşturmamıza yardımcı kod parçası

4.6 Piksel Ekleme

Bu aşamada decoder ve encoder inputlarının eşit uzunlukta olmasını sağlamayı amaçlıyoruz. Birçok algoritma input datasının son halinde eşit uzunluk beklemektedir. Belirlediğimiz uzunluk 13'tür.

4.7 Seq2seq Modeli Oluşturma

4.7.1 tf.keras.Model

Uygulamamızda kullandığımız katmanları groplayan, eğitimimiz için kullanışlı özellikler sağlayan bir modüldür. Sequential modülü yerine Model kullanmamızın sebebi eğitim ve tahmin aşamalarında daha fazla kontrol sahibi olmamız.

4.7.2 Katmanlar

4.7.2.1 Input

Encoderimize yollayacağımız datayı girdiler haline getirdik. İlk parametremiz bu girdilerin boyutudur. Kodumuzun önceki kısımlarında pad_sequence oluştururken de 13 değerini belirlemiştik burada da aynı değeri kullanıyoruz.

```
5 enc_inp = Input(shape=(13, ))
6 dec_inp = Input(shape=(13, ))
```

Şekil.23 Girdilerimizi Input hale getiren Input katmanımızın kod parçası

```
179 encoder_inp = pad_sequences(encoder_inp, 13, padding='post', truncating='post')
180 decoder_inp = pad_sequences(decoder_inp, 13, padding='post', truncating='post')
```

Şekil.24 Padding katmanı, encoder-decoder input oluşturma kod parçası

4.7.2.2 Kelime Gömme İşlemi

Bu katmanımız ile daha önce oluşturduğumuzdan bahsettiğimiz (bknz. 3.2.3) vocab datamızı sıkıştırıyoruz. Birinci parametremiz olan input_dim değerini vocab datamızın büyüklüğü olan VOCAB_SIZE olarak belirliyoruz. VOCAB_SIZE değerini 1 artırma sebebimiz ekstra bir etiketi eklememizdir. Output_dim parametresini 50 olarak belirledik. Bu parametre, sıkıştırma sonucu olmasını istediğimiz boyutu temsil etmektedir. Input_length parametresinin 13 olduğunu daha önce pad_sequence bölümünde belirlemiştik. Son olarak trainable parametresini de true olarak belirledik.

4.7.2.3 LSTM

Tf.keras.layer'dan import ettiğimiz LSTM modülünde encoder ve decoder için 3'er parametre kullandık. Bu parametrelerden ilki LSTM yapımızın vermesini istediğimiz boyut sayısını belirtir. Biz 300 olarak belirledik. Her adımdan hidden state ve sequence çıktısı almak için şekildeki gibi return_sequences ve return_state parametrelerini true olarak belirledik.

```
17 enc_lstm = LSTM(400 ,return_sequences=True, return_state=True)
```

```
24 dec_lstm = LSTM(400,  return_sequences=True, return_state=True)
```

Şekil.25 dec_lstm ve enc_lstm oluşturma kod parçası, hücre sayısı 400

4.7.2.4 Dense

Dense katmanımız, katmanlar arası geçişler sağlamaktadır. Şekil.26'de görüldüğü üzere iki adet parametre ile kullanım sağladık. Activation = "softmax" kullanımının sebebi modelimizin olasılık odaklı olmasıdır. Yine Şekil.26'de görüldüğü gibi decoder outputunu dense layerda parametre olarak kullandık.

```
28 dense = Dense(VOCAB_SIZE, activation='softmax')
29
30 dense_op = dense(dec_op)
31
```

Şekil.26 Dense katmanı için "dense" tanımlandı, "dec_op" parametre olarak verilerek katmanımız oluşturuldu

