DOKUZ EYLÜL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPISAL OLMAYAN METİNLER İÇİN ADLANDIRILMIŞ VARLIK TANIMA ALGORİTMALARI VE UYGULAMALARI

MUSTAFA GENCER

Ağustos, 2022
İZMİR

YAPISAL OLMAYAN METİNLER İÇİN ADLANDIRILMIŞ VARLIK TANIMA ALGORİTMALARI VE UYGULAMALARI

Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü
Yüksek Lisans Tezi
Bilgisayar Bilimleri Anabilim Dalı

MUSTAFA GENCER

Ağustos, 2022
İZMİR

YÜKSEK LİSANS TEZİ SINAV SONUÇ FORMU

MUSTAFA GENCER, tarafından **DR. ÖĞR. ÜY. RESMİYE NASİBOĞLU** yönetiminde hazırlanan **"YAPISAL OLMAYAN METİNLER İÇİN ADLANDIRILMIŞ VARLIK TANIMA ALGORİTMALARI VE UYGULAMALARI"** başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

Dr. Öğr. Üyesi Resmiye NASİBOĞLU

Yönetici

Prof. Dr. Ali MERT

Jüri Üyesi

Dr. Öğr. Üyesi Erdem ALKIM

Jüri Üyesi

Prof.Dr. Okan FISTIKOĞLU
Müdür
Fen Bilimleri Enstitüsü

TEŞEKKÜR

Tez çalışmam boyunca kıymetli bilgi ve tecrübelerini benden esirgemeyen, yüzleştiğim sorunlar da her zaman bana destekçi olan tez danışmanım Dr. Ögr. Üyesi Resmiye Nasiboğlu'na,

Yüksek lisans kabulümden bugüne kadar sabırla bana yardımcı olan ve destek veren çok değerli akademik bilgi ve birikimini yoluma ışık olarak tutan kıymetli hocam Prof. Dr. Efendi Nasiboğlu'na,

Tez fikrimi bulmamda bana yardımcı olan ve projemi yaparken karşılaştığım sorunlarda bana hep destek olan değerli arkadaşım Mikail Okyay'a,

Yüksek lisansa başlarken beni cesaretlendiren ve tüm hayatım boyunca kıymetli desteklerini esirgemeyen annem Fatma Gencer'e, babam İbrahim Gencer'e, kardeşlerim Simge Gencer ve Sefa Gencer'e,

Sonsuz minnet ve teşekkürlerimi sunarım.

Mustafa GENCER

YAPISAL OLMAYAN METİNLER İÇİN ADLANDIRILMIŞ VARLIK TANIMA ALGORİTMALARI VE UYGULAMALARI

Ö Z

Adlandırılmış varlık tanıma (AVT) problemi, veri çıkarımı, doğal dil işleme ve metin madenciliği gibi alanların alt dalı olarak ele alınmaktadır. Adlandırılmış varlık tanıma, yapılandırılmamış metinlerdeki varlık isimlerinin uygunluklarına göre önceden belirlenen kişi ismi, organizasyon ismi veya yer ismi gibi sınıflara atama yapmak için kullanılan bir araçtır. AVT çalışmaları pek çok alanda kullanıma sahiptir. Bunlara örnek olarak sohbet botlarının oluşturulması, sosyal ağlarda içerik önerisi oluşturma, özgeçmişleri işlemek veya müşteri çağrılarını sınıflandırmak ve onlardan öngörü elde etmek vb. söylenebilir.

Bu tez çalışmasında ilk olarak iki farklı durum üzerinde AVT yapılmıştır. İlk olarak İngilizce haber yazılarından oluşan bir veri seti üzerinde iki farklı ön eğitimli kütüphane olan Spacy ve Stanford NLP kütüphaneleri kullanılarak kişi adı, yer adı, organizasyon adı vb. varlık adları tanınmaya çalışılmıştır. Bu çalışmanın sonunda kütüphaneler ile elde edilen doğruluk oranları, kütüphanelerin çalışma yapısı, hızları vb. ölçütler karşılaştırılmıştır. Çalışmanın devamında ise Twitter'daki Türkçe tweetler kullanılarak küfür, hakaret ve uygunsuz kelimeler adlandırılmış varlık tanım problemi olarak ele alınmış ve bu kelimeler farklı yöntemler ile tespit edilmeye çalışılmıştır. Önce metinlerde geçen kelime ve kelime öbekleri etiketlenmiş daha sonra ise etiketlenen kelimeler vektörleştirilmiştir. Vektörler, RNN, çift yönlü RNN, GRU, çift yönlü GRU, LSTM, çift yönlü LSTM ve önceden eğitilmiş çok dilli BERT modeli kullanılarak eğitim yapılmıştır. Modellerin çalışma sonuçları analiz edilmiş ve sonuçları kıyaslamalı olarak değerlendirilmiştir.

Anahtar kelimeler: Metin madenciliği, adlandırılmış varlık tanıma, doğal dil işleme, küfür tespiti.

NAMED ENTITY RECOGNITION ALGORITHMS AND APPLICATIONS FOR NON-STRUCTURAL TEXTS

ABSTRACT

Named entity recognition (NER) problem is considered as a sub-branch of fields such as data extraction, natural language processing and text mining. NER is a tool used to assign classes such as predetermined person name, organization name or place name according to the suitability of entity names in unstructured texts. NER studies have use in many fields. Some of the examples are the creation of chatbots, suggesting content on social networks, processing resumes or categorizing customer calls and gaining insights from them etc.

In this study, NER was performed on two different conditions. Firstly, Spacy and Stanford NLP libraries used on a dataset consisting of news articles in English to recognize the entity names like the name of the person etc. At the end of this study, the accuracy rates obtained for each libraries, the working structure of the libraries, their speed criterias were compared. In the rest of the study, using Turkish tweets on Twitter, swearing, insults and inappropriate words were handled as a named entity definition problem and these words were tried to be determined by different methods. First, the words and phrases in the texts were labeled, and then the labeled words were vectorized. Vectors are trained using RNN, bidirectional RNN, GRU, bidirectional GRU, LSTM, bidirectional LSTM and a pre-trained multilingual BERT model. The study results of the models were analyzed and the results of the models were evaluated comparatively.

Keywords: Text mining, named entity recognition, natural language processing, profanity detection.

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

TABLOLAR LİSTESİ

Sayfa

KISALTMALAR

AVT	: Adlandırılmış Varlık Tanıma
NLP	: Natural Language Processing
LSTM	: Long Short Term Memory
CNN	: Convolutional Neural Network
BiLSTM	: Bidirectional Long Short Term Memory
CRF	: Conditional Random Fields
BERT	: Bidirectional Encoder Representations from Transformers
RNN	: Recurrent Neural Network
GRU	: Gated Recurrent Units

BÖLÜM BİR

GİRİŞ

Bir metni değerlendirirken veya anlarken, insanlar, değerler, konumlar vb. gibi adlandırılmış varlıkları doğal olarak tanırız. Örneğin, "Jack Dorsey, Amerika Birleşik Devletleri'nden bir şirket olan Twitter'ın kurucularından biridir." cümlesinde üç tür varlık tanımlanabilir:

Kişi Adı: Jack Dorsey,

Şirket Adı: Twitter,

Konum Adı: Amerika Birleşik Devletleri.

Ancak bilgisayarlar bakımından varlıkları kategorize edebilmeleri için önce onları tanımalarına yardımcı olmamız gerekir. Bu, makine öğrenimi ve doğal dil işleme aracılığıyla yapılır. Doğal dil işleme dilin yapısını ve kurallarını inceler. Metin ve konuşmadan anlam çıkarabilen akıllı sistemler yaratır. Bunun yanında makine öğrenimi ise makinelerin öğrenmesine ve zaman içinde gelişmesine yardımcı olur. Bir varlığın ne olduğunu öğrenmek için, bir adlandırılmış varlık tanıma modelinin bir kelimeyi veya bir varlığı oluşturan kelime dizisini (örneğin, "İzmir şehri") tespit edebilmesi ve hangi varlık kategorisine ait olduğunu bilmesi gerekir. AVT, bir metindeki kişi adları, yerler, markalar, parasal değerler ve daha fazlası gibi temel öğeleri kolayca tanımlamanıza yardımcı olur. Çalışmalarda kullanılan bazı adlandırılmış varlık türleri Tablo 1.1'de verilmiştir.

Bir metindeki ana varlıkları ayıklamak, yapılandırılmamış verileri sıralamaya ve büyük veri kümeleriyle uğraşmanız gerektiğinde çok önemli olan bilgileri algılamaya yardımcı olur (Sienčnik, 2015). Adlandırılmış varlık tanımanın bazı ilginç kullanım örnekleri şunlardır:

Müşteri çağrı hatlarında, müşteri isteklerini daha hızlı değerlendirmek ve yanıtlamak için AVT teknikleri kullanılabilmektedir (Subramaniam, Faruquie, Ikbal, Godbole ve Mohania, 2009). Müşterilerin sorunlarını ve sorgularını kategorilere

ayırmak gibi tekrarlayan müşteri hizmetleri görevleri otomatikleştirilebilir ve sorun çözüm oranlarını iyileştirmeye ve müşteri memnuniyetini artırmaya yardımcı olacak değerli zamandan tasarruf edilebilir. Ürün adları veya seri numaraları gibi ilgili veri parçalarını çekmek için varlık ayıklama da kullanabilir ve bu sorunu ele almak için sorunlu durumları en uygun temsilciye veya ekibe yönlendirmeyi kolaylaştırılabilir (Luo, Xiao ve Chang, 2011).

Tablo 1.1 AVT için varlık örnekleri ve tanımları[1]

TÜR	**TANIMI**
Kişi	İnsan, hayali karakter adları
Gruplar	Ulus, din, politik grup adları
Organizasyon	Şirket, ajans, enstitü adları
Yer	Ülke, şehir, eyalet adları
Konum	Dağ, su kaynağı, navigasyon dışı yer adları
Ürün	Otomobil, araç, yiyecek adları
Olay	Adlandırılmış kasırga, savaş, spor olaylarının adları
Sanatsal aktivite	Kitap, şarkı vb. Adları
Kanuni belge	Kanunla adlandırılmış belge adları
Dil	Adlandırılmış herhangi bir dil adı
Tarih	Mutlak veya göreli tarihler veya dönem adları
Zaman	Günden daha kısa zaman adları
Yüzde	% işareti içeren yüzdeler
Para	Birim dahil parasal değerler
Nitelik	Ağırlık veya mesafe olarak ölçümler
Sıralama ifadeleri	Birinci, ikinci vb.

Çevrimiçi değerlendirmeler ve incelemeler, müşteri geri bildirimi için kaynak oluşturabilir (Meng vd., 2012). Müşterilerin ürünler hakkında neleri beğendiği, beğenmediği ve işletmenin iyileştirilmesi gereken yönleri hakkında bilgiler sağlayabilir. AVT sistemleri, tüm bu müşteri geri bildirimlerini düzenlemek ve tekrar eden sorunları saptamak için kullanılabilir. Örneğin, olumsuz müşteri geri bildirimlerinde en sık bahsedilen konumları ve illeri saptamak için AVT kullanılabilir, bu sayede müşteri belirli bir ofis veya şubeye yönlendirebilir (Han vd. 2017).

[1] https://devopedia.org/named-entity-recognition

Netflix ve YouTube gibi birçok modern uygulama, optimum müşteri deneyimleri oluşturmak için öneri sistemlerine güvenir (Guo, Xu, Cheng ve Li, 2009). Bu sistemlerin çoğu, kullanıcı arama geçmişine dayalı önerilerde bulunabilen adlandırılmış varlık tanımaya dayanır. Örneğin, Netflix'te çok sayıda komedi izliyorsanız, Komedi varlığı olarak sınıflandırılmış daha fazla öneri alırsınız ya da Youtube'da izlenilen video türüne göre benzer video türleri önerilmektedir. Yine burada video türleri AVT olarak ele alınmaktadır (Bowden vd. 2018).

İşverenler, günlerinin pek çok saatini işe başvuranların özgeçmişlerini gözden geçirerek, doğru adayı arayarak geçirirler (Deepak, Teja ve Santhanavijayan, 2020; Pawar Srivastava ve Palshikar, 2012). Her özgeçmiş aynı türde bilgi içerir, ancak bunlar genellikle farklı şekilde düzenlenir ve biçimlendirilir. Varlık adı tanıma yöntemleri kullanılarak, kişisel bilgiler, ad, adres, telefon numarası, doğum tarihi, e-posta, şirket adları, beceriler, sertifikalar, eğitim ve deneyimleriyle ilgili verilere ulaşılabilir (Deepak vd. 2020; Pawar vd. 2012). Bu sayede adaylarla ilgili en alakalı bilgiler anında çıkarılır ve bu adaylar iş için hızlı bir elemeye tabi tutulur.

AVT problemi ile ilgili ilk çalışmalardan biri Rau (1991) tarafından yapılmıştır. Yazar, adlandırılmış varlık tanımayı metin içerisindeki şirket isimlerini bulmak için kullanmıştır. Daha sonra adlandırma yapmak için farklı sınıf isimleri kullanılsa da son ve güncel çalışmalarda CoNLL 2003 (Conference on Computational Natural Language Learning) ve MUC-6 (Message Understanding Conference) konferanslarında kullanılan veya ortaya atılan varlık adı tanımları kabul görmüştür ve kullanılmaktadır. CoNLL'de AVT problemini genel olarak metinde geçen ve ENAMEX olarak adlandırılan kişi adı (Person), yer adı (Location) ve organizasyon adı (Organization) için sınıflandırma işlemi olarak kabul edilmektedir (Sang ve Meulder, 2003). MUC-6' da ise ENAMEX sınıfı dışında NUMEX (parasal değerler, sayısal değerler ve yüzde ifadeleri) ve TIMEX (saat, tarih) değerlerini AVT problemine yeni sınıflar olarak dahil edilmiştir (Krupka, 1995). Bu üç AVT sınıfı, yani ENAMEX, NUMEX ve TIMEX sınıfları dışında çalışmanın kapsamına bağlı olarak veri çıkarımı için alana özgü varlık tanımlamaları da yapılabilmektedir.

```
Mr.  <ENAMEX TYPE="PERSON">Dooner</ENAMEX> met with <ENAMEX TYPE="PERSON">Martin
Puris</ENAMEX>, president and chief executive officer of <ENAMEX
TYPE="ORGANIZATION">Ammirati & Puris</ENAMEX>, about <ENAMEX
TYPE="ORGANIZATION">McCann</ENAMEX>'s acquiring the agency with billings of <NUMEX
TYPE="MONEY">$400 million</NUMEX>, but nothing has materialized.
```

Şekil 1.1 Örnek AVT gösterimi (Grishman, 1995)

Şekil 1.1'de MUC-6 Konferansından alınan yazıdan bazı AVT örnekleri verilmiştir. Burada "Mr." önekinden sonra gelen "Dooner" ismi ENAMEX olarak adlandırılmış ve kelime tipi kişi (Person) olarak belirlenmiştir. "Ammirati & Puris" ise yine ENAMEX olarak atanmış ama kelime tipi olarak organizasyon (Organization) seçilmiştir. Son olarak "$400 million" NUMEX sınıfına atanmış ve kelime tipi olarak para (Money) olarak belirlenmiştir. AVT da bu gibi yapılandırılmamış metinlerdeki varlıkları bulup daha önceden belirlenmiş sınıflara atama işlemi gerçekleştirilmektedir.

Web ve mobil uygulamaların kullanımı arttıkça, kullanıcı katkılarından kaynaklanan uygunsuz içeriğin varlığı daha sorunlu hale gelir. Sosyal haber siteleri, forumlar ve herhangi bir çevrimiçi topluluk, bir topluluğun sosyal normlarına ve beklentilerine uygun olmayanları sansürleyerek, kullanıcı tarafından oluşturulan içeriği yönetmelidir. Bu tür içeriğin sansürlenmemesi, yalnızca potansiyel kullanıcıları veya ziyaretçileri caydırmakla kalmaz, aynı zamanda bu tür içeriğin kabul edilebilir olduğunu da gösterebilir.

Gelişen teknoloji ile birlikte sosyal ağlar çok insan tarafından kullanılmaktadır. Sosyal medya kullanan kişiler her türlü resim, metin veya video içeriklerini paylaşabilmektedir. Paylaşılan bu içerikler ise uygunsuz yani aile yapısını etkiler nitelikte olabilmektedir. Sosyal medyadaki metinlerde AVT ise oldukça yaygındır. Küfür, hakaret veya aşağılayıcı sözler olarak da adlandırılabilecek sosyal açıdan saldırgan bir dildir. Her şeyin dijital olarak yönetildiği günümüzde, insanların kullandığı birçok çevrimiçi platform ve forum bulunmaktadır. Instagram gibi herhangi bir sosyal medya platformundan bir örnek alırsak, onların gizlilik politikası, kullanıcıların herhangi bir müstehcen/kaba dili herkese açık bir platformda

paylaşamayacaklarını veya yazamayacaklarını göstermektedir (Su, Huang, Chang ve Lin, 2017).

Birçok kurum ve kuruluş, kamusal alanlardaki yasa dışı faaliyetlerin tespit edilmesi için çeşitli görüntü işleme, sosyal medya metinlerindeki küfürleri tespit etmek için çeşitli metin işleme teknolojileri kullanmaktadır. Bununla birlikte, mevcut küfür algılama sistemleri, çeşitli faktörler nedeniyle hala kusurlu olmaya devam etmektedir (Su vd., 2017; Sood, Antin ve Churchill, 2012a; Laboreiro ve Oliveira, 2014). Küfür tespitinin genellikle kolay bir iş olduğu düşünülür. Bununla birlikte, geçmiş çalışmalar, mevcut liste tabanlı sistemlerin kötü performans gösterdiğini göstermiştir (Sood vd. 2012a, Sood vd. 2012b; H. S. Lee, H. R. Lee, Park ve Han, 2018). Değişen küfürlü argoya uyum sağlamada, gizlenmiş veya yalnızca kısmen sansürlenmiş (örneğin, "@ss, f$#%") veya kasıtlı veya kasıtsız olarak yanlış yazılmış (örneğin, "aptaalll" gibi) küfürlü terimleri tanımlamada başarısız olurlar. Bu nedenlerden dolayı, sistemi atlatmaları kolaydır ve hatırlanmaları çok zayıftır. İkinci olarak, liste tabanlı yaklaşım, her türlü çözüme uydurulan tek boyutlu bir çözümdür (Laboreiro vd., 2014). Saygısız veya uygunsuz tanımının, kullanımının ve algılarının tüm bağlamlarda geçerli olduğuna dair varsayımlarda bulunurlar.

Bu tez çalışmasında iki farklı veri setinde üzerinde iki farklı AVT çalışması gerçekleştirilmiştir. İlk çalışmada haber yazılarından oluşan İngilizce veride yer adı, kişi adı vb. gibi varlık isimleri bulunmaya çalışılmıştır. AVT çalışmasında Spacy ve Standord NLP adlı iki farklı ön eğitimli kütüphane kullanılmıştır. Çalışmanın sonunda da doğruluk oranları, çalışma yapıları ve süreleri karşılaştırılmıştır. İkinci çalışma da ise Twitter'dan alınan tweetler kullanılarak cümle içerisinde geçen küfür ve hakaret içeren kelimeler ve kelime grupları varlık adı olarak tanımlanmıştır. Etiketleme için IOB2 etiketleme yöntemi kullanılmıştır. Daha sonrasında etiketlenen kelime grupları BİLSTM ve BERT modelleri ile eğitilerek küfür ve hakaret olarak tanımlanan kelimeler tahmin edilmiştir.

BÖLÜM İKİ

LİTERATÜR İNCELEMESİ

2.1 Adlandırılmış Varlık Tanıma Çalışması Üzerine Literatür İncelemesi

Güneş ve Tantuğ (2018) tarafından yapılan çalışmada Milliyet gazetesinin web sitesinden alınan ve daha önceden etiketlenmiş olan Türkçe veriler kullanılmıştır. Kullanılan veri setini yapay sinir ağına eklemek için sözcükler vektörlerden oluşan bir dizi haline getirilmiştir. Sözcük vektörlerini oluşturmak için gözetimsiz öğrenme kullanılmıştır. Sözcük vektörlerini oluştururken Mikolov, Chen, Corrado ve Dean (2013) tarafından oluşturulan skip-gram modeli ile sürekli vektör uzayı temsili kullanılmıştır. Sözcük vektörlerini öğretebilmek amacıyla 184 milyon sözcükten oluşan haber yazıları derlemesi kullanılmıştır. Ayrıca ikinci olarak bu çalışmada biçim bilimsel analize de başvurulmuştur. Biçim bilim, sözcüklerin eklerine, köklerine ve gövdelerine bakarak sözcükler arasındaki ilişkileri inceleyen bilim dalı olarak tanımlanabilir.

Genel olarak bakıldığında yapay zekâ eğitimi için kelimelerin 3 ana temsil özelliğinden faydalanılmıştır. Bunlar sözcük vektörleri, yazım özellikleri ve biçim bilimsel özelliklerdir. Kelimeleri vektörleştirmek amacıyla açık kaynak kod olan Word2Vec kodları kullanılmıştır. Kelimelerin yazımsal özelliklerini ifade etmek için "hepsi büyük", "hepsi küçük", "ilk harfi büyük", "ayraç var-yok", "nokta var-yok", "sayısal değer-değil" gibi kelime özelliklerine bakılmıştır. Biçim bilimsel özelliklerde ise türemiş sözcüklerde son türemedeki sözcük türü etiketi (POS) ve biçim bilimsel etiketler ayrı birer özellik olarak ele alınmıştır. Güneş vd., (2018) çalışmasında adlandırılmış sınıf tanıma etiketi olarak ENAMEX ve kelime türü olarak organizasyon (ORG), kişi (PER), yer (LOC) adları kullanılmıştır. Bunların dışında kalan kelimeler, diğer (O) olarak sınıflandırılmıştır. Çalışmada, AVT işlemi için LSTM modelinin bir türü olan BLSTM (Deep Bidirectional Long Short-Term Memory – DBLSTM) kullanılmıştır (Hochreiter ve Schmidhuber, 1997). LSTM, yinelenen sinir ağının (RNN) bir alt dalıdır. LSTM genellikle sıralı veya zamana bağlı olarak değişen dinamik yapıları tahmin etmek için kullanılır.

Klasik sinir ağlarında ki tüm veri girişleri ve bu girdilere bağlı oluşan sonuçlar diğer giriş ve çıkışlardan ayrı veya kopuk olarak oluşmaktadırlar. Bununla birlikte genellikle, doğal dil işleme konusu gibi sıralı bilgi içeren yapılarda klasik sinir ağları pek güzel sonuçlar ortaya koyamamaktadır. Böyle bir durumda RNN devreye girer ve eldeki dizinin her bir elemanı için aynı işi, önceki sonuçları dikkate alarak gerçekleştirir. Bu sayede dizili durumdaki girdilerin bütün sıralama yapısı veya başka bir deyişle şeması muhafaza edilmiş olur. RNN tüm çıktılar kendinden önceki elemanların çıktılarına bağlı olarak değişmekte veya meydana gelmektedir. Fakat kelimeler arası mesafe arttıkça RNN'in önceki verileri kullanması zorlaşır.

Güneş vd. (2018) makalesinde kullanılan LSTM yapısı Derin çift yönlü (deep bidirectional) olarak hazırlanmış yani hem ileri (forward) hem de geri (backward) beslemeli, 50 katmanlı olarak uygulanmıştır. Daha sonra ileri ve geri besleme ile oluşturulan LSTM yapıları birleştirilmiş ve sonuna 100 ve 4 adet iki katmandan oluşan klasik sinir ağı yapısı eklenmiştir.

Öğrenme sırasında aşırı öğrenmeden kaçınmak için farklı katmanlarda DBLSTM ve unutma katsayısı kullanılmıştır. En iyi sonuç, 5 katmanlı ve 0.4 unutma katsayılı DBLSTM yapısında elde edilmiştir. Ayrıca, DBLSTM'de ki katman sayısı arttıkça başarının arttığı gözlemlenmiştir. Sonuçları değerlendirirken F1-skor parametresi kullanılmış. F1-skor parametresinin hesaplamaları için 2.1, 2.2 ve 2.3 denklemleri ve Tablo 2.1 kullanılmıştır. Özellikle tek katmanlı yapıdan iki katmanlı yapıya geçildiğinde başarıda %12,31 puanlık bir artış gözlemlenmiştir. Kütüphane olarak da Tensorflow kütüphanesinin içindeki Keras modülü kullanılmıştır.

Tablo 2.1 Hata Matrisi

	Actually Positive (1)	**Actually Negative (0)**
Predicted Positive (1)	True Positives (TP)	False Positive (FP)
Predicted Negative (0)	False Negatives (FN)	True Negatives (TN)

Çalışmaların sonuçlarını ENAMEX veri etiketleri ile ölçüm uygulanabilmesi amacıyla, IOB2 yapısına dönüştürülmüştür. Ölçme sırasında CoNLL-2003 ölçme

stratejisi uygulanmıştır. Eğitim sırasında 3 farklı girdi yapısı oluşturulmuştur. İlk başta sadece sözcük vektörleri tek başına kullanılmış, daha sonra buna yazım özellikleri eklenmiş ve en son olarak da bu ikisine biçim bilimsel özellikler eklenmiştir. En iyi sonuç, 3 girdinin birlikte kullanıldığı zaman, F1 skoru %93,69 olarak elde edilmiştir. Bununla birlikte özellikle yazım özelliklerinin modele eklenmesinden sonra önemli bir artış olduğu gözlenmiştir. Ulaşılan başarı sonucu Türkçe için oluşturulan AVT modellerinde ulaşılmış en iyi sonuç olarak nitelendirilmiştir.

$$Precision = \frac{TP}{TP+FP} \tag{2.1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.2}$$

$$F_1 = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{2.3}$$

Başarı unsuru olarak doğru pozitif (True Positive-TP), doğru negatif (True Negative-TN), yanlış pozitif (False Positive-FP), yanlış negatif (False Negative-FN) değerleri baz alınmıştır. Bu değerler kullanılarak keskinlik, hassasiyet ve F1 skor değerleri tanımlanmıştır. Başarı unsurlarının tanımları ve skor değerlerinde nasıl kullanıldıkları aşağıda verilmiştir:

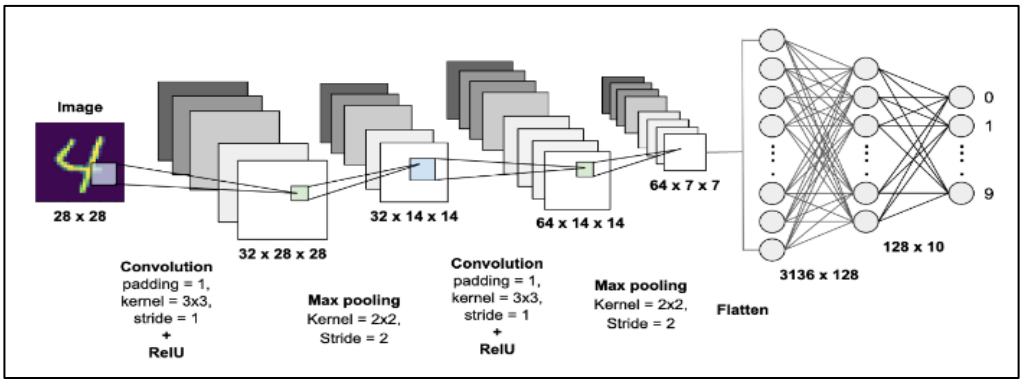
Doğru pozitifler doğru tahmin edilen pozitif değerlerdir, yani gerçek sınıfın değeri pozitif ve tahmin edilen sınıfın değeri de pozitiftir. Doğru negatifler gerçek sınıfın değerinin negatif ve tahmin edilen sınıfın değerinin de negatif olduğu anlamına gelen doğru tahmin edilen negatif değerlerdir. Yanlış pozitifler gerçek sınıf negatif ve tahmin edilen sınıf pozitif olduğunda kullanılır. Yanlış negatifler gerçek sınıf pozitif, ancak tahmin edilen sınıf negatif olduğunda kullanılır. Kesinlik, doğru tahmin edilen pozitif gözlemlerin toplam tahmin edilen pozitif gözlemlere oranıdır. Hassasiyet, doğru tahmin edilen pozitif gözlemlerin gerçek sınıftaki tüm gözlemlere oranıdır. F1 skoru, kesinlik ve hassasiyetin harmonik ortalamasıdır. Bu nedenle, bu puan hem yanlış pozitifleri hem de yanlış negatifleri hesaba katar.

Shen, Yun, Lipton, Kronrod ve Anandkumar, (2017) makalesinde derin öğrenme (Deep Learning) yöntemi ile aktif öğrenme yöntemi birleştirilerek, bir AVT uygulaması yapılmıştır. Genellikle derin öğrenme yapabilmek için çok sayıda veri gerekmektedir. İşte bu makalede bu soruna çözüm olarak oluşturulan bir yapı gösterilmiştir. Yapı ana hatlarıyla CNN-CNN-LSTM olarak tasarlanmıştır. İlk olarak Yann Le Cun tarafından önerilen CNN uygulaması resim tanıma işlemlerinde ve görüntü işlemede çok tercih edilen bir yöntemdir (LeCun vd., 1990). CNN modelinin ilk aşamasında, belirli sayıda filtre resmin RGB piksellerinden oluşan matris üzerinde gezdirilir ve daha sonra da konvolüsyon (evrişim) işlemi denklem 2.4'te gösterildiği gibi uygulanır:

$$(f * g) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau) g(t-\tau) d\tau \tag{2.4}$$

Konvolüsyon işleminden sonra filtrelenmiş matrislere max pooling işlemi uygulanır. Max pooling işleminde 2 x 2'lik bir matris, piksel matrislerin üzerinde gezdirilir ve 4 sayının içerisinden en büyük olanı bir sonraki aşamaya aktarılmak üzere seçilir. Bu aşamadan sonra düzlendirme (flattening) aşaması gelir (Şekil 2.1). Burada elde edilen matrisler tek bir uzun dizi haline getirilir. Bir başka deyişle matrisler sırasıyla arka arkaya dizilir ve daha sonra gelecek olan tam ilişkili sinir ağına (Fully Connected Neural Network) girdi olarak verilir. CNN algoritmasının görüntü işleme çalışmalarında çokça tercih edilmesinin sebebi, çeşitli filtreler ile resmin özelliklerinin çıkarılabilmesidir. Örneğin bir kedi resminin tanınması işlemini ele alacak olursak bir filtre kedinin bıyık kısmını tespit ederken, diğer bir filtre kuyruk kısmını tespit etmektedir. Bu sayede kedi resminin başka hayvan resimleriyle olan ayrımı kolayca ve başarılı bir şekilde yapılabilmektedir.

Shen vd., (2017) makalesinde, ilk CNN yapısı karakter kodlayıcı (character encoder), ikinci CNN yapısı kelime kodlayıcı (word encoder) olarak ve son olarak LSTM ise etiket çözücü (tag decoder) olarak tasarlanmıştır. Karakter kodlayıcı kelimelerin karakterlerine göre özelliklerini çıkarmak için kullanılır. Kelime kodlayıcı bir kelimenin etrafındaki kelime dizilerine bakarak özellik vektörü oluşturur. Etiket kodlayıcı ise kelimeler dizisinin oluşma veya olma olasılıklarını oluşturur.



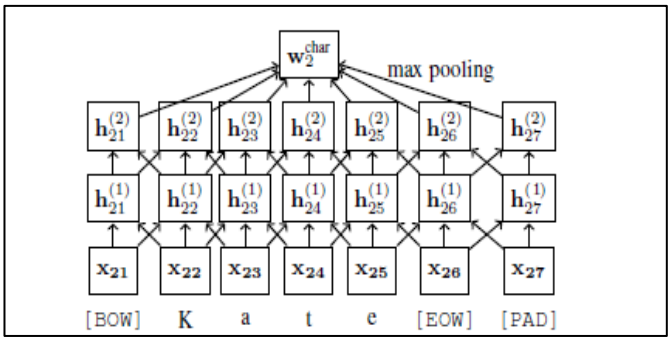
Şekil 2.1 CNN model yapısı

Çizelgede cümle içerisindeki kelimelerin nasıl temsil edildiği gösterilmektedir. Cümlenin başına BOS (Beginning of the sentence) tokeni, cümlenin sonuna ise EOS (Ending of the sentence) tokeni getirilmiştir. Bunlara ek olarak BOW (Beginning of the word) tokeni ve EOW (Ending of the word) tokenleri kullanılmıştır. Birden çok cümlenin hesaplanması için, benzer uzunluktaki cümleler gruplara ayrılmış ve uzunlukları bir demet içinde üniform hale getirmek için cümle sonunda PAD tokenleri eklenmiştir. Biçimlendirilmiş cümle $f(X_{ij})$ olarak belirtilir; burada, $\{X_{ij}\}$, i'inci sözcükteki j'inci karakter bazında bir "one-hot encoding" kodlamasıdır. Şekil 2.2'de formatlanmış cümle örneği görülebilir.

Formatted Sentence	[BOS]	Kate	lives	on	Mars	[EOS]	[PAD]
Tag	O	S-PER	O	O	S-LOC	O	O

Şekil 2.2 Formatlanmış cümle örneği (Shen vd., 2017)

Çalışmada her i kelimesinin özelliklerinin çıkartılması için CNN uygulanmıştır. Şekil 2.3'te, karakter seviyesinde kodlama (character-level encoding) için iki katmanlı örnek CNN mimarisi gösterilmektedir. Ayrıca her kelimenin karakter seviyesinde kodlama vektörü w_i^{char} ile ifade edilmektedir.

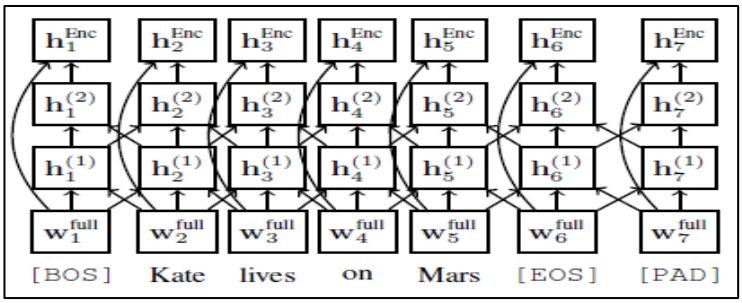


Şekil 2.3 Karakter seviyesinde kodlamaya örnek CNN yapısı (Shen vd., 2017)

Bu işlemlerden sonra karakter seviyesinde özellikler, o kelimeye karşılık gelen gömülü gizli bir kelime ile birleştirilmiştir. Bu vektör w_i^{emb} olarak adlandırılmış. Ayrıca söz seviyesinde girdiler de w_i^{full} ile ifade edilmiştir.

$$w_i^{full} := \left(w_i^{char}, w_i^{emb}\right) \tag{2.5}$$

Gizli kelime yapılarıyla word2vec eğitimi başlatılmış ve ardından eğitim süresince bu yapılar güncellenmiştir. Eğitim verilerinde gizli kelimelere genelleme yapmak için, her kelimeyi, kelime bırakma yöntemine (word-drop method) benzeyen bir yaklaşım olan, eğitim sırasında %50 olasılıkla özel bir [UNK] (bilinmeyen) tokenle değiştirilmiştir. Kelime düzeyde temsiller, her kelimenin pozisyonu kullanılarak CNN ile çıkarılmıştır. Şekil 2.4'te Kelime Seviyesinde kodlamaya örnek CNN yapısı gösterilmiştir.



Şekil 2.4 Kelime seviyesinde kodlamaya örnek CNN yapısı (Shen vd., 2017)

Yapının son kısmını da LSTM ile yapılan etiket çözücü (tag decoder) oluşturmaktadır. Etiket çözücü, kelime seviyesi kodlayıcı özelliklerine bağlı olarak

etiket dizileri üzerinde bir olasılık dağılımını oluşturur. Bu da $P\left[y_{2,}y_3 \ldots\ y_{n-1}|\{h_i^{Enc}\}\right]$ ile ifade edilmiş olur. Burada popüler olan Koşullu Rastgele Alanlar (Conditional Random Fields-CRF) algoritması etiket çözücü olarak kullanılmıştır. Bu algoritmayı açıklayan denklem 2.6 aşağıda gösterilmektedir. Bu denklemdeki W, A, b; öğrenebilen parametreler, t_i ise vektörün koordinatlarıdır:

$$P\left[y_{2,}y_3 \ldots\ y_{n-1}|\{h_i^{Enc}\}\right] \propto \exp\left(\sum_{i=2}^{n-1}\{Wh_i^{Enc}+b\}_{t_i}+A_{t_{i-1}t_i}\right. \quad (2.6)$$

Sözü geçen çalışmada öğrenme süreci birden fazla turdan oluşturulmuştur. Her turun başında, aktif öğrenme algoritması cümleleri önceden tanımlanmış limite kadar açıklanacak şekilde seçilmiştir. Ek açıklamaları aldıktan sonra, artırılmış veri kümesi üzerinde eğitim alarak model parametreleri güncellenmiş ve bir sonraki tura geçilmiştir. Bir cümleyi açıklama maliyetinin cümledeki kelime sayısıyla orantılı olduğunu ve seçilen cümledeki her kelimenin bir kerede açıklanması gerektiği varsayılmıştır. Kısmen açıklamalı cümlelere izin verilmemiş veya veri setine dahil edilmemiştir. Ayrıca bu çalışmada veri seti olarak CoNLL-2003 English ve OntoNotes-5.0 English kullanılmıştır.

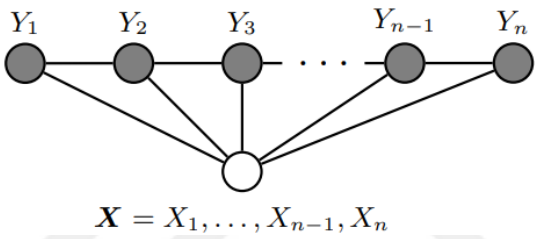
Mikolov vd. makalesinde kurulan algoritmadaki ana önemli etken hız etkenidir. Sonuç bölümünde verilen tablolara göre makale test setindeki verilerde epoch başına 11 saniye gibi aynı alandaki çalışmalara göre çok önemli bir hıza ulaşmıştır. Ayrıca eğitim setinde de epoch başına 22 saniyelik hız ile aynı alandaki çalışmalara göre birçok çalışmadan daha hızlıdır. Hız anlamında eşit olduğu çalışmalarda da F1 skoru başarısı (%90,69) daha yüksek olarak gözlenmiştir.

Özkaya ve Diri (2011) makalesinde Şartlı Rastgele Alan (Conditional Random Field-CRF) yöntemi kullanılarak, kural tabanlı AVT işlemi gerçekleştirilmiş ve eposta metinlerindeki cümlelerde AVT üzerine çalışılmıştır. Sıralı veri, söz dizimsel analiz kullanılarak etiketlenmiştir.