4.8 Model

4.8.1 Decoder Model

LSTM için 300 hücre kullandık. Bu yüzden 300 adet h ve c çıktımızın oluşması beklenmektedir. Şekil.27 kod satırı 10 ve 11'de görüldüğü üzere ⁹ `decoder_state_input_h` ve `decoder_state_input_c` için `Input` katmanı kullandık. Decoderimizin inputu için gerekli state'ler oluştuktan sonra `decoder_states_inputs` değişkenimiz ile decoder inputumuzun son halini oluşturmuş olduk. Oluşturduğumuz inputları kullanarak elde ettiğimiz outputlar Şekil.27 satır 13'de görüldüğü üzere `state_h`, `state_c` ve `decoder_outputs` değişkenlerine atanmıştır. Daha sonra decoder outputundan elde ettiğimiz stateler ile Şekil.27 satır 13'deki görüldüğü gibi state listesine ekledik. Son olarak satır 15'de görüldüğü gibi önceki satırlarda oluşturduğumuz değişkenleri kullanarak decoder modelimizi oluşturduk.

```
decoder_state_input_h = Input(shape=(400,))
decoder_state_input_c = Input(shape=(400,))
decoder_states_inputs = [decoder_state_input_h, decoder_state_input_c]
decoder_outputs, state_h, state_c = dec_lstm(dec_embed, initial_state=decoder_states_inputs)
decoder_states = [state_h, state_c]
dec_model = Model([dec_inp, decoder_states_inputs],
                  [decoder_outputs] + decoder_states)
```

Şekil.27 Decoder inputunun oluşturulduğu, bu inputlar ile output statelerinin oluşturulmasıyla decode modelin tamamlandığı kod parçası

4.8.2 Encoder Model

Kullandığımız LSTM decoderde olduğu gibi 300 hücrelidir. Önce Şekil.28'da görüldüğü gibi Input katmanımız ile input verimizi oluşturduk. Bu input verimizi kullanarak LSTM'den çıktı elde edebilmek için embedding katmanı ile verimizi işlenebilir hale getirdik. Hazır LSTM fonksiyonu ile oluşturduğumuz LSTM'e embedding katmanında hazırladığımız input verisini yollayarak encoder output, h state ve c state elde ettik. Bu elde ettiğimiz stateler bize decoderda lazım olacaktır. Bu yüzden state list oluşturup bu stateleri listemize koyduk.

```
enc_embed = embed(enc_inp)
enc_lstm = LSTM(400, return_sequences=True, return_state=True, dropout=0.05)
enc_op, h, c = enc_lstm(enc_embed)
enc_states = [h, c]
```

Şekil.28 Input katmanında oluşturduğumuz encoder input datamızı embed edip, oluşturduğumuz LSTM modele yolluyoruz. Gerekli encoder output stateleri elde ediyoruz

4.9 Model Test Aşaması

Bu aşamada botumuza bir cümle yazıyoruz. Yazdığımız bu cümle ilk olarak clean_text fonksiyonuyla ön işlemeye giriş yapıyor. Bu aşamadan sonra girdimiz dizi haline getiriliyor. Bir sonraki aşama için key değerlerini tutmaya yarayacak txt[] dizisi oluşturuluyor. Dizi haline getirdiğimiz cümle girdimiz döngü içinde kelime'lere ayrılarak vocab datamızda var olup olmadığı kontrol ediliyor. Var ise hangi key değerine sahip olduğu ayrı bir lst[] listesinde tutuluyor.

Bu list döngünün içindeki geçici bir listtir. Döngünün farklı bir katmanında lst’den txt’ye veri aktarımı söz konusudur. Döngünün ardından elde ettiğimiz txt listini pad_sequence’te kullanıyoruz. Elde ettiğimiz yeni verimizi txt değişkenine atayıp “enc_model” modelimizde parametre olarak kullanıyoruz. Daha sonra 1 adet satır ve sütundan, 0 değerlerinden oluşan numpy dizisi tanımlıyoruz. Bu dizinin 0. indislerine <SOS> etiketini koyuyoruz. Daha sonra bir döngü içerisinde decoder modelimize encoder predictlerimizi ve boş numpy dizimizi parametre olarak atayarak çalıştırıyoruz. Oluşan decoder outputumuzu modelimizde tanımlı dense katmanına yolluyoruz. Dense katmanındaki softmax özelliği sayesinde sonuç olarak olasılıklar elde ediyoruz. Kullandığımız arg_max hazır fonksiyonu ile en yüksek olasılıklı datanın indeksini elde ediyoruz. Elde ettiğimiz indis ile sözlükte tahmin ettiğimiz kelimeyi sample_word değişkenine atıyoruz. Daha sonra if else yapısı ile <EOS> etiketi kontrolü yapıyoruz. Etiket varsa veya tahmin ettiğimiz sonuç cümlesinin kelime sayısı 13’ten büyük olursa stop_condition ile döngümüzden çıkıyoruz. Etiket yoksa döngümüz devam ediyor.