<İsim>Ahmet<İsim><TamlayanEki>in<TamlayanEki><İsim>baba
<İsim><TamlananEki>sı<TamlananEki><Sıfat>beyaz<Sıfat><İsim>k
oyun<İsim><NesneEki>u<NesneEki><İsim>araba<İsim><DiğerZarfl
ar>ile<DiğerZarflar><İsim>köy<İsim><DolaylıTümleçEki>e<Dolaylı

Şekil 2.5 Söz dizimsel analiz kullanarak etiketlenme örneği

Şartlı Rastgele Alan algoritmasında dizili kelimeleri işaretlemek veya bölümlendirmek amacıyla kullanılan, Maksimum Gizli Markov Modeli ve Entropi Markov Modelinin genel durumunu gösteren bir olasılık ortaya çıkar. CRF Şekil 2.6'da gösterildiği gibi, y gibi belirli bir işaret dizisinin x değeriyle şartlı olasılığını hesaplamak için yönsüz çizge modeli kullanılmaktadır.



Şekil 2.6 Diziler için zincir yapılı CRF'lerin grafik yapısı (Wallach, 2004)

Bu çalışmada veri seti akademik, kurumsal ve kişisel olmakla birlikte toplam 150 e-posta metni ile çalışılmıştır. Önemli özelliklerin ortaya konmasına yardımcı olan unvanların, bazı özel kelimelerin ve kısaltmaların olduğu sözlüklere de ihtiyaç duyulmuştur. O yüzden, bu çalışmada 175 adet kısaltma listesi (İng., Fr., Tşk, Sok., Mah., TD, Sn, A.Ş., Ltd.,.), 35 adet unvan listesi (Prof., Dr., Av., Gen.,.), ve 32 adet özel kelimeler listesi (Sayın, Hanım, Bey, Hocam, Üniversite, Bakanlık, Hastane, Dağ, Tepe, vb.) kullanılmıştır.

Çalışmada e-posta metinlerindeki kelime özelliklerini elde ederken 3 ana yapıdan yararlanılmıştır. Bunlar başlık bilgisi, etiketlenebilen özellikler ve kural tabanlı özelliklerdir. Başlık bilgisi “From, To, Cc, Bcc, Send, Sender, In-Reply-to, Reply-to, Forwarded by” gibi başlık kalıplarından sonra varlık adı geleceği düşünülerek o kısımlarda varlık isimleri araştırılmıştır.

Etiketlenen özellikler kısmında kelimenin büyük harf ile başlayıp başlamaması, içinde nokta, virgül gibi noktalama işareti içerip içermemesi, cümlenin ilk kelimesi olup olmaması, kelimenin başlık bilgisinde olup olmaması, kelime unvan listesinde veya özel kelimeler listesinde yer alıp almaması gibi filtreler ile özellikler oluşturulmuştur.

Kural tabanlı özelliklerde her kelimeden önce ve sonra gelen kelimeler belirlenerek isim, soy isimler yakalanmaya çalışılmıştır. Daha sonra ise 1'li, 2li ve 3'lü n-gramlara bakılarak bazı son ekler belirlenmeye çalışılmıştır. Ayrıca kelime uzunluğuna bakılarak da o kelimenin kısaltma veya unvan adı olup olmayacağının tahmininde işe yarayacağı düşünülmüştür. Son olarak da art arda gelen kelimelerde isim olarak nitelendirilebilecek kelimenin hangisinin isim, hangisinin soy isim olabileceği belirlenmeye çalışılmıştır.

Bu çalışmada üç farklı sınıf (yer, kişi, kurum ismi) için 50 farklı eposta taranmıştır. Adlandırılmış varlık tanınmasında en başarılı sonuçlar %87 ile kurumsal epostalar olmuştur. En düşük doğru tanıma oranı ise %72 ile kişi adlarında olmuştur. Bu çalışmada doğru tanımadaki en büyük etkenlerden genellikle noktalama işaretlerinin doğru yerde kullanılması ve yazım yanlışı yapılmaması olduğu vurgulanmıştır.

Schiersch vd., (2020) çalışmasında, sokak adları, durak adları ve güzergâh adları gibi coğrafi varlıklarla ve standart adlandırılmış varlık türleriyle açıklamalı Almanca veri seti oluşturulmuş. Ayrıca, kazalar, trafik sıkışıklıkları, satın almalar ve grevler gibi trafik ve endüstri ile ilgili n-ary (birbiri ile ilişkili n adet varlık adının bulunması) ilişkileri ve olayları içeren bir dizi açıklama eklenmiştir. Veri seti genel olarak çevrimiçi gazete, radyo istasyonları, polis ve demiryolu şirketlerinden gelen haber metinleri, Twitter mesajları ve trafik raporlarından oluşturulmuş. Bu çalışma coğrafi varlıkların açıklanmasını amaçlayan hem AVT algoritmalarının hem de n-ary ilişki çıkarma sistemlerinin eğitimine ve değerlendirilmesine olanak tanıyan bir özelliğe sahiptir.

Veri setini oluşturmak için Twitter Search API, uberMetrics Search API, RSS Feeds gibi metin sağlayıcı servisler kullanılmıştır. Sadece Almanca metinleri çekebilmek için python ile yazılmış "langid" kütüphanesi kullanılmıştır. AVT çalışması için Stanford Core NLP (Manning vd., 2014) araçlarından yararlanılmıştır. Ayrıca varlıklar arasında ilişki çıkarımı yapmak için DARE algoritması kullanılmıştır. DARE, kural öğrenme ve ilişki çıkarmadan oluşan, serbest metinler üzerinde ilişki çıkarımı için minimal denetimli bir makine öğrenme sistemidir. İlişki çıkarma algoritması DARE, tüm ilişki argümanlarını birbirine bağlayan minimum bağımlılık alt grafiklerini öğrenir (Krause, Li, Uszkoreit ve Xu, 2012).

Sarı ve Aktaş (2018) makalesinde ders içeriği olarak hazırlanmış coğrafya ve tarih metinlerinin içerisinde adlandırılmış varlık adı bulunmaya çalışılmıştır. Bu çalışma da amaç girdi metinlerde varlık adlarını belirleyerek terimler sözlüğü oluşturabilmek olarak verilmiştir. Oluşturulan sistem cümleler üzerine çalışan ve kural tabanlı bir sistemdir. Cümleleri belirlemek için öncelikle gereksiz boşluk, karakter ve semboller çıkarılmıştır. Cümle sonlarını belirlemek için regex (regular expression) kullanılmıştır. Regex işlemi sırasında oluşabilecek hatalar için Türkçe kısaltmaların olduğu bir sözlükten yararlanılmıştır. Cümle sonları belirlendikten sonra birimleştirme (tokenizer) işlemi yapılmıştır. Sözcükleri birimleştirmek için her sözcüğün; büyük harf içerip içermediği, tamamının büyük harften oluşup oluşmadığı, son karakterinin nokta olup olmaması, sayı içerip içermemesi (içerdiği sayıların gün ay yıl sayısı olması da işaretlenmiştir), kesme işareti, virgül, noktalı virgül, parantez içermesi, yüzde içermesi dikkate alınmıştır.

Wang vd. (2016) çalışmasında Twitter'dan toplanan 100 milyon mesaj kullanılarak metin içerisinde geçen hashtag, duygusal sınıflandırma etiketi olarak kabul edilmiştir. Ayrıca, emoji ve N-gram gibi özellikler çıkarılmış ve toplanan konu mesajları, dolaylı duygusal modele dayalı olarak dört farklı duygu kategorisine sınıflandırılmıştır. Son olarak, duygusal veri setini sınıflandırmak için makine öğrenmesi yöntemleri kullanılmış ve %89 doğruluk sonucu elde edilmiştir. Bu çalışmada AVT yapmak için metinler ön işlemden geçirilmiştir. İlk önce kullanıcı adları ve web adresleri silinmiştir. Yazım hataları olan ve arka arkaya ikiden fazla harfe sahip olan kelimeleri

tweetlerin yanlış kelimeler sözlüğünü kullanarak düzeltilmiştir. “good4you” vb. kısaltmalar düzeltilerek doğrusu ile yer değiştirilmiştir. Ve son olarak duygusal sınıflandırmada ki etiketleme hataları düzeltilmiştir. Burada kelimeler “Mutlu Aktif”, “Mutlu Pasif”, “Mutsuz Aktif” ve “Mutsuz Pasif” olmak üzere dört farklı sınıfa ayrılmıştır. Kullanılan algoritmalar ise Naïve Bayes, Logistic regression, SVM ve KNN algoritmalarıdır. AVT ksımında ise %95 civarında başarı elde edilmiştir.

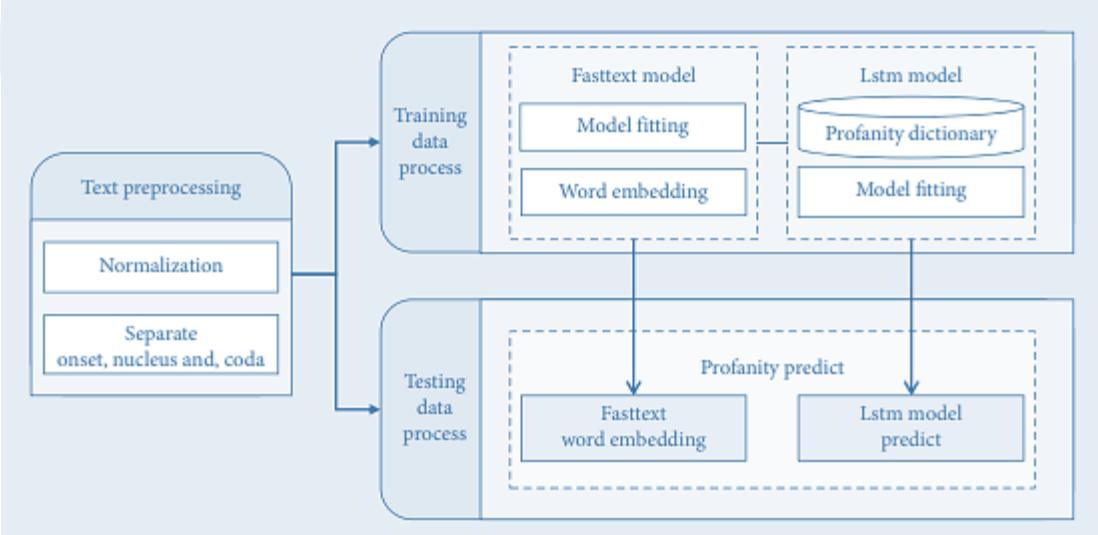
Satheesh vd. (2020) çalışmasında özgeçmiş tarama süreci makine öğreniminin bir alanı olan gelişmiş doğal dil işleme kullanılarak otomatikleştirilmiştir. Model, işe alım görevlilerinin iş tanımına göre özgeçmişleri kısa sürede taramalarına yardımcı olmuştur. Özgeçmişlerden Spacy NER modelini kullanarak gerekli varlıkları otomatik olarak çıkararak işe alım sürecini kolay ve verimli hale getirmiştir ve ardından her bir özgeçmişin puanını gösteren bir grafik oluşturulmuştur. Puanlara dayanarak, işe alım görevlisinin niteliksiz adaylardan gelen özgeçmiş yığınlarını karıştırmadan gerekli adayları seçebileceği söylenmiştir. Çalışmada modeli eğitmek için manuel olarak açıklamalı eğitim verileri oluşturulmuştur. Bu nedenle, belgeleri otomatik olarak ayrıştıran ve gerekli varlıkların ek açıklamalarını oluşturmaya izin veren Dataturks adlı bir çevrimiçi otomasyon aracı kullanılmıştır.

2.2 Küfür tespiti üzerine literatür incelemesi

Siber uzayda kullanıcı tarafından oluşturulan içeriğin miktarı 21. yüzyılda artmaya devam ediyor. Kişiler tarafından saygısız metinlerin kullanılması dijital alanın özgürlüğünü ve bütünlüğünü tehdit etmektedir. Bu tür saygısız metinleri kontrol etmek için geleneksel olarak manuel denetleme ve raporlama mekanizmaları kullanılmıştır. İnsan yorumuna bağımlılık ve sonuçların gecikmesi bu sistemin önündeki en büyük engeller olmuştur. Süreci otomatikleştirmek için derin öğrenme tabanlı yaklaşımlar, geleneksel evrişim ve yineleme tabanlı sıralı modellerin kullanımı oldukça yaygındır.

Yi, Lim, Ko ve Shin (2021) makalesinde kelime gömme ve LSTM modelini kullanarak bir küfür tespit yöntemi önerilmiştir. Önerilen yöntem, eğitim yapmak için metni “onset”, “nucleus” ve “coda” olarak 3 ayrı parçaya bölmüştür (Şekil 2.7). Ayrıca, FastText modelini kullanarak sadece kelimelerin anlamlarını değil aynı

zamanda morfoloji bilgilerini de dikkate alınmaktadır. Ayrıca, LSTM modelini kullanarak bağlam akışı konusunda eğitim yaparak, önceki çalışmalarda önerilen metodolojilerle tespit edilemeyen küfürleri tespit edilmiştir. Önerilen yöntemle 40.005 küfürlü ve 40.254 küfürsüz cümleden 40.126 cümle küfür, 40.133 cümle küfür değil olarak tahmin edilmiştir. Sınıflandırma performans testi göstergelerine göre doğruluk %96,15, geri çağırma oranı %96,29 ve kesinlik %96 olarak tespit edilmiştir. Önceki bir çalışmada önerilen düzenleme mesafesi (edit distance) algoritması ile bu çalışmada önerilen yöntem arasındaki karşılaştırmalı analizin sonucu, önerilen yöntemin daha doğru küfür tespiti yapabildiğini doğrulamıştır. Şekil 2.7'de önerilen LSTM ve kelime gömme modelinin yapısı gösterilmektedir.



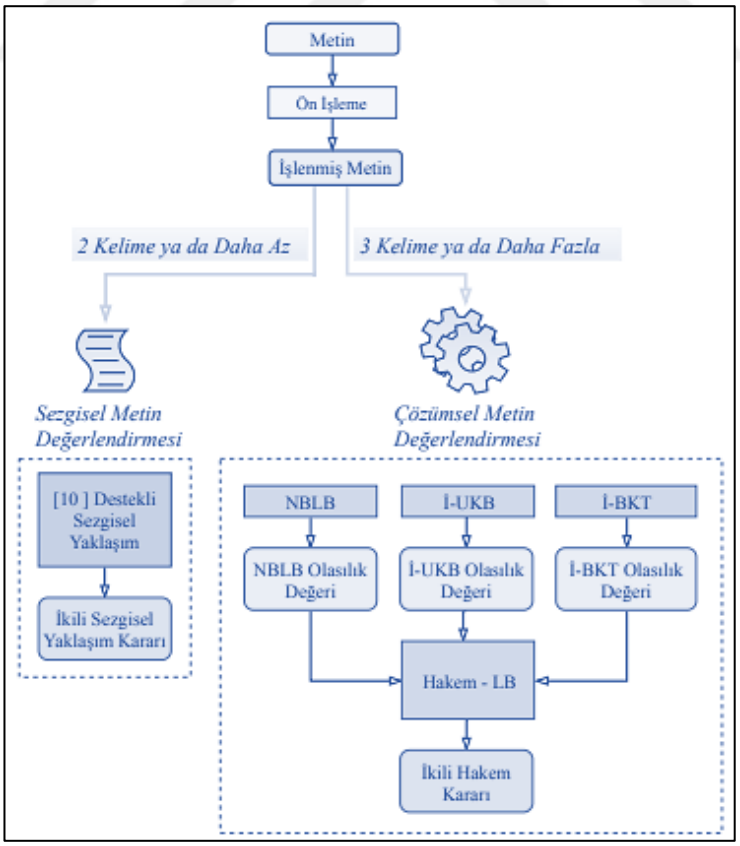
Şekil 2.7 Küfür tespiti için önerilen model (Yi, vd. 2021)

Ratadiya ve Mishra (2019) çalışmasında küfür içeren metinlerin tespiti için çok başlı dikkat temelli (attention-based) bir yaklaşım önerilmişlerdir. Performansı daha da iyileştirmek için model güç ağırlıklı ortalama birleştirme teknikleri ile birleştirilmiştir. Önerilen yaklaşımın ek bellek gereksinimi yoktur. Ayrıca model önceki yaklaşımlara kıyasla daha az karmaşıktır. Model tarafından kamuya açık gerçek dünya verileri üzerinde elde edilen iyileştirilmiş sonuçların aynı şeyi daha da doğruladığı ifade edilmiştir. Bu çalışma da gösterilen katkılar üç kısımda sıralanmıştır. Çoğu yaklaşımda kullanılan tekrarlama mekanizmasını tamamen atlanmasına rağmen iyi sonuçlar elde edebilmiştir. Tekrarlama olmadığında diziyle ilgili bilgileri sağlamak için konumsal kodlama etkin bir şekilde kullanılmıştır. Son olarak, birleştirme

kullanarak padding'e rağmen bir dizide bulunan maksimum bilginin etkili bir şekilde tutulduğu gösterilmiştir. Önerilen yöntemde, her modelin tahminine ve modelin doğruluğuna göre bir ağırlık atanır. Bireysel ağırlık değeri 0 ile 1 arasındadır ve tüm modellere atanan ağırlıkların toplamı 1'e eşit olmalıdır. Tahminlerin ağırlıklarla çarpımlarının toplamı nihai tahmin olarak kabul edilmektedir. Ağırlıklı ortalama tahminleri denklem 2.7'de gösterilmiştir. P_i, i'inci modelin tahminini ve W_i ise i'inci modele atanan kat sayıyı göstermektedir.

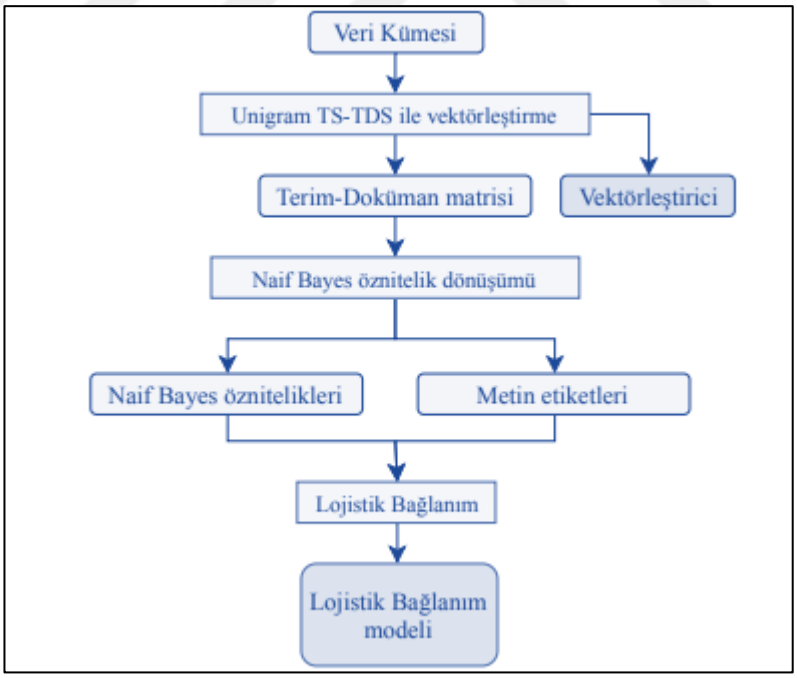
$$W\,eightedAverageP\,rediction = \sum_{i=1}^{n} P_i W_i \tag{2.7}$$

Türkçe küfür tespit çalışmalarından birisini de Çelik ve Yıldırım (2020) gerçekleştirmiştir. Çalışmada önce metin ön işlemlerden geçirilmiştir. Daha sonra eğer kelime sayısı 2 veya daha az ise metin sezgisel modele, 3 kelime ya da daha fazla ise yapay zekâ modeline yönlendirilmiştir. 3 faklı yapay zekâ modelinin döndürdüğü olasılıklara bakarak hakem modelin karar vermesi sağlanmıştır. Şekil 2.8'de çalışmanın ana yapısı gösterilmektedir.



Şekil 2.8 Türkçe küfür tespit için örnek model yapısı Çelik ve Yıldırım (2020)

Çelik vd., (2020) çalışmasında metin ön işleme aşamasında kelimeler küçük harf yapılmış, http benzeri bağlantılar kaldırılmış, hatalı kelimeler düzenlenmiş, dolgu kelimeleri yok edilmiş, metnin içerisinde geçen rakamlar silinmiş, kelimelerdeki yinelenen harfler yeniden düzeltilmiş, kısaltma olarak yazılmış kelimeler yeniden düzeltilmiş, noktalama işaretleri silinmiş, tek harfli heceler ve tüm fazla olan boşluklar kaldırılmıştır. Yazarlar çalışmalarının sezgisel model kısmında Ratcliff-Obershelp algoritması yardımıyla metin içerisindeki kelimeler ile küfür listesi içerisindeki kelimelerin dizi benzerliğini (string similarity) bulmuştur. Elde edilen dizi benzerliği belirlenen eşiği aşarsa sözcük küfür olarak tanımlanmıştır. Ayrıca java tabanlı Türkçe metin işleme kütüphanesi olan zemberek kullanılarak kelimelerin sonuna farklı Türkçe ekler getirilmiş ve dizi benzerlikler o şekilde elde edilmiştir. Yapay zekâ tabanlı model kısmında ise 3 farklı model kullanılmıştır. Bunlar: Naif Bayes tabanlı Lojistik Bağlanım (NBLB), İki Yönlü Uzun Kısa Süreli Bellek (İUKB) ve İki Yönlü Kapılı Tekrarlayan Hücre (İBKT). Son kısım ise lojistik tabanlı hakem bir model ile karar verilmektedir.



Şekil 2.9 NBLB modeli çalışma yapısı (Çelik vd., 2020)

Şekil 2.9'da NBLB modelinin çalışma yapısı gösterilmiştir. İlk aşamada metin Terim Sıklığı-Ters Dokuman Sıklığı (TS-TDS) yöntemleri kullanılarak

vektörleştirilir. Daha sonra oluşturulan vektörler Naive Bayes özellik dönüştürücüye iletilmektedir. Etiketler ve özellik dönüştürücüden geçirilen vektörler hakem modele beslenir. İkinci model olan Çift Yönlü Uzun Kısa Bellek yönteminin ayrıntılarına bölüm 3.2'den ulaşılabilir. Yazarların bu yöntemi tercih etmesinin sebebi hafıza hücreleri sayesinde metindeki uzak anlamsal bağları yakalamayı hedeflemektir. Ayrıca bu modelde en iyi sonucu elde edebilmek için hiper parametre optimizasyonu gerçekleştirilmiştir.

Son model olan İBKT modeli ise anlamsal bağları yakalamak için ve ikinci model İUKB'den daha hızlı olduğu için tercih edilmiştir. Bu iki model yapısal olarak birbirine yakın oldukları için birbirlerinin çıktılarını pekiştirmişlerdir. İBKT modeli de İUKB modeli gibi benzer bir parametre optimizasyonundan geçirilmiştir. Sonuçta model küfür tespitinde Hakem model %96,1, İBKT modeli %95,5, İUKB modeli %95,8, NBLB modeli ise %95,5 f1 skoru doğruluk oranına sahiptir.

BÖLÜM ÜÇ

DOĞAL DİL İŞLEME

3.1 Doğal Dil İşlemenin Tanımı

Tarihsel olarak bilgisayarların insan dilini anlaması zor bir süreç olagelmiştir (Jones, 1994; Joseph, Hlomani, Letsholo, Kaniwa ve Sedimo, 2016). Elbette bilgisayarlar metin girdilerini toplayabilir, saklayabilir ve okuyabilir ancak temel dil bağlamından yoksundurlar. Temel dil bağlamını daha iyi anlamak ve işlemek amacıyla doğal dil işleme (Natural Language Processing-NLP) geliştirilmiştir. Doğal dil işleme bilgisayarların metinleri ve konuşulan kelimeleri okumasını, analiz etmesini, yorumlamasını ve anlam türetmesini amaçlayan yapay zekâ alanı olarak adlandırılabilir (Joseph vd., 2016). Bu uygulama, bilgisayarların insan dilini anlamasına yardımcı olmak için dilbilimini, istatistikleri ve makine öğreniminini kullanır. Doğal dil işleme teknolojileri bilgisayarların insan dilini metin veya ses verileri biçiminde işlemesini sağlar. Ayrıca bilgisayarların konuşmacının veya yazarın niyetini anlamasına olanak tanır (Miller, 2019; Özkaya vd., 2011; Ratadiya vd., 2019).

Doğal dil işleme, metni bir dilden diğerine çeviren, sözlü komutlara yanıt veren ve gerçek zamanlı olarak bile büyük hacimli metni hızla özetleyen bilgisayar programlarını çalıştırır. Doğal dil işleme ile sesle çalışan GPS sistemleri, dijital yardımcılar, konuşmadan metne dikte yazılımı, müşteri hizmetleri sohbet robotları ve diğer tüketici kolaylıkları biçiminde pek çok gerçek hayat probleminde kullanılmaktadır (Joseph vd., 2016). Bunun yanında doğal dil işleme, iş operasyonlarını kolaylaştırmaya, çalışan üretkenliğini artırmaya ve kritik iş süreçlerini basitleştirmeye yardımcı olan kurumsal çözümlerde de büyüyen bir rol oynamaktadır (Stenetorp vd., 2012).

En eski doğal dil işleme uygulamaları, belirli görevleri yerine getirebilen, ancak görünüşte sonsuz bir istisna akışına veya artan metin ve ses verisi hacmine uyum sağlamak için kolayca ölçeklenemeyen, elle kodlanmış, kurallara dayalı sistemlerdi (Jones, 1994). Metin ve ses verilerinin öğelerini otomatik olarak ayıklamak,

sınıflandırmak ve etiketlemek için bilgisayar algoritmalarını makine öğrenimi ve derin öğrenme modelleriyle birleştiren istatistiksel yöntemler ile doğal işleme çalışmalarına son dönemde oldukça büyük bir ilgi mevcuttur (Alaparthi ve Mishra 2021; Çelik ve Yıldırım, 2020). Günümüzde, evrişimli sinir ağlarına (CNN) ve tekrarlayan sinir ağlarına (RNN) dayalı derin öğrenme modelleri ve teknikleri, çalıştıkça öğrenen ve çok büyük hacimli yapılandırılmamış ve etiketlenmemiş metinlerden ve ses veri kümelerinden, doğru anlamlar çıkaran doğal dil işleme sistemlerine olanak tanımaktadır. (Yin, Kann, Yu ve Schütze, 2017)

İnsanların öğrenmesi yıllar alan dil kalıplarından birkaçı; eş sesli sözcükler, alaycılık, deyimler, metaforlar, dilbilgisi ve kullanım istisnaları, cümle yapısındaki farklılıklar olarak sayılabilir. Bazı doğal dil işleme görevleri, insan metinlerini ve ses verilerini, bilgisayarın aldığı şeyleri anlamlandırmasına yardımcı olacak şekilde parçalamaktadır. Bu kullanım alanlarından bazıları şu şekilde sıralanabilir: Konuşmadan metne çeviri yapma, sözcük türlerini dil bilgisel olarak etiketleme (part of speech tagging), adlandırılmış varlık tanıma, eş referans çözümlemesi, metinlerde duygu analizi bu kullanım alanlarından birkaçıdır (Joseph vd., 2016).

Konuşmadan metne olarak da adlandırılan konuşma tanıma, ses verilerini güvenilir bir şekilde metin verilerine dönüştürme görevidir (Adam, 2020). Sesli komutları takip eden veya sözlü soruları yanıtlayan tüm uygulamalar için konuşma tanıma gereklidir. Konuşma tanımayı özellikle zorlaştıran şey, insanların hızlı konuşması, kelimeleri farklı vurgu ve tonlamalarla yapması, farklı aksanlarda geveleyerek ve sıklıkla yanlış dilbilgisi kullanmasıdır (Adam, 2020).

Sözcük türlerini dil bilgisel olarak etiketleme olarak da adlandırılan konuşma etiketlemenin (part of speech tagging) bir kısmı, belirli bir kelimenin veya metin parçasının kullanım ve bağlamına dayalı olarak konuşmanın bir bölümünü belirleme sürecidir (Brants, 2000). Sözcük türü etiketi sayesinde kelimenin cümle içerisinde ki görevi anlaşılabilir ve bu sayede AVT gibi farklı doğal dil işleme görevleri gerçekleştirilebilir (Deshmukh ve Kiweleka, 2020).

Adlandırılmış varlık tanıma, kelimeleri veya tümceleri faydalı varlıklar olarak tanımlar. Örneğin İzmir'i bir yer olarak veya Ahmet'i bir kişi adı olarak tanımlar (Rau, 1991).

Eş referans çözümlemesi, iki kelimenin aynı varlığa atıfta bulunup bulunmadığını ve ne zaman atıfta bulunduğunu belirleme görevidir (Sukthanker, Poria, Cambria ve Thirunavukarasu, 2020). En yaygın örnek, belirli bir zamirin atıfta bulunduğu kişiyi veya nesneyi belirlemektir. Ancak aynı zamanda metindeki bir metaforu veya deyimi tanımlamayı da içerebilir. Örneğin, "ayı bir hayvan değil, büyük tüylü bir insandır" (Sukthanker, vd., 2020).

Duygu analizi, metinden öznel nitelikler tutumları, duyguları, alaycılığı, kafa karışıklığını, şüpheyi çıkarmaya çalışır (Zhou ve Wu, 2018). Doğal dil işleme teknikleri ile doğal dil oluşturma, bazen konuşma tanımanın veya konuşmadan metne dönüşümün tersi olarak tanımlanır; yapılandırılmış bilgiyi insan diline yerleştirme görevidir (Alaparthi ve Mishra, 2021)

3.2 Adlandırılmış Varlık Tanıma Çalışmalarında Etiketleme Yöntemleri

AVT, doğal dil işleme için önemli bilgileri çıkarma işlemidir (Rau, 1991). Bu işlemde belgede adı geçen kişi adları, kuruluşlar, konumlar, miktarlar, parasal değerler, sayısal değerler, yüzde ve diğer tüzel kişilik türleri gibi varlıklar, NLP algoritmaları aracılığıyla makineler tarafından tanınabilir ve anlaşılabilir hale getirilir (Güneş vd., 2018). AVT kullanılan sürece bağlı olarak çalışır. Amaç, belgede belirtilen tüm varlıkların önemli bilgilerini çıkarmaktır. Aslında AVT, varlıkları tanımlayıp konumlandırarak yapılandırılmış ve yapılandırılmamış metinleri işler. Bu nedenle, temel olarak, önce varlıkları tanımlamak ve ardından bunları kişi, kuruluş veya konum gibi belirli bir kategoriye ayırmak için sınıflandırmaya çalışır (Güneş vd., 2018).

AVT sistemi oluşturmak için gereken en önemli etkenlerden bir tanesi ise etiketlenmiş veridir (Luo vd., 2011; Özkaya vd., 2011). AVT çalışmalarında, yapay zekâ ve makine öğrenmesi kullanılan sistemlerde varlık sınıflarını tanıtmak ve sonuçların doğruluğunu tespit etmek için çok miktarda etiketlenmiş veriye ihtiyaç

duyulmuştur (Pawar, Srivastava ve Palshikar, 2012). Hem daha iyi sonuçlar elde etmek için hem de değerlendirme yapmak için pek çok varlık etiketleme yöntemi bulunmuştur. Çalışmanın bu bölümünde de AVT çalışmalarında kullanılan bazı etiketleme yöntemleri anlatılmıştır.

3.2.1 Sözcük Türü Etiketleme (Part of Speech Tagging)

Metinler anlamsal olarak benzer cümlelerden oluşur. Bir başka deyişle metinler kendi içinde anlam bütünlüğüne sahip cümlelerden meydana gelir. Cümleler ise anlamsal ve yapısal olarak birbirini tamamlayan sözcüklerden oluşur. Sözcükler cümlede pek çok görevde kullanılmaktadır. Sözcüğün görevi (fiil, isim, edat, bağlaç vb.) anlamına ve cümle içinde bulunduğu konuma göre değişebilmektedir. İşte bu yüzden sözcüğün cümlede ki görevini tanımlama işlemine sözcük türü etiketleme denir (Brill, 1992).

Sözcük türünün etiketlenmesi farklı dillerde farklı şekillerde olabilmektedir. Örneğin İngilizce'de "the", "a" ve "an" gibi kelimeler Türkçe'de bulunmamaktadır. Anlamsal olarak önüne geldiği sözcükleri belirli ya da belirsiz olarak niteleyen bu kelimeler İngilizce'de "article" olarak nitelendirilmektedir (Dos Santos ve Zadrozny, 2014). Bu sözcük türünün Türkçe'de tam olarak karşılığı bulunmamaktadır. Şekil 3.1'de İngilizce dilinde örnek bir sözcük etiketleme türü gösterilmiştir. Aşağıda bazı sözcük türleri ve örnek sözcükler verilmiştir.

Sıfat: güzel, yeşil, harika…

Edat: için, ile, içinde…

Zarf: pek, yukarıda, yakında…

Bağlaç: ve, ama, henüz…

İsim: kedi, maymun, elma…

Zamir: o, ben, sen…

Fiil: olmak, çalışmak, durmak…

Şekil 3.1 Sözcük türü etiketleme örneği (Dos Santos vd., 2014)

Doğal dil işleme çalışmalarında sözcük türü etiketleme bir anlam ayrımı görevidir. Bir kelimenin birden çok tür etiketi olabilir. Amaç, mevcut bağlam göz önüne alındığında doğru etiketi bulmaktır. Örneğin "adamın solu" derken isim olan kelime, "gül soldu" derken fiil olarak nitelendirilebilmektedir. Sözcük türü etiketleme işlemi gerçekleştirirken pek çok farklı yöntem kullanılmaktadır (Adam, 2020). Aşağıda sözcük türleri etiketlemek için kullanılan yöntemler anlatılmıştır.

El ile etiketleme, bir cümledeki her kelimeye bir etiket uygulayan insanların sözdizimi kurallarında bilgi sahibi olması anlamına gelir. Bu zaman alıcı ve eski usul otomatik değildir (Brill, 1995). Bu yöntem aynı zamanda üç ve dördüncü sözcük türü etiketleme yöntemlerinin temelini oluşturmaktadır (Brants, 2000; Lafferty, McCallum ve Pereira, 2001).

Sözcük türü etiketleme süreci bir dizi kural kullanarak sözcük türünü belirleme işlemidir (Brill, 1992). Örneğin önceki kelime bir artikel ise ve sonraki kelime bir isim ise, o zaman bir sıfattır vb. kurallar kullanarak sözcük türü etiketlenmeye çalışılır (Brill, 1992). Bu işlem bir uzman tarafından yapılmalıdır ve kolayca karmaşık hale gelebilmektedir. "En ünlü ve yaygın olarak kullanılan kural tabanlı etiketleyici" konumu genellikle E. Brill's Tagger'a atfedilirmiştir (Brill, 1992).

Olaslıksal kelime etiketleme, bir kelimenin belirli bir etikete ait olma olasılığına dayalı olarak veya bir kelimenin bir önceki veya sonraki kelime dizisine dayalı bir etikete ait olma olasılığına dayalı olarak bir kelimeyi sözcük türüne atamanın otomatik yoluna verilen isimdir (Brants, 2000). Bunlar şimdiye kadar tercih edilen, en çok

kullanılan ve en başarılı yöntemlerdir. Ayrıca uygulanması daha basit olanlardır (Brants, 2000). Bu yöntemler arasında iki tür otomatik olasılıksal yöntem tanımlanabilir: Ayrımcı Olasılıksal Sınıflandırıcılar (Lojistik Regresyon, SVM ve Koşullu Rastgele Alanlar) ve Üretken Olasılıksal Sınıflandırıcılar (Naive Bayes ve Hidden Markov Modelleri-HMM) (Adafre ve Rijke, 2005).

Akıllı etiketleme ise sözcük türleri etiketlerini çıkarmak için derin öğrenme tekniklerini kullanan yöntemlerdir. Bu temsiller normalde sinir ağları kullanılarak öğrenilir ve kelimeler hakkında sözdizimsel ve anlamsal bilgileri yakalar. Kelime temsillerini öğrenirken kelime morfolojisi ve şekli hakkındaki bilgiler normalde göz ardı edilir. Özellikle morfolojik olarak zengin dillerle uğraşırken, konuşmanın bir bölümünü etiketleme gibi görevler için, sözcük içi bilgi son derece yararlıdır (Dos Santos vd., 2014). Bu amaçla derin öğrenme yöntemleri kullanılarak sözcük türü etiketleme yapılır. Şimdiye kadar, derin öğrenme içeren yöntemlerin sözcük türü etiketlemede olasılıksal kelime etiketleme yöntemlerinden üstün olduğu gösterilmemiştir (Deshmukh vd., 2020; Schmid, 1994).

3.2.2 IO Etiketleme

IO etiketleme AVT çalışmalarında pek yaygın olarak kullanılmamaktadır. Diğer varlık etiketleme yöntemlerine göre temel seviyede bir varlık adı etiketleme çeşididir. Kelime sayısına ve grubun büyüklüğüne bakmaksızın varlık adı (içinde-Inside) ve varlık adı değil (dışında-Outside) olarak iki grupta etiketleme yapılmaktadır (Adafre vd., 2005).

3.2.3 IOB Etiketleme

IOB formatı, (iç, dış, başlangıç kısaltması), hesaplamalı dilbilimde bir yığınlama görevinde belirteçleri etiketlemek için yaygın bir etiketleme biçimidir. Bu yöntem ilk defa, Ramshaw ve Marcus (1999) tarafından makalelerinde sunulmuştur. Bir etiketin önündeki I, etiketin bir yığının içinde olduğunu gösterir. Bir O etiketi, bir belirtecin hiçbir parçaya ait olmadığını gösterir. Bir etiketten önceki B öneki, etiketin, aralarında O etiketleri olmadan hemen başka bir parçayı izleyen bir yığının başlangıcı olduğunu belirtir. Yalnızca bu durumda kullanılır: O etiketinden sonra bir yığın geldiğinde,

yığının ilk simgesi I- önekini alır. Şekil 3.2'de IO ve IOB etiketleme yapısına örnek etiketlenmiş cümleler gösterilmiştir (Ramshaw vd., 1999).