5. SONUÇ

1 Sohbet botu çalışmasında, LSTM ve sıradan seq2seq modeli ile sohbet bot 1 uygulaması geliştirilmiştir. Çalışmada, veri kümesi olarak Movie Dialog Corpus veri kümesi kullanılmıştır. Çalışma ile, Movie Dialog Corpus veri kümesinden alınan verilerle kullanıcının sorduğu sorulara uygun cevaplar veren bir sohbet botu tasarımları yapılmıştır. Çalışma, categorical_crossentropy kayıp fonksiyonu ile 40 adımlik dönem için eğitilmiştir. Kayıp oranı son adımdan sonra 0.5621 değerlerine düşürülmüştür. Elde edilen doğruluk oranı ise yüzde 85 dir. Elde edilen verilere göre eğitim sayısı arttıkça doğruluk oranı da artmaktadır. TPU destekli bilgisayarlarda daha kapsamlı veri kümesi kullanıldığında kapsamı geniş cevaplar üretilebileceği ve yüksek doğrulukta sonuçlar elde edilebileceği düşünülmektedir.

SIRADAN SIRAYA LSTM MODELİ İLE CHATBOT TASARIMI

ORJİNALİK RAPORU

% **21**
BENZERLİK ENDEKSİ

% **18**
İNTERNET KAYNAKLARI

% **1**
YAYINLAR

% **6**
ÖĞRENCİ ÖDEVLERİ

BİRİNCİL KAYNAKLAR

- | | | |
|---|--|-------------|
| 1 | web.archive.org
Internet Kaynağı | % 6 |
| 2 | www.cbot.ai
Internet Kaynağı | % 5 |
| 3 | tr.wikipedia.org
Internet Kaynağı | % 3 |
| 4 | Submitted to Marmara University
Öğrenci Ödevi | % 2 |
| 5 | static.dergipark.org.tr
Internet Kaynağı | % 1 |
| 6 | Submitted to Tobb University of Economics & Technology
Öğrenci Ödevi | % 1 |
| 7 | Submitted to Bozok Üniversitesi
Öğrenci Ödevi | % 1 |
| 8 | www.ibm.com
Internet Kaynağı | <% 1 |
| 9 | ichi.pro
Internet Kaynağı | <% 1 |

10	www.coursehero.com Internet Kaynağı	<% 1
11	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Öğrenci Ödevi	<% 1
12	acikbilim.yok.gov.tr Internet Kaynağı	<% 1
13	Submitted to Robert College Öğrenci Ödevi	<% 1
14	Submitted to Kocaeli Üniversitesi Öğrenci Ödevi	<% 1
15	enpopulersorular.com Internet Kaynağı	<% 1
16	Manyu Dhyani, Rajiv Kumar. "An intelligent Chatbot using deep learning with Bidirectional RNN and attention model", Materials Today: Proceedings, 2021 Yayın	<% 1
17	Submitted to Uludag University Öğrenci Ödevi	<% 1
18	ufek2022.boun.edu.tr Internet Kaynağı	<% 1
19	jibm.ut.ac.ir Internet Kaynağı	<% 1
	academic.oup.com	

20

İnternet Kaynağı

<% 1

21

avesis.cumhuriyet.edu.tr

<% 1

Internet Kaynağı

22

docplayer.biz.tr

<% 1

Internet Kaynağı

23

gurmezin.com

<% 1

Internet Kaynağı

Alıntıları çıkart

Kapat

Eşleşmeleri çıkar

Kapat

Bibliyografyayı Çıkart

Kapat