	IO Kodlama	IOB Kodlama
Mehmet	PER	B-PER
Edvard	PER	B-PER
Munch	PER	I-PER
'un	O	O
resmini	O	O
Ahmet	PER	B-PER
'e	O	O
gösterdi	O	O

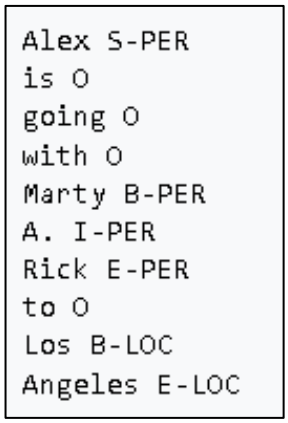
Şekil 3.2 IOB etiketleme yapısı (Ramshaw vd., 1999)

3.2.4 IOB2 Etiketleme

Yaygın olarak kullanılan diğer bir benzer biçim, her yığının başında B etiketinin kullanılması dışında IOB biçimiyle aynı olan IOB2 biçimidir (yani, tüm parçalar B etiketi ile başlar) (Sang ve Veenstra, 1999). Örnek olarak IOB etiketleme yapısı Şekil 3.2'de gösterilmiştir. Ayrıca resimde yine bir başka etiketleme çeşidi olan IO (içinde-dışında) yapısı da gösterilmiştir.

3.2.5 BIOES-BILOU Etiketleme

BIOES (Beginning, Inside, Outside, End, Single element) etiketleme yapısında IOB2 etiketlemeye ek olarak varlık adının sonu için E ve tek kelimeden oluşan varlık adı etiketlemek için S etiketi kullanılır (Yang, Liang ve Zhang, 2018). BIOES etiketleme türü BILOU türü ile benzer bir yapıda işlemektedir. L harfi "last" anlamında kelimenin sonunu etiketlemek için kullanılmaktadır (Yang vd., 2018). U harfi ise "unique" anlamında tek kelimeden oluşan varlık adlarını işaretlemek için kullanılır. Şekil 3.3'de bir cümlenin örnek BIOES etiketleme yapısı gösterilmiştir.



Şekil 3.3 Örnek BIOES etiketleme yapısı

BÖLÜM 4

YAPAY ZEKÂ VE MAKİNE ÖĞRENMESİ YÖNTEMLERİ

4.1 Yapay zekâ

Yapay zekanın doğuşu, Alan Turing'in 1950'de yayınlanan “Computing Machinery and Intelligence” adlı makale ile gerçekleşti (Turing ve Haugeland, 1950). Bu yazıda, Alan Turing genel olarak daha önce ele alınmamış bir durumu ele alıyor: "Düşünen makineler oluşturulabilir mi?". Turing bu düşündeden yola çıkarak bir bilgisayarın insan gibi yanıt verip vermediğini belirlemek için kullanılan bir test olan Turing Testini sunar (Turing vd., 1950). Aldığı eleştirilere rağmen, Turing testi hala yapay zekâ tarihinin önemli bir parçası ve felsefi söylemde devam eden bir kavram olmaya devam etmektedir.

Stuart Russell ve Peter Norvig daha sonra, “Artificial Intelligence: A Modern Approach” adlı makaleyi yayımlamıştır, bu çalışma, yapay zekâ deneylerinde en sık kullanılan kaynaklardan birisi olagelmiştir (Russell ve Norvig, 2005). Makalede, yazarlar, her biri bilgisayar ortamlarının akla uygunluğa ve fikire karşı fiile dayanmış olacak şekilde nasıl çalıştığıyla ilgili olan dört olası yapay zekâ durumunu inceliyorlar (Russel vd., 2005).

Yapay zekâ kavramı incelerken makine öğrenmesi, derin öğrenme ve sinir ağları terimlerini de tanımlamak gerekmektedir. Derin öğrenme makine öğrenmesinin özelleşmiş halidir. Makine öğrenmesi ise yapay zekânın bir alt dalıdır (Brunette, Flemmer ve Flemmer, 2009). Yazının devam eden bölümlerinde bu kavramların ayrıntılı tanımlarını ve aralarındaki farkları açıklayacağız.

4.2 Makine Öğrenmesi

Makine öğrenimi, tıpkı insanların yaptığı gibi, zaman içinde öğrenme işlemini gerçekleştirerek hata payını en aza indirmek için elde edilmiş bilgi ve bilgisayar bilimleri tekniklerini kullanan bir tür yapay zekadır (Brunette vd., 2009).

Makine öğrenimi, gelişen yapay zekâ çalışma bölgesinin ehemmiyetli bir parçasıdır. Matematiksel temelli sistemlerin dahil edilmesi ile, algoritmasal teknikler sınıflandırmalar veya tahminler yapmak için eğitilir ve veri madenciliği projelerindeki temel bilgileri ortaya çıkarır (Brunette, vd., 2009). Bu iç görüler daha sonra uygulamalar ve işletmeler içinde karar vermeyi yönlendirerek ideal olarak temel büyüme ölçümlerini etkileyebilmektedir (Brunette vd., 2009).

Makine öğrenmesi kavramına ek olarak yakın çalışma alanlarından biri olan derin öğrenmeyi tanımlamak gereklidir. Yapay zekânın alt dalları derin öğrenme, makine öğrenimi ve sinir ağlarının tümüdür (LeCun, Bengio ve Hinton, 2015). Bununla birlikte, derin öğrenme, makine öğreniminin bir alt kümesidir ve sinir ağları, derin öğrenmenin bir alt kümesidir. Derin öğrenme ve makine öğrenimi arasındaki fark, her bir algoritmanın nasıl öğrendiğidir (LeCun vd., 2015). Derin öğrenme otomasyonu, gerekli insan müdahalesi miktarını azaltarak, verilerden özelliklerin çıkarılmasıyla ilgili işin çoğunu yapabilir. Bu, daha geniş veri öbeklerinin eğitilmesini mümkün kılar. Alışılmış makine öğrenimi, öğrenmek için daha fazla insan girdisine ihtiyaç duymaktadır (LeCun vd., 2015). Bilirkişiler, genellikle daha fazla yapılandırılmış verinin öğrenilmesini gerektiren veri girdileri arasındaki farkları anlamak için bir dizi özellik belirler (LeCun vd., 2015).

Makine öğrenimi derin olursa, algoritmayı çalıştırmak amacıyla işaretlenmiş veri kümelerinden faydalınabilinir, bununla birlikte her zaman işaretlenmiş bir veri kümesine ihtiyaç duyulmaz (LeCun vd., 2015). Düzenlenmemiş bilgileri işlenmemiş bir halde mesela resim veya metin gibi bir şekilde kabul edebilir ve değişik bilgi sınıflarını birbirinden ayırt eden özellikler kümesini kolay bir şekilde tayin edebilir. Derin öğrenmenin makine öğrenmesinden ayrılan yanı ise, bilgileri kullanılabilir hale getirmek için manuel bir işleme ihtiyaç duymaz, bu sayede makine öğrenimini birden farklı durumlarda genişletmemize imkân sağlar (LeCun vd., 2015).

Makine öğreniminde kullanılan algoritmalar çoğunlukla, yaklaşık bir değerlendirme veya oranlama yapmak ya da kategorize etmek için kullanılır (LeCun vd., 2015). Bu sistemler genellikle üç ana parçadan oluşmaktadır. İlk adım karar verme

sürecidir. Algoritmalar bilgilerdeki işaretlenebilen veya işaretlenemeyen girdilere bakarak, bir model hakkında bir sezgide bulunmaktadırlar (LeCun vd., 2015). İkinci adım bir hata fonksiyonunu devreye sokmaktır. Bir hata fonksiyonu, modelin tahminini değerlendirmeyi sağlar (LeCun vd., 2015). Bilinen örnekler varsa, modelin keskinliğini belirlemek amacıyla belirli bir hata fonksiyonu ile mukayese gerçekleştirilebilir. Son adım ise modeli en iyi durma getirme işi yani optimizasyon sürecidir. Eğitim için ayrılan bilgi noktaları modele daha güzel uyuyorsa var olan numune ile model değerlendirmesi arasındaki uyuşmazlığı gidermek amacıyla ağırlıklar yeniden düzenlenir (LeCun vd., 2015). Algoritma, keskinlik limitine varana değin model kat sayılarını bağımsız bir şekilde yenileyerek bu kıymetlendirme ve daha iyi hale getirme işini tekrarlamaktadır.

Makine öğrenmesi pek çok alanda kullanılmaktadır (LeCun vd., 2015). Bu alanlardan bir tanesi konuşma tanımadır. Konuşma tanımlamanın yapılması, konuşmadan metine tercüme şeklinde de tanınır ve kullanılan dili yazılı şekilde daha iyi hale getirmek amacıyla doğal dil işlemeyi benimseyen bir işlemdir (LeCun vd., 2015). Oldukça fazla mobil alet, sesli çağrı gerçekleştirmek veya mesajlaşma konusunda daha fazla erişilebilirlik sağlamak için sistemlerine konuşma tanıma özelliğini dahil etmektedirler (LeCun vd., 2015).

Makine öğrenmesi müşteri hizmetleri alanında da tercih edilmektedir (LeCun vd., 2015). Çevrimiçi sohbet robotları, insan arkadaşlığının almak için uğraşıyor. Sevkiyat tarzı durumlarla alakalı çokça ele alınan soruları cevaplar veya kişiye özgü öneriler veya müşteriler için farklı öneriler göstererek, sosyal medya ve diğer internet mecralarında kullanıcı etkileşimine yönelik fikirlerimizi farklılaştırırlar (LeCun vd., 2015). Örnekler arasında yapay aracılarla internet sitelerinde metin gönderme robotları, Twitter ve Instagram benzeri metin gönderme uygulamaları ve çoğunlukla yapay yardımcılar ve sesli yardımcılar ile birlikte gerçekleştirilen işler gösterilebilir (LeCun vd., 2015).

Bilgisayarlı sistemlerle görüntü teknolojileri makine öğrenmesi kullanarak yapılabilmektedir (LeCun vd., 2015). Böylelikle, makinelerin ve akıllı yapıların

bilgisayar resimlerinden, videolardan veya başka görsel gereçlerden faydalı veriler ortaya çıkarmasını sağlar ayrıca girdilere dayanarak harekete geçebilir (LeCun vd., 2015). Öneri temin etme kabiliyeti, bilgisayarlı görü sistemlerini görsel ayırt etme işlerinden farklılaştırır. Evrişimli sinir ağları aracılığıyla desteklenen bilgisayarlı görme, internet ortamında görsel işaretleme, sağlık ile ilgili görevlerde radyoloji görüntüleme veya motorlu taşıt alanında yalnız başına giden arabalarda uygulamalara sahiptir (LeCun vd., 2015).

Geçmiş tüketim davranışı verilerini kullanan yapay zekâ algoritmaları, daha etkili çapraz satış stratejileri geliştirmek için kullanılabilecek veri eğilimlerini keşfetmeye yardımcı olabilir. Bu, çevrimiçi perakendeciler için ödeme işlemi sırasında müşterilere ilgili eklenti önerileri yapmak için kullanılır (LeCun vd., 2015).

Otomatik hisse senedi ticareti yapmak içinde makine öğrenmesi yöntemlerinden faydalanılabilmektedir (LeCun vd., 2015). Hisse senedi portföylerini optimize etmek için tasarlanan yapay zekâ ile yönetilebilen çok hacimli alışveriş platformları kendi kendine günde çok fazla işlem yapabilir (LeCun vd., 2015).

4.2.1 Denetimli Öğrenme

Denetimli öğrenme, etiketli veri kümelerinin kullanımıyla tanımlanan bir makine öğrenimi yaklaşımıdır (LeCun vd., 2015). Bu veri kümeleri, bilgiyi kategorize etmek veya çıktıları hassasiyetle yaklaşık olarak değerlendirmek amacıyla algoritmaları hazırlamak veya denetlemek amacıyla tasarlanmıştır. Model, işaretlenmiş girdi ve çıktıları kullanarak doğruluğunu ölçebilir ve zamanla öğrenebilir. Denetimli öğrenme, veri madenciliği sırasında sınıflandırma ve regresyon olmak üzere iki tür probleme ayrılabilir (LeCun vd., 2015).

Sınıflandırma problemleri, test verilerini elmaları portakallardan ayırmak gibi belirli kategorilere doğru bir şekilde atamak için bir algoritma kullanır (LeCun vd., 2015). Veya gerçek dünyada, istenmeyen postaları gelen kutunuzdan ayrı bir klasörde sınıflandırmak için denetimli öğrenme algoritmaları kullanılabilir. Doğrusal

sınıflandırıcılar, destek vektör makineleri, karar ağaçları ve rastgele orman, tümü yaygın sınıflandırma algoritması türleridir (Wang, Ma ve Zhou, 2009).

Regresyon, bağımlı ve bağımsız değişkenler arasındaki ilişkiyi anlamak için bir algoritma kullanan başka bir denetimli öğrenme yöntemi türüdür (LeCun vd., 2015). Regresyon modelleri, belirli bir işletme için satış geliri projeksiyonları gibi farklı veri noktalarına dayalı sayısal değerleri tahmin etmek için yararlıdır. Bazı popüler regresyon algoritmaları lineer regresyon, lojistik regresyon ve polinom regresyondur (LeCun vd., 2015).

4.2.2 Denetimsiz Öğrenme

Denetimsiz öğrenme, işaretlenmemiş bilgi öbeklerini çözümlemek ve gruplamak için makine öğrenimesi yöntemlerinden yararlanır (LeCun vd., 2015). Bahsedilen yöntemler, kendi kendine verilerdeki saklı yapıları bulur dolayısıyla denetimsizdir. Denetimsiz öğrenme modelleri genel olarak üç ana görev için kullanılmaktadır. Bunlar kümeleme, ilişkilendirme ve boyut azaltma olarak sıralanabilir (LeCun vd., 2015).

Kümeleme (Clustering), etiketlenmemiş verileri benzerliklerine veya farklılıklarına göre gruplandırmaya yönelik bir veri madenciliği tekniğidir. Örneğin, K-ortalama kümeleme algoritmaları, benzer veri noktalarını gruplara atar; burada K değeri, gruplamanın boyutunu ve ayrıntı düzeyini temsil eder. Bu teknik, pazar bölümlendirme, görüntü sıkıştırma vb. için yararlıdır (LeCun vd., 2015).

İlişkilendirme (Association), belirli bir veri kümesindeki değişkenler arasındaki ilişkileri bulmak için farklı kurallar kullanan başka bir denetimsiz öğrenme yöntemi türüdür. Bu yöntemler, “Bu Ürünü Alan Müşteriler Ayrıca Bunu da Satın Aldı” önerileri doğrultusunda, pazar sepeti analizi ve öneri motorları için sıklıkla kullanılmaktadır (Wang vd., 2009).

Boyut azaltma (Dimensionality reduction), belirli bir veri kümesindeki özelliklerin veya boyutların sayısı çok yüksek olduğunda kullanılan bir öğrenme tekniğidir. Veri bütünlüğünü korurken, veri girişi sayısını yönetilebilir bir boyuta düşürür. Genellikle

bu teknik, örneğin otomatik kodlayıcıların resim kalitesini iyileştirmek için görsel verilerden paraziti çıkarması gibi veri ön işleme aşamasında kullanılmaktadır (LeCun vd., 2015).

İki yaklaşım arasındaki temel ayrım, etiketli veri kümelerinin kullanılmasıdır. Basitçe söylemek gerekirse, denetimli öğrenme etiketli girdi ve çıktı verilerini kullanırken denetimsiz öğrenme algoritması kullanmaz (LeCun vd., 2015).

Denetimli öğrenmede, algoritma, veriler üzerinde yinelemeli olarak tahminler yaparak ve doğru yanıtı ayarlayarak eğitim veri kümesinden öğrenir. Denetimli öğrenme modelleri denetimsiz öğrenme modellerinden daha doğru olma eğilimindeyken, verileri uygun şekilde etiketlemek için önceden insan müdahalesi gerektirirler. Örneğin, denetimli bir öğrenme modeli, günün saatine, hava koşullarına vb. bağlı olarak işe gidip gelme sürenizin ne kadar süreceğini tahmin edebilir. Ama önce, yağmurlu havanın sürüş süresini uzattığını bilmek için onu eğitmeniz gerekmektedir (LeCun vd., 2015).

Bunun aksine denetimsiz öğrenme modelleri, etiketlenmemiş verilerin doğal yapısını keşfetmek için kendi başlarına çalışmaktadırlar. Çıktı değişkenlerini doğrulamak için hala bazı insan müdahalesine ihtiyaç duyabilmektedir. Örneğin, denetimsiz bir öğrenme modeli, çevrimiçi alışveriş yapanların genellikle aynı anda ürün grupları satın aldığını belirleyebilmektedirler. Bununla birlikte, bir veri analistinin, bir öneri motorunun bebek kıyafetlerini bir sıra bebek bezi, elma püresi ve geniş saplı bardaklarla gruplandırmasının mantıklı olduğunu doğrulaması gerekmektedir (LeCun vd., 2015).

Denetimli öğrenmede amaç, yeni veriler için sonuçları tahmin etmektir. Beklenecek sonuçların türünü önceden bilinmektedir. Denetimsiz bir öğrenme algoritmasıyla amaç, büyük hacimli yeni verilerden iç görüler elde etmektir. Makine öğreniminin kendisi, veri kümesinden neyin farklı veya ilginç olduğunu belirlemektedir (LeCun vd., 2015).

Denetimli öğrenme modelleri, diğer şeylerin yanı sıra istenmeyen posta algılama, duygu analizi, hava durumu tahmini ve fiyat tahminleri için idealdir. Buna karşılık, denetimsiz öğrenme, anormallik tespiti, öneri motorları, müşteri kişilikleri ve tıbbi görüntüleme için çok uygundur (LeCun vd., 2015).

Denetimli öğrenme, makine öğrenimi için tipik olarak R veya Python gibi programlama dillerinin kullanımıyla hesaplanan basit bir yöntemdir. Denetimsiz öğrenmede, büyük miktarda sınıflandırılmamış veriyle çalışmak için güçlü araçlara ihtiyaç olmaktadır. Denetimsiz öğrenme modelleri, amaçlanan sonuçları üretmek için büyük bir eğitim setine ihtiyaç duydukları için hesaplama açısından karmaşıktır (LeCun vd., 2015).

Denetimli öğrenme modellerinin eğitilmesi zaman alabilir ve girdi ve çıktı değişkenleri için etiketler uzmanlık gerektirir. Ayrıca, çıktı değişkenlerini doğrulamak için insan müdahalesi olmadıkça, denetimsiz öğrenme yöntemleri bazen yanlış sonuçlara sahip olabilmektedir (LeCun vd., 2015).

Büyük verileri sınıflandırmak, denetimli öğrenmede gerçek bir zorluk olabilir, ancak sonuçlar son derece doğru ve güvenilirdir. Buna karşılık, denetimsiz öğrenme, büyük hacimli verileri gerçek zamanlı olarak işleyebilir. Ancak, verilerin nasıl kümelendiğine dair şeffaflık eksikliği ve daha yüksek hatalı sonuç riski bulunmaktadır. Yarı denetimli öğrenme bu şekilde kullanılmaktadır (LeCun vd., 2015).

4.2.3 Yarı Denetimli Öğrenme

Yarı denetimli öğrenme hem etiketli hem de etiketsiz veriler içeren bir eğitim veri kümesi kullanılan makine öğrenmesi sitemleridir. Verilerden ilgili özellikleri çıkarmanın zor olduğu ve yüksek hacimli veriler olduğunda özellikle yararlıdır.

Yarı denetimli öğrenme, az miktarda eğitim verisinin doğrulukta önemli bir iyileşmeye yol açabileceği tıbbi görüntüler için idealdir. Örneğin, bir radyolog, tümörler veya hastalıklar için küçük bir taraması işlemini etiketleyebilir, böylece

makine, hangi hastaların daha fazla tıbbi müdahaleye ihtiyaç duyabileceğini daha doğru bir şekilde tahmin edebilir (LeCun vd., 2015).

4.2.4 Takviyeli Öğrenme

Takviyeli makine öğrenmesi yöntemi, denetimli öğrenmeyi benzeyen şekilde davranışsal bir modeldir. Bu model yapısı, tecrübe etme ile hazırlanır. Takviyeli öğrenme yönteminde, öğrenmeyi gerçekleştirmek için bir ajan oluşturur. Ajan, belirsiz, potansiyel olarak karmaşık bir ortamda bir hedefe ulaşmayı öğrenir. Takviyeli öğrenmede, bir yapay zekâ oyun benzeri bir durumla karşı karşıyadır. Bilgisayar, soruna bir çözüm bulmak için deneme yanılma yöntemini kullanır. Makinenin programcının istediğini yapmasını sağlamak için yapay zekâ, gerçekleştirdiği eylemler için ya ödül ya da ceza alır. Amacı toplam ödülü maksimize etmektir (LeCun vd., 2015).

Tasarımcı, ödül politikasını yani oyunun kurallarını belirlemesine rağmen, modele oyunun nasıl çözüleceğine dair hiçbir ipucu veya öneri vermez. Tamamen rastgele denemelerden başlayıp karmaşık taktikler ve insanüstü becerilerle bitirerek ödülü en üst düzeye çıkarmak için görevin nasıl gerçekleştirileceğini bulmak modele bağlıdır. Aramanın ve birçok denemenin gücünden yararlanarak pekiştirmeli öğrenme şu anda makinenin yaratıcılığının en etkili göstergesidir. Yeterince güçlü bir bilgisayar altyapısı üzerinde bir pekiştirmeli öğrenme algoritması çalıştırılırsa insanların aksine yapay zekâ binlerce paralel oyundan deneyim toplayabilir (LeCun vd., 2015).

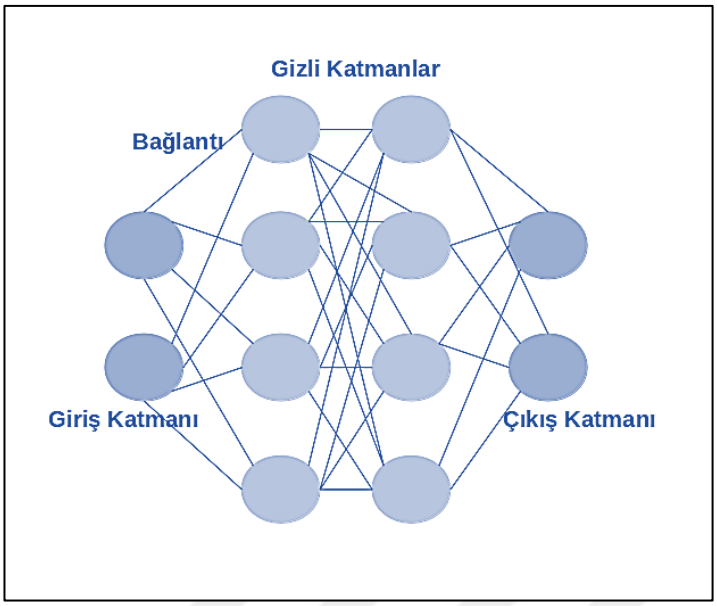
Takviyeli öğrenmedeki ana zorluk, gerçekleştirilecek göreve büyük ölçüde bağlı olan simülasyon ortamının hazırlanmasında yatmaktadır. Satranç, Go veya Atari oyunlarında modelin insanüstü olması gerektiğinde simülasyon ortamını hazırlamak nispeten basittir. Otonom bir araba sürebilen bir model oluşturmaya gelince, arabayı sokakta sürmeden önce gerçekçi bir simülatör inşa etmek çok önemlidir. Modelin, bin arabayı bile feda etmenin minimum maliyetle gerçekleştiği güvenli bir ortamda nasıl fren yapılacağını veya bir çarpışmadan nasıl kaçınılacağını bulması gerekmektedir. LeCun vd. (2015) modeli eğitim ortamından gerçek dünyaya aktarmanın işlerin zorlaştığı yer olarak tanımlamışlardır.

Aracıyı kontrol eden sinir ağını ölçeklemek ve ayarlamak başka bir zorluktur. Ödüller ve cezalar sistemi dışında ağ ile iletişim kurmanın bir yolu yoktur. Bu, özellikle yeni bilgi edinmenin eski bilgilerin bir kısmının ağdan silinmesine neden olduğu bir unutmaya yol açabilir. Ajan için başka bir zorluk ise yerel bir optimuma ulaşma görevidir. Son olarak tasarlandığı görevi gerçekleştirmeden ödülü optimize edecek ajanlar oluşabilmektedir. Temsilcinin ödül kazanmayı öğrendiği ancak yarışı tamamlamayı öğrenmediği durumlar oluşabilmektedir (LeCun vd., 2015).

4.3 Yapay Sinir Ağları

İnsan davranışlarını kopyalayarak makine komutlarının yapay zekâ ve alt dallarında ki kalıpları tanımasına yardım eden ve istenilen hataları gidermesini sağlayan sistemlere yapay sinir ağları denir. Yapay sinir ağları, makine öğrenmesi alanının bir alt grubudur. İsimleri ve ögeleri, biyolojik sinir sisteminin karşılıklı sinyal alışveriş yapısını kopyalayarak insanların sisteminden ilham almıştır (Abiodun vd., 2018).

Yapay sinir ağları, belirli olan tek girdi katmanı, tek veya daha çok saklı katman ve bir sonuç katmanı ilave edilmiş katmanlardan meydana gelir. Her boğum bir ötekine sinyal gönderir ve alakalı tek ağırlık ve limite sahiptir. Rastgele tek boğumun sonucu belli bir limit sayısının üstündeyse böyle boğumlar etkinleştirilir ve yapının bir ileriki katmanına bilgi postalanır. Tersi olursa, yapının bir ileriki katmanına herhangi bir bilgi geçişi olmaz. Burada ki çoklu sinir ağı katmanı yalnızca bir sinir ağı yapısında ki katmanların yoğunluğunu işaret eder (Abiodun vd., 2018). Girdileri ve çıktıyı içerecek şekilde üçten fazla katmandan oluşan bir sinir ağı yapısı, tek derin öğrenme yöntemi veya daha derin bir sinir ağı olarak kabul edilebilir. Dörtten az katmana sahip olan tek sinir ağı, sadece ilk seviyede bir sinir ağı olarak tanımlanmaktadır. Şekil 4.1’de sinir ağlarının yapısı gösterilmiştir.



Şekil 4.1 Sinir ağları yapısının gösterimi

Sinir ağları gelişerek eğitilmek ve hassasiyeti artırmak amacıyla eğitim verilerine dayandırılır. Ayrıca, bu öğrenme yöntemleri hassasiyet elde etmek amacıyla ince ayar çekildiğinde, bilgisayar bilimi ve alt dallarında gayet etkili vasıtalardır ve bilgileri aşırı hızlı şekilde kategorize edilmesine ve gruplanmasına imkân sağlar (Abiodun vd., 2018). Konuşma belirleme veya görsel belirlemedeki işler bilir kişiler aracılığıyla gerçekleştirilen el ile tanımlama ile mukayese edildiğinde çok kısa sürebilmektedir (Abiodun vd., 2018).

.

Tüm boğumlar girdi bilgileri, model kat sayıları, bir eşik ve bir sonuçtan meydana gelen lineer regresyon yöntemi şeklinde ele alınmaktadır. Denklem 4.1 ve 4.2'de sinirler üzerindeki girdi, ağırlıklar ve çıktıların işlemleri gösterilmiştir.

$$\sum_{i=1}^{m} w_i x_i + yanlılık = w_1 x_1 + w_2 x_2 + w_3 x_3 + y \tag{4.1}$$

$$çıktı = f(x) = \begin{cases} 1, \sum w_1 x_1 + b \geq 0 \ olduğunda \\ 0, \qquad \sum w_1 x_1 + b < 0, d.d. \end{cases} \tag{4.2}$$

Tek girdi tabakası belirlendikten sonra model kat sayıları atanmaktadır. Belirlenen kat sayılar rastgele bir değişkenin ehemmiyetini belirlemeye fayda sağlamaktadır ve

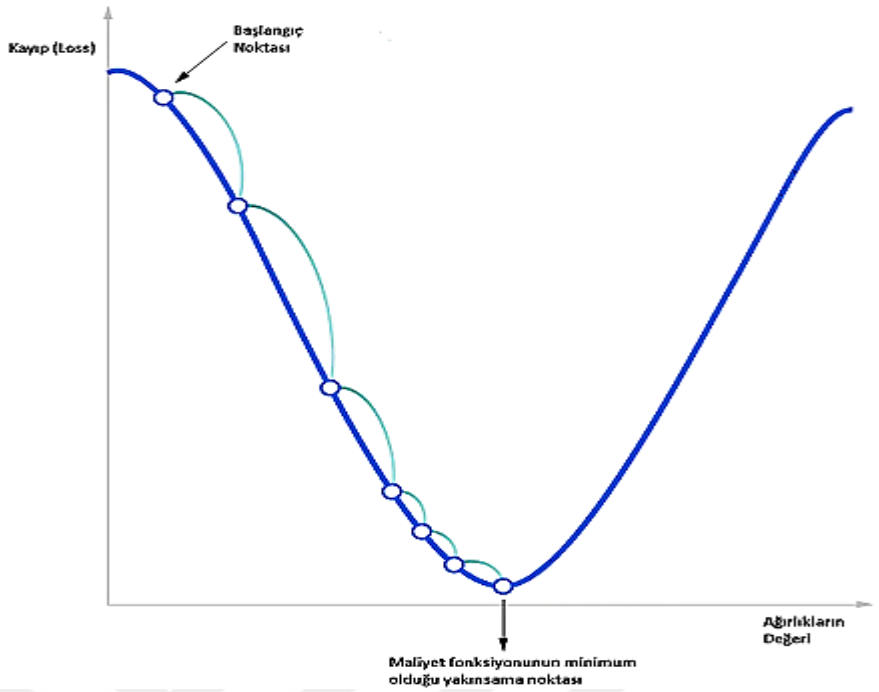
fazla olanlar öteki girdilere göre sonuçlara daha çok yardımcı olmaktadır veya katkı sağlamaktadır. Bütün girdiler sonrasında alakalı kat sayılarla çarpılır ve toplanır. Bundan sonra bu sonuç, bir sonraki sonucu tayin eden bir etkinleştirme fonksiyonuna sokulur. Elde edilen sonuç muayyen bir limiti geçerse, boğumu etkinleştirir ve verileri ağdaki bir sonraki katmana iletir. Böylece düğümün sonucunun bir ileriki boğumun girdisi olarak geçmesine sebep olur. Bilgileri bir tabakadan ileriki tabakaya ileterek işleyen ağlar ileri yönde beslemeli ağ adını almaktadır. (LeCun vd., 2015).

Sinir ağları verileri tek düğümden ötekine basamaklandıran karar ağaçlarına benzer şekilde davranmaktadır. 0 ile 1 aralığında x sayılarını elde etmek, tek bir parametrede ki herhangi bir belirli değişikliğin rasgele bir boğumun sonucu üstündeki katkısını ve bir ileriki sonucu üstündeki faydasını azaltacaktır (LeCun vd., 2015).

Model oluşturulurken bir maliyet fonksiyonu ile doğruluğunu analiz etmek gerekmektedir. Bu yöntem ortalama kare hatası (Mean Squared Error) olarak adlandırılır. Maliyet fonksiyonu grafiği Şekil 4.2'de gösterilmiştir. Denklem 4.3'de, i örneğin indeksini temsil eder, $\hat{y}$ tahmin edilen sonuçtur, y gerçek değerdir ve m ise örnek sayısıdır.

$$Maliyet\ fonksiyonu = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2 \tag{4.3}$$

Amaç, rastgele bir inceleme amacıyla hassasiyeti temin etmek için hata fonksiyonunu indirgemektir. Algoritma kat sayıları ve önyargısını ayarlarken hata fonksiyonunu ve pekiştirmeli öğrenmeyi kullanarak lokal en aza ulaşır. Modelin kat sayılarını düzenlediği işlem gradyan iniş yoluyladır ve hata fonksiyonunu en aza indirgemek amacıyla yönü tayin etmesine müsade edilir. Modelin parametreleri asgari aşamalı şekilde yakınsamak amacıyla düzenlenir (LeCun vd., 2015).



Şekil 4.2 Maliyet Fonksiyonu Grafiği (Gradient Descent)

Sinir ağı modelini geri difüzyon etkisiyle de eğitilebilir yani sonuçtan girişe geri şekilde ilerler. Geri difüzyon her bir nöronla alakalı yanlışı hesaplamaya ve ilişkilendirmeye müsaade ederek modellerin parametrelerini optimize etmeye ve uydurmaya çalışmaktadır. Sinir ağları modellerinin parametrelerini optimize etmek için en çok kullanılan algoritmalardan birisi de "Gradient Descent" algoritmasıdır (LeCun vd., 2015).

Gradient descent makine öğrenimi modellerini ve sinir ağlarını eğitmek için yaygın olarak kullanılan bir optimizasyon algoritmasıdır. Eğitim verileri bu modellerin zaman içinde öğrenmesine yardımcı olur ve bu algoritma içindeki maliyet fonksiyonu parametre güncellemelerinin her yinelemesinde doğruluğunu ölçerek özellikle bir ayarlayıcı görevi görür. Fonksiyon sıfıra yakın veya sıfıra eşit olana kadar, model parametrelerini mümkün olan en küçük hatayı verecek şekilde ayarlamaya devam edecektir. Makine öğrenimi modelleri doğruluk için optimize edildiğinde, yapay zekâ ve bilgisayar bilimi uygulamaları için güçlü araçlar olabilirler (LeCun vd., 2015).

Gradient descent algoritması için bir yön ve bir öğrenme oranı (learning rate) olmak üzere iki veri noktasına ihtiyaç vardır. Bu faktörler gelecekteki yinelemelerin kısmi türev hesaplamalarını belirleyerek yerel veya küresel minimuma yani yakınsama noktasına kademeli olarak ulaşmasını sağlar (Abiodun vd., 2018).

Öğrenme hızı adım boyutu veya alfa olarak da adlandırılır ve minimuma ulaşmak için atılan adımların boyutudur. Bu genellikle küçük bir değerdir ve maliyet fonksiyonunun davranışına göre değerlendirilir ve güncellenir. Yüksek öğrenme oranları daha büyük adımlarla sonuçlanır, ancak minimumu aşma riskleri vardır. Tersine düşük bir öğrenme oranı küçük adım boyutlarına sahiptir. Daha fazla hassasiyet avantajına sahip olsa da minimuma ulaşmak için daha fazla zaman ve hesaplama gerektirdiğinden yineleme sayısı genel verimliliği tehlikeye atmaktadır (LeCun vd., 2015).

4.3.1 Aktivasyon Fonksiyonu

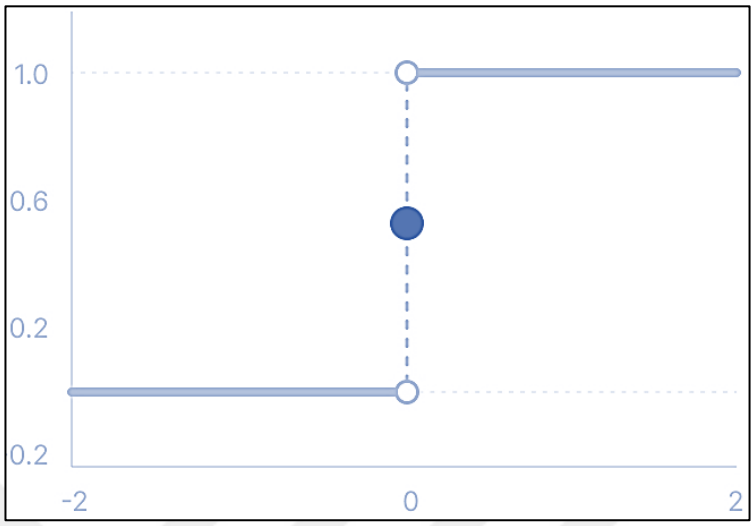
Bir Aktivasyon Fonksiyonu, bir nöronun aktive edilip edilmeyeceğine karar verir. Bu, nöronun ağa girişinin önemli olup olmadığına daha basit matematiksel işlemler kullanarak tahmin sürecinde karar vereceği anlamına gelir. Aktivasyon Fonksiyonunun rolü bir düğüme veya bir katmana beslenen bir dizi girdi değerinden çıktı elde etmektir.

Tüm gizli katmanlar genellikle aynı aktivasyon fonksiyonunu kullanır. Ancak, çıktı katmanı tipik olarak gizli katmanlardan farklı bir aktivasyon fonksiyonu kullanır. Seçim, model tarafından yapılan tahminin amacına veya türüne bağlıdır. İleri beslemeli yayılımda aktivasyon fonksiyonu mevcut nöronu besleyen girdi ile bir sonraki katmana giden çıktısı arasında matematiksel bir kapı oluşturur.

4.3.1.1 İkili Adım Fonksiyonu

İkili adım fonksiyonu, bir nöronun etkinleştirilip etkinleştirilmemesi gerektiğine karar veren bir eşik değerine bağlıdır. Aktivasyon fonksiyonuna beslenen girdi, belirli bir eşik ile karşılaştırılır; eğer girdi ondan büyükse, o zaman nöron aktive olur, aksi

takdirde deaktive edilir, yani çıktısı bir sonraki gizli katmana iletilmez. Şekil 4.3'de eşik değeri 0 olan bir ikili adım fonksiyonu gösterilmiştir.



Şekil 4.3 İkili Adım Fonksiyonu

İkili adım fonksiyonunun matematiksel gösterimi denklem 4.4'te verilmiştir. İkili adım fonksiyonunun bazı sınırlamaları vardır. Örneğin çok değerli çıktılar sağlayamaz, çok sınıflı sınıflandırma problemlerinde kullanılamaz. Adım fonksiyonunun gradyanı sıfırdır, bu da geri yayılım sürecinde bir engele neden olur.

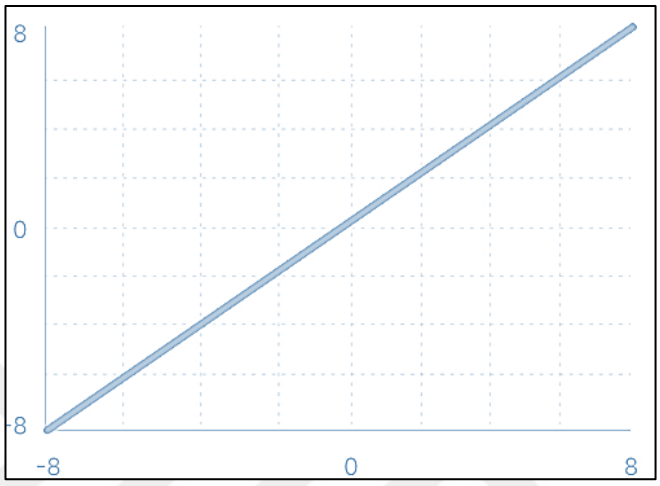
$$f(x) = \begin{cases} 0, & x < 0 \; için \\ 1, & x \geq 0 \; için \end{cases} \tag{4.4}$$

4.3.2 Doğrusal Aktivasyon Fonksiyonu

Doğrusal aktivasyon fonksiyonu verilen girdi ile doğru orantılı olarak çıktı verir. Bir başka değişle giriş değerini değiştirmeden aynen yansıtır. Bununla birlikte, bir lineer aktivasyon fonksiyonunun iki ana problemi vardır. Fonksiyonun türevi sabit olduğundan ve x girişiyle hiçbir ilişkisi olmadığından geri yayılımı kullanmak mümkün değildir. Matematiksel gösterimi denklem 4.5'de verilmiştir.

$$f(x) = x \tag{4.5}$$

Doğrusal bir etkinleştirme işlevi kullanılırsa sinir ağının tüm katmanları tek bir katmana dönüşecektir. Sinir ağındaki katman sayısı ne olursa olsun son katman yine de ilk katmanın doğrusal bir işlevi olacaktır. Yani esasen doğrusal bir aktivasyon işlevi sinir ağını sadece bir katmana dönüştürür. Doğrusal aktivasyon fonksiyonu grafiği Şekil 4.4'de gösterilmiştir.



Şekil 4.4 Doğrusal Aktivasyon Fonksiyonu

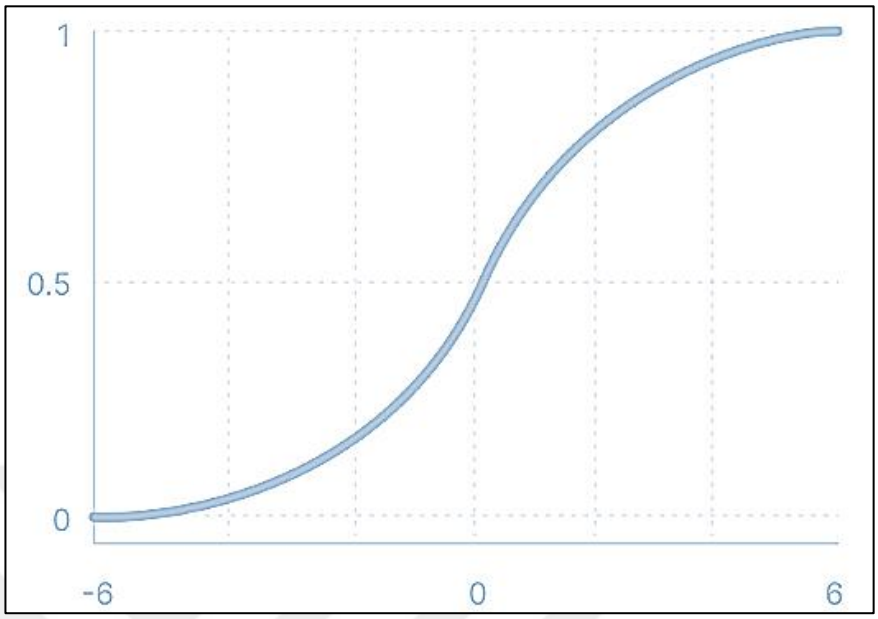
4.3.3 Sigmoid Aktivasyon Fonksiyonu

Sigmoid aktivasyon fonksiyonu herhangi bir gerçek değeri girdi olarak alır ve 0 ile 1 aralığında bir değer döndürür. Girdi ne kadar büyükse (daha pozitifse), çıktı değeri 1'e o kadar yakın olur veya girdi ne kadar küçükse (daha negatifse) o kadar yakın olursa çıktı değeri 0'a o kadar yakın olacaktır. Fonksiyonun matematiksel gösterimi denklem 4.6'da gösterilmiştir.

$$f(x) = \frac{1}{1+e^{-x}} \tag{4.6}$$

Sigmoid aktivasyon fonksiyonu en yaygın kullanılan fonksiyonlardan biridir. Çıktı olarak olasılığı tahmin etmemiz gereken modeller için yaygın olarak kullanılır. Herhangi bir şeyin olasılığı yalnızca 0 ile 1 aralığında olduğundan, aralığı nedeniyle sigmoid doğru seçimdir. Fonksiyon türevlenebilirdir ve düzgün bir gradyan sağlar, yani çıktı değerlerinde sıçramaları önler. Bu durum, sigmoid aktivasyon

fonksiyonunun bir S şekli ile temsil edilir. Ayrıca Sigmoid aktivasyon fonksiyonun grafiği Şekil 4.5'de gösterilmiştir.



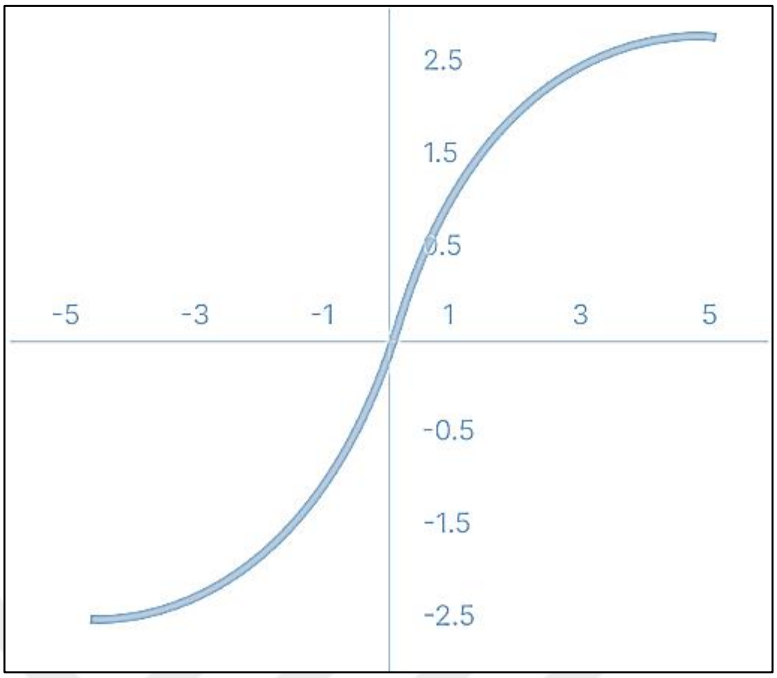
Şekil 4.5 Sigmoid aktivasyon fonksiyonu

4.3.4 Tanh Aktivasyon Fonksiyonu

Tanh fonksiyonu sigmoid aktivasyon fonksiyonuna çok benzemektedir. Fonksiyon 1 ile -1 çıkış aralığına yayılmıştır. Sigmoid fonksiyonu gibi S şekline sahiptir. Tanh'da giriş ne kadar büyükse (daha pozitifse), çıkış değeri 1'e o kadar yakın olacaktır, girdi ne kadar küçükse (daha negatifse), çıktı -1'e o kadar yakın olur. Tanh aktivasyon fonksiyonunun matematiksel gösterimi denklem 4.7'de verilmiştir.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{4.7}$$

Bu aktivasyon fonksiyonunu kullanmanın bazı avantajları vardır. Tanh aktivasyon fonksiyonunun çıktısı sıfır merkezlidir. Dolayısıyla çıktı değerlerini kuvvetle negatif, nötr veya kuvvetle pozitif olarak kolayca eşleyebiliriz. Değerleri -1 ile 1 arasında olduğu için genellikle bir sinir ağının gizli katmanlarında kullanılır. Bu nedenle gizli katmanın ortalaması 0 veya ona çok yakın çıkar. Verileri merkezileştirmeye yardımcı olur ve bir sonraki katman için öğrenmeyi çok daha kolay hale getirir. Tanh aktivasyon fonksiyonun grafiği Şekil 4.6'da gösterilmiştir.



Şekil 4.6 Tanh aktivasyon fonksiyonu

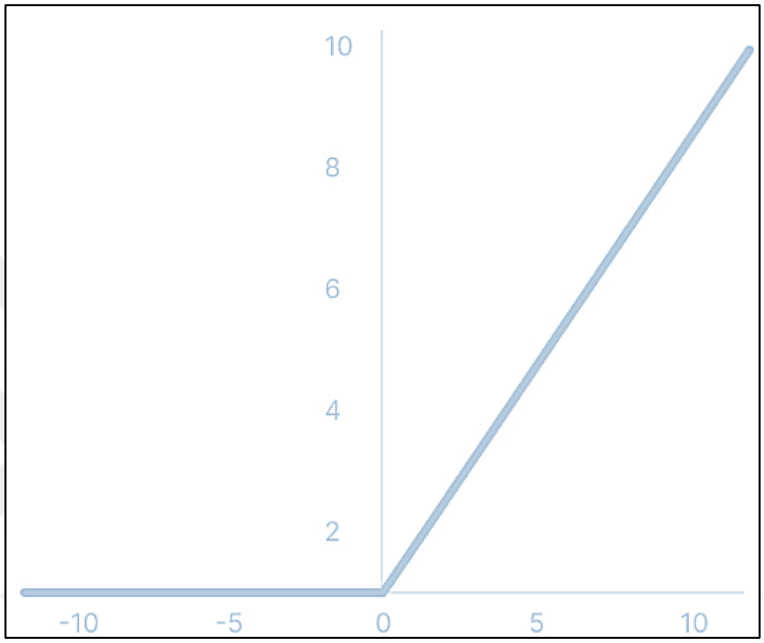
4.3.5 Doğrultulmuş Lineer Birim Aktivasyon Fonksiyonu

Doğrultulmuş Lineer Birim Fonksiyonu (Rectified Lineer Unit Function-ReLU) doğrusal bir işlev izlenimi vermesine rağmen, ReLU bir türev işlevine sahiptir ve aynı anda onu hesaplama açısından verimli hale getirirken geri yayılıma izin verir. Buradaki ana nokta, ReLU işlevinin tüm nöronları aynı anda etkinleştirmemesidir. Nöronlar doğrusal dönüşümün çıktısı 0'dan küçükse devre dışı bırakılır. Doğrultulmuş lineer birim aktivasyon fonksiyonunun matematiksel gösterimi denklem 4.8'de gösterilmiştir.

$$f(x) = \max(0, x) \tag{4.8}$$

Aktivasyon fonksiyonu olarak ReLU kullanmanın avantajları vardır. Yalnızca belirli sayıda nöron etkinleştirildiğinden, ReLU fonksiyonu sigmoid ve tanh fonksiyonlarına kıyasla hesaplama açısından çok daha verimlidir. ReLU, lineer, doygun olmayan özelliği nedeniyle, kayıp fonksiyonunun global minimumuna doğru gradyan inişinin yakınsamasını hızlandırır.

Doğrultulmuş lineer birim aktivasyon fonksiyonunun grafiği Şekil 4.7'de gösterilmiştir. Doğrultulmuş lineer birim aktivasyon fonksiyon grafiğinin negatif tarafı, gradyan değerini sıfır yapar. Bu nedenle geri yayılım sürecinde bazı nöronların ağırlıkları ve yanlılıkları güncellenmez. Bu, asla etkinleştirilmeyen ölü nöronlar yaratabilir. Tüm negatif girdi değerleri hemen sıfır olur, bu da modelin verilere uygun şekilde uyma veya onlardan eğitim alma yeteneğini azaltabilmektedir.

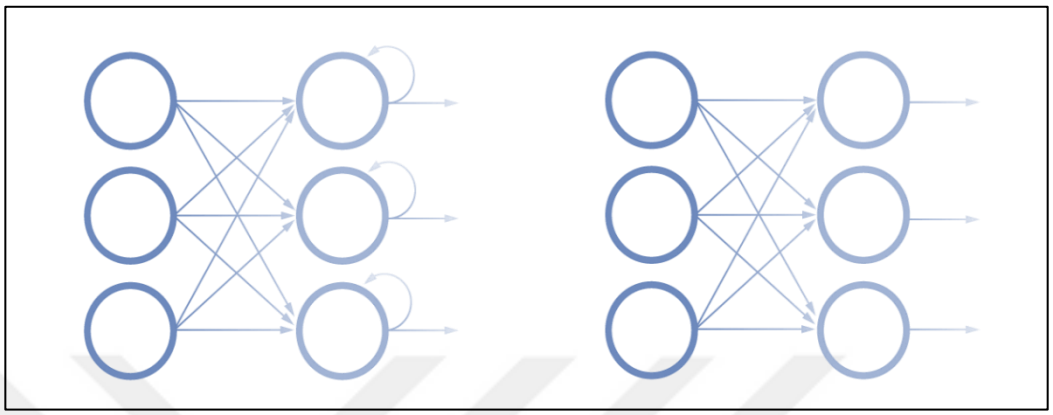


Şekil 4.7 Doğrultulmuş lineer birim aktivasyon fonksiyonu

4.4 Tekrarlayan Sinir Ağları (RNN)

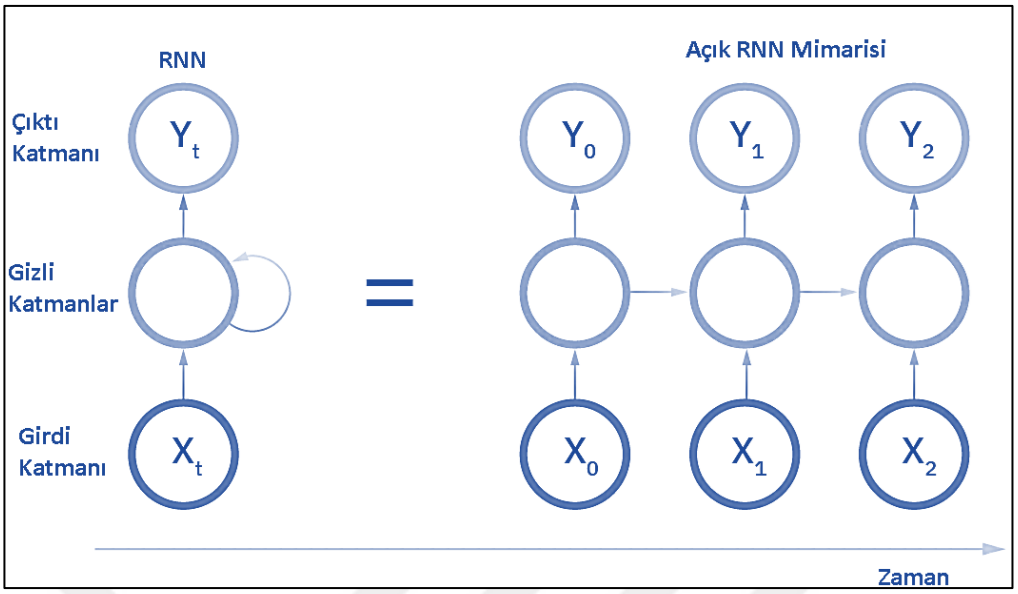
Yineleyen nöron ağları (RNN), ardışık dizileri veya zaman dizileri bilgilerini işleyen bir çeşit nöron ağıdır (Yin, vd., 2017). Dil tercümesi, doğal dil işleme, konuşma dili tanımlama ve görsel yazıları benzeri dizili veya vakitsel sorunlar ile yaygın olarak kullanılabilmektedir. Google Çeviri gibi popüler uygulamalarda da kullanılabilmektedirler. İleri beslemeli ve evrişimli sinir ağları gibi, tekrarlayan (yenileyen) sinir ağları da öğrenmek için eğitim verilerini kullanır. Var olan girdi ve sonucu değiştirmek amacıyla önceki girdilerden veri elde ettikleri için "hafızaları" ile ayrışırlar. Süregelen sinir ağları, girdilerin ve çıktıların birbirinden tamamen alakasızlığını kabul ederken, tekrarlayan sinir ağlarının sonucu, sıralı dizi içerisinde ki önceki unsurlarla sınırlıdır (Yin vd., 2017). Gelecekteki olaylar, belirli bir dizinin

çıktısının belirlenmesinde de yardımcı olurken, tek yönlü tekrarlayan sinir ağları, bu olayları tahminlerinde açıklayamaz. Şekil 4.8'de solda tekrarlayan sinir ağlarının ve sağda ileri beslemeli sinir ağlarının yapılarının karşılaştırılması gösterilmiştir (LeCun vd., 2015).



Şekil 4.8 Tekrarlayan sinir ağı ve ileri beslemeli sinir ağı karşılaştırması

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. Şekil 4.9'da örnek RNN mimarisi gösterilmiştir (LeCun vd., 2015).



Şekil 4.9 RNN mimarisi

4.5 Çift Yönlü Tekrarlayan Sinir Ağları

Çift Yönlü Tekrarlayan Sinir Ağları, gizliden gizliye bağlantıların zıt zamansal düzende aktığı ikinci bir gizli katman sunarak tek yönlü RNN'yi genişletir (Berglund vd., 2015). Bu nedenle model hem geçmişten hem de gelecekten gelen bilgileri kullanabilir (Berglund vd., 2015; Schuster ve Paliwal, 1997).

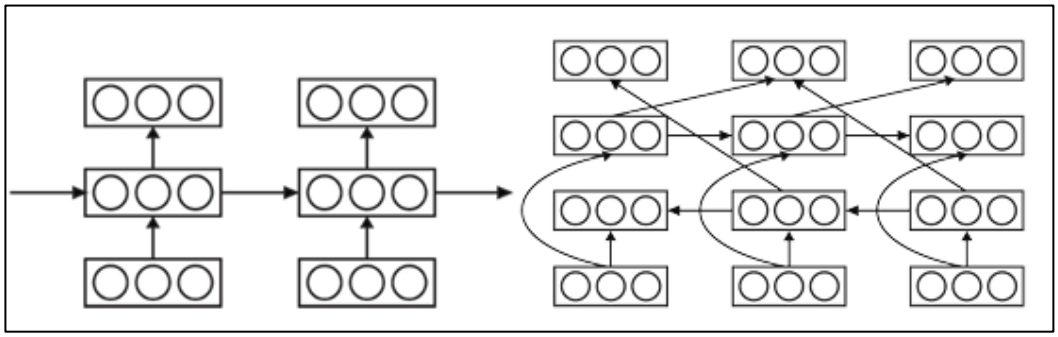
$$P(y_t|\{x_d\}_{d\neq t}) = \phi(W_y^f h_{t-1}^f + W_y^b h_{t+1}^b + b_y) \quad (4.9)$$

$$h_t^f = \tanh(W_y^f h_{t-1}^f + W_x^f x_t + b_h^f) \quad (4.10)$$

$$h_t^b = \tanh(W_y^b h_{t+1}^b + W_x^b x_t + b_h^b) \quad (4.11)$$

Şekil 4.10'da sol tarafta tek yönlü yinelenen sinir ağları ve sağ tarafta ise çift yönlü yinelenen sinir ağlarının yapısı gösterilmektedir. Normal RNN ile karşılaştırıldığında ileri ve geri yönleri ayrı ağırlıklara ve gizli aktivasyonlara sahiptir. Sırasıyla bu yönler için için üst simge i ve g harfleri ile gösterilir. Bağlantıların asiklik olduğuna dikkat edilmelidir (Berglund vd., 2015). Denklemler 4.9, 4.10 ve 4.11'de çift taraflı yinelenen sinir ağlarının fonksiyonel yapısı gösterilmiştir (Berglund vd., 2015). Ayrıca önerilen formülasyonda y_t'nin x_t'den bilgi almadığına dikkat edilmelidir (Berglund vd., 2015).

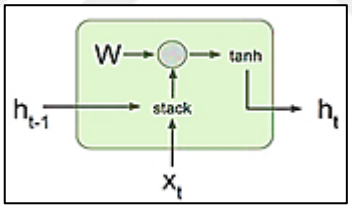
Bu nedenle, girdi dizisindeki diğer tüm zaman adımlarını basitçe Y = X ayarlayarak verilen bir zaman adımını tahmin etmek için modeli denetimsiz bir şekilde kullanabilmektedir (Schuster vd., 1997).



Şekil 4.10 Tek yönlü RNN ve çift yönlü RNN yapısı (Berglund vd., 2015)

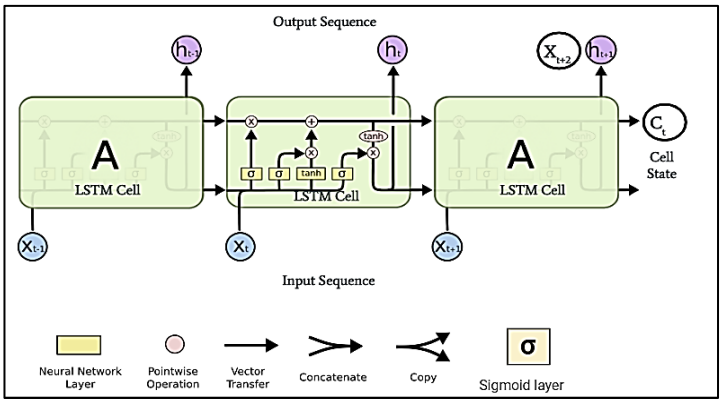
4.6 Uzun Kısa Vadeli Bellek (LSTM)

Tekrarlayan sinir ağlarındaki (RNN) tüm çıktılar kendinden önceki elemanların çıktılarına bağlı olarak değişmekte veya meydana gelmektedir. Fakat kelimeler arası mesafe arttıkça RNN'in önceki verileri kullanması zorlaşır.



Şekil 4.11 Yinelenebilir sinir ağı yapısı

Şekil 4.11'de, Xt girdi, h(t-1) bir önce ki adımda üretilen gizli durum (hidden state), W ağırlık matrisi ve tanh ise ağrılık fonksiyonunu oluşturmaktadır. RNN yapısı, "kayıp gradyan" (vanishing gradient) ve "taşan gradyan" (exploding graident) problemleri yüzünden, yani oluşan çıktı değerinin çok küçük veya çok büyük çıkması yüzünden, günümüzde çok tercih edilmemektedir. RNN'in bir alt yapısı olan LSTM ise sağladığı avantajlardan ötürü daha çok kullanılmaktadır.



Şekil 4.12 Uzun kısa vadeli hafıza ağları yapısı

Şekil 4.12'de görüleceği üzere birleşen oklar vektörlerin bir araya gelmesini, okların ayrılması ise kopyalanarak iki vektörün meydana gelmesini göstermektedir. Ayrıca, LSTM yapısında duruma göre birden farklı aktivasyon fonksiyonu da kullanılabilmektedir. Bunlara ek olarak hücre durumu (cell state), LSTM için çok önemli bir elemandır. Hücre durumu, bir hücreden diğer hücreye veri geçişini düzgün bir biçimde gerçekleştirmektedir. Ayrıca, hücre durumu güncellenerek optimize edebilmektedir.

LSTM kendini güncel tutmak için bazı geçitlere sahiptir. Bunlar, unutma kapısı (forget gate), girdi kapısı (input gate) ve çıktı kapısı (output gate) kapılarıdır.

$$f_t = \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right) \tag{4.12}$$

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \tag{4.13}$$

$$\acute{C}_t = tanh(W_c . [h_{t-1}, x_t] + b_c) \tag{4.14}$$

$$C_t = f_t * C_{t-1} + i_t * \acute{C}_t \tag{4.15}$$

Unutma kapısı, kendisine gelen verilerin hangisini unutacağına sigmoid fonksiyonu ile denklemde gösterildiği gibi karar verir. Çıkan değerler 0'a yakınsa unutmaya yakın davranır. Değerler 1'e yakınsa hiç değişiklik yapmaz ya da yakınlık oranına uygun

olarak az değiştirir. Girdi kapısı, genellikle hangi bilgilerin sonradan kullanılacağına karar verir.

Sigmoid fonksiyonu sayesinde girdi kapısı hangi değerlerin kullanılması gerektiğine 4.12 ve 4.13 numaralı denklemler kullanılarak karar verecektir. 4.14 numaralı denklem tanh fonksiyonunda ise hücre durumu üzerine eklenmeye aday olan verilerden bir vektör oluşturulur. Sonrasında, bu oluşan iki vektör birleştirilerek Hücre durum vektörü üzerine eklenir. Daha sonra da Girdi ve Unutma kapılarından gelen bilgiler kullanılarak 4.15 numaralı denklemde gösterildiği gibi hücre durumu güncellenir. Hücre durumuna göre de her seferinde 4.16 ve 4.17 denklemleri ile çıktı vektörü güncellenir:

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_0) \tag{4.16}$$

$$h_t = o_t * tanh(C_t) \tag{4.17}$$

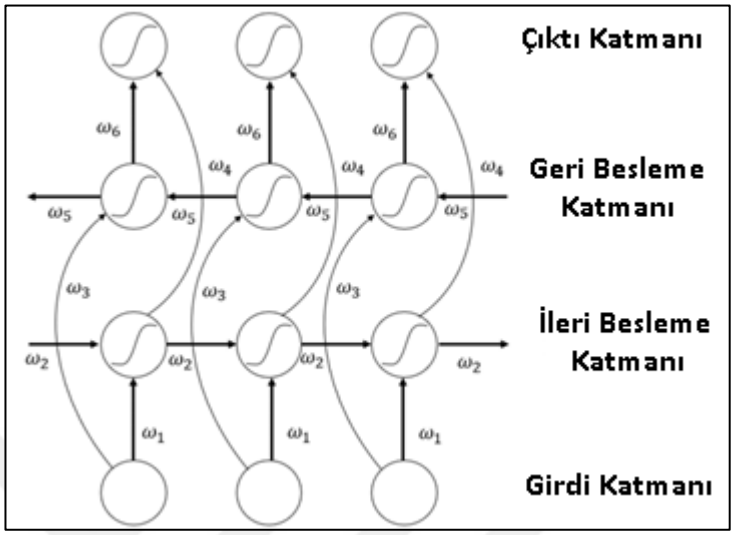
Ayrıca LSTM'de geri besleme (back propagation) yapılırken de RNN'in aksine her seferinde aynı W ağırlık matrisi ile çarpılmak yerine, her adımda farklı bir unutma kapısı ile çarpıldığı için gradyan kaybı veya taşması sorunundan da kurtulmuş olunur.

4.7 Çift Yönlü Uzun Kısa Vadeli Bellek (BiLSTM)

Çift Yönlü Uzun Kısa Vadeli Bellek modeli, çift yönlü RNN ağı olan bir LSTM ağıdır. Çift Yönlü LSTM veya BiLSTM, iki LSTM'den oluşan bir dizi işleme modelidir. Model giriş katmanında hem ileri yönlü hem de geri yönlü LSTM katmanı bulundurur. Giriş katmanında ki çift yönlü LSTM katmanları birleştirilerek çıktı katmanını oluşturur (LeCun vd., 2015).

Özetle, BiLSTM, bilgi akışının yönünü tersine çeviren bir LSTM katmanı daha ekler. Kısaca, giriş dizisinin ek LSTM katmanında geriye doğru aktığı anlamına gelir. Ardından, her iki LSTM katmanından gelen çıktıları, ortalama, toplam, çarpma veya birleştirme gibi çeşitli yollarla birleştirir. BiLSTM modeli, ağ için mevcut olan bilgi miktarını etkin bir şekilde artırarak, algoritma için mevcut olan bağlamı iyileştirir

(Zhou vd., 2018). Doğal dil işleme çalışmalarında çokça tercih edilen bir modeldir. Daha etkili sonuçlar elde etmeye yardımcı olabilmektedir (Zhou vd., 2018). Şekil 4.13'de BiLSTM modelinin yapısı gösterilmiştir.



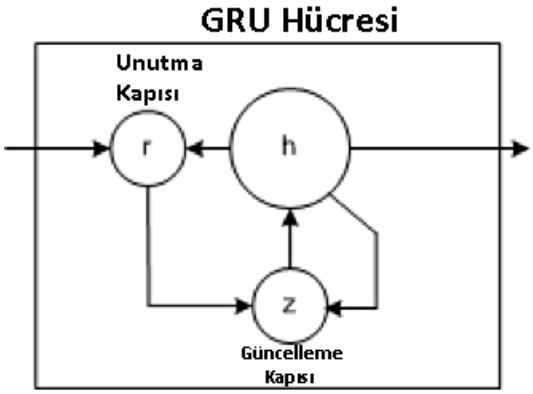
Şekil 4.13 Çift yönlü uzun kısa vadeli bellek model yapısı

4.8 Kapılı Tekrarlayan Birim (GRU)

Kapılı tekrarlayan birim (GRU), örneğin konuşma tanımada bellek ve kümeleme ile ilişkili makine öğrenimi görevlerini gerçekleştirmek için bir dizi boğum aracılığıyla bağlantıları kullanmayı amaçlayan belirli bir yinelenen sinir ağı modelinin parçasıdır (Cho, Van Merriënboer, Bahdanau ve Bengio, 2014; Heck ve Salem, 2017). Kapılı tekrarlayan birimler, tekrarlayan sinir ağlarında yaygın bir sorun olan kaybolan gradyan problemini çözmek için sinir ağı giriş ağırlıklarının ayarlanmasına yardımcı olur (Cho vd., 2014).

Genel tekrarlayan sinir ağı yapısının bir iyileştirmesi olarak, kapılı tekrarlayan birimler, bir güncelleme kapısı ve bir sıfırlama kapısı olarak adlandırılan yapıya sahiptir (Cho vd., 2014). Bu iki vektörü kullanarak model, model aracılığıyla bilgi akışını kontrol ederek çıktıları iyileştirir (Heck vd., 2017). Diğer tekrarlayan ağ modelleri gibi, kapılı tekrarlayan birimlere sahip modeller de bilgiyi belirli bir süre boyunca tutabilir, bu nedenle bu tür teknolojileri tanımlamanın en basit yollarından biri, onların "bellek merkezli" bir sinir ağı türü olmalarıdır (Cho vd., 2014). Buna

karşılık, kapılı tekrarlayan birimleri olmayan diğer sinir ağları türleri genellikle bilgiyi tutma yeteneğine sahip değildir (Cho vd., 2014).



Şekil 4.14 GRU hücresi çalışma yapısı

Kapılı tekrarlayan birim adı verilen LSTM'nin basitleştirilmiş hali olarak 2014 yılında tanıtıldı (Cho vd., 2014). Bu model, LSTM modelinde bulunan çıkış kapısından kurtulan iki kapıya sahiptir (Cho vd., 2014). GRU, LSTM'den daha basittir, daha hızlı eğitilebilir ve yürütülmesinde daha verimli olabilir (Heck vd., 2017). Ancak, LSTM daha anlamlı olabilir ve daha fazla veri ile daha iyi sonuçlara yol açabilir (Cho vd., 2014). Şekil 4.14'de örnek bir GRU hücresinin çalışma yapısındaki unutma ve güncelleme kapısı resmedilmiştir (Heck vd., 2017).

Konuşma tanımaya ek olarak, insan genomu, el yazısı analizi ve çok daha fazlası üzerinde araştırma yapmak için kapılı tekrarlayan birimleri kullanan sinir ağı modelleri kullanılabilir (Ravanelli, Brakel, Omologo ve Bengio, 2018). Bu yenilikçi ağlardan bazıları borsa analizinde ve devlet işlerinde kullanılmaktadır (Ravanelli vd., 2018). Birçoğu, makinelerin simüle edilmiş bilgiyi hatırlama yeteneğinden yararlanır.

4.9 Çift Yönlü Kapılı Tekrarlayan Birim (BI-GRU)

RNN, dizi verilerini işlemek için geliştirilmiş özel bir sinir ağıdır (Tao, Liu, Li ve Sidorov, 2019). Ancak, basit RNN'nin, kaybolan gradyan ve patlayan gradyan gibi, RNN'nin uzun vadeli bağımlılık görevlerini öğrenmesini zorlaştıran bazı dezavantajları vardır (Tao vd., 2019). Bu sorunları çözmek için özel bir RNN yapısı, yani LSTM ve GRU geliştirilmiştir (Tao vd., 2019). İlki, içerdiği kapılar (giriş kapısı,

unut kapısı ve çıkış kapısı) aracılığıyla uzun vadeli bilgileri takip edebilir (Hochreiter vd., 1997).

İkincisi, LSTM'nin geliştirilmiş bir versiyonudur; uzun vadeli bağımlılıkları öğrenir (Bahdanau, Cho ve Bengio, 2014). LSTM'den farklı olarak GRU'nun hafıza birimi yoktur ve 3 kapı yerine 2 kapıya (güncelleme kapısı ve sıfırlama kapısı) sahiptir, daha basit bir mimariye sahip olmak daha az hesaplama gerektirir ve daha hızlı eğitilebilir. GRU'nun yapısı o kadar karmaşık olmasada, araştırmalar performansının LSTM ile karşılaştırılabilir olduğunu göstermektedir (Chung, Gulcehre, Cho ve Bengio, 2014).

Çift Yönlü Kapılı Tekrarlayan Birim (Bidirectional Gated Recurrent Unit), Kapılı Tekrarlayan Birim algoritmasının hem ileri hem de geri yönde beslemeli halidir. Sinir ağlarının grafiksel gösterimi Şekil 4'te verilmiştir. Bir GRU içinde, güncelleme kapısı (z) bir sonraki duruma hangi bilgilerin tutulabileceğini belirtir ve sıfırlama kapısı (r), önceki durum bilgisinin yeni giriş bilgileri ile nasıl birleştirildiğini belirtir (Tao vd., 2019). GRU birimindeki bir sonraki çıktı ve durum değeri için hesaplama formülü aşağıdaki gibidir (Tao vd., 2019):

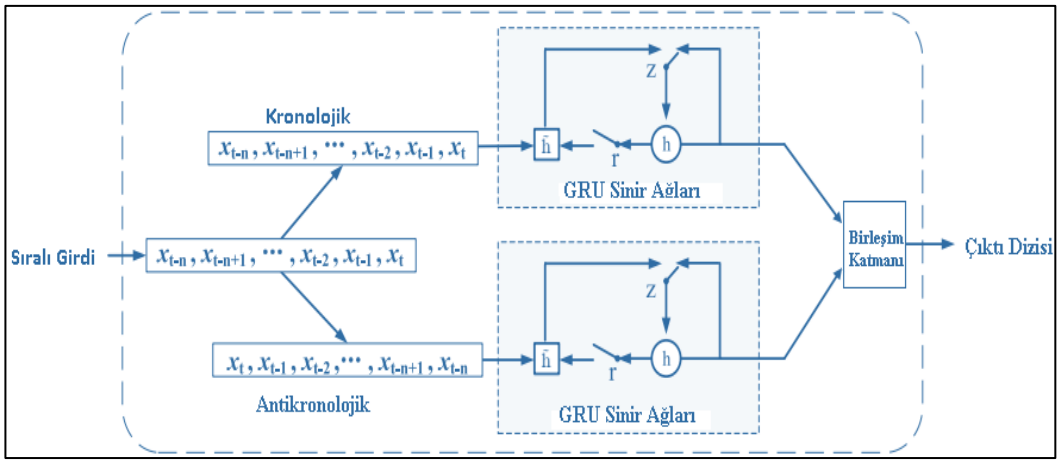
$$z_t = \sigma(W_z * [x(t), h(t-1)]) \tag{4.18}$$

$$r_t = \sigma(W_r * [x(t), h(t-1)]) \tag{4.19}$$

$$\bar{h}_t = \sigma(W_h * [x(t), (r_t * h(t-1)]) \tag{4.20}$$

$$h(t) = (1 - z_t) * h(t-1) + z_t * \bar{h}_t \tag{4.21}$$

Çift yönlü GRU yapısı denklem 4.18, 4.19, 4.20 ve 4.21'de gösterilmiştir. Burada σ aktivasyon fonksiyonudur, x(t) girdidir, h(t − 1) önceki çıktıdır, W_z, W_r ve W_h sırasıyla güncelleme geçidi, sıfırlama geçidi ve aday çıktının ağırlıklarıdır (Tao vd., 2019). Çift yönlü GRU, zaman serisinin iki yönünden kronolojik ve antikronolojik olarak gelen girdi dizisini işleyen ve ardından temsillerini bir araya getiren iki sıradan GRU modelinden oluşur (Tao vd., 2019). Şekilde çift yönlü kapılı tekrarlayan birim model yapısı gösterilmiştir.



Şekil 4.15 Çift yönlü kapılı tekrarlayan birim model yapısı

4.10 BERT

BERT (Bidirectional Encoder Representations from Transformers), transformatörlerden çift yönlü enkoder temsilleri anlamına gelir ve Google tarafından geliştirilmiş bir dil gösterimi modelidir (Devlin vd., 2018). 2018'de Google "AI Language" araştırmacıları tarafından geliştirilmiştir ve duygu analizi ve AVT gibi en yaygın 11'den fazla dil görevine İsviçre çakısı çözümü olarak hizmet etmiştir (Devlin vd., 2018). Bert modeli, doğal dil çıkarımı ve yorumlama gibi cümle seviyesindeki görevleri, bunları bütünsel olarak analiz ederek, belirteçlerin tanımlanması ve soru cevaplaması gibi görevleri, modellerin birbirine yakın bulanık çıktı üretmesi için gerekli olduğu cümlenin ögeleri seviyesinde görevleri gerçekleştirebilir (Tenney, Das ve Pavlick, 2019).

BERT, çok çeşitli dil görevlerinde kullanılabilir. Duygu durumu analizi yaparak bir filmin incelemelerinin ne kadar olumlu veya olumsuz olduğunu belirleyebilir (Adafre vd., 2005). Soru yanıtlamak için eğitildiğinde sohbet robotlarının soruları yanıtlamasına yardımcı olabilir (Yang vd., 2019). Bir e-posta yazarken metin tahminleme yaparak bir sonraki cümleyi tahmin edebilir (Shi ve Demberg, 2019). Uzun yasal sözleşmeleri hızlıca özetleyebilir (Miller, 2019). Polysemy çözünürlüğü ile birden çok anlamı olan ("yüz" gibi) kelimeleri çevreleyen metne göre ayırt edebilir (Mun, 2021). Doğal dil işleme çalışmaları, Google Translate'in, sesli yardımcıların Alexa, Siri vb. sohbet robotlarının, Google aramalarının, sesle çalışan küresel

konumlandırma sistemlerinin ve daha fazlasının çalışmasında kullanılmaktadır (Yoo ve Jeong, 2020).

BERT, ön eğitimli modellerin göreve özgü mimariye olan ihtiyacı azalttığını göstermektedir. BERT, çok görevli mimarinin daha iyi performans gösteren, hem büyük bir cümle düzeyi ve hem de belirteç seviyesi görevleri üzerine son teknoloji performansı elde eden ilk ince ayar (fine-grained) temelli gösterim modelidir.

3,3 Milyar kelimeden oluşan devasa bir veri seti, BERT modelinin eğitimi için kullanılmıştır. BERT modeli, yaklaşık olarak Vikipedi'den 2,5 milyar kelime ve Google BooksCorpus'dan 800 milyon kelime alınarak özel olarak eğitilmiştir. Bu büyük bilgi veri kümeleri, BERT'in yalnızca İngilizce diline değil, aynı zamanda dünyamıza ilişkin derin bilgisine de katkıda bulunmuştur.

BERT'in eğitimi, yeni Transformer mimarisi sayesinde mümkün oldu ve TPU'lar (Tensor Processing Units-Google'ın büyük ML modelleri için özel olarak oluşturulmuş özel devresi) kullanılarak hızlandırıldı. 64 adet TPU ile birlikte 4 gün boyunca BERT eğitimi aldı.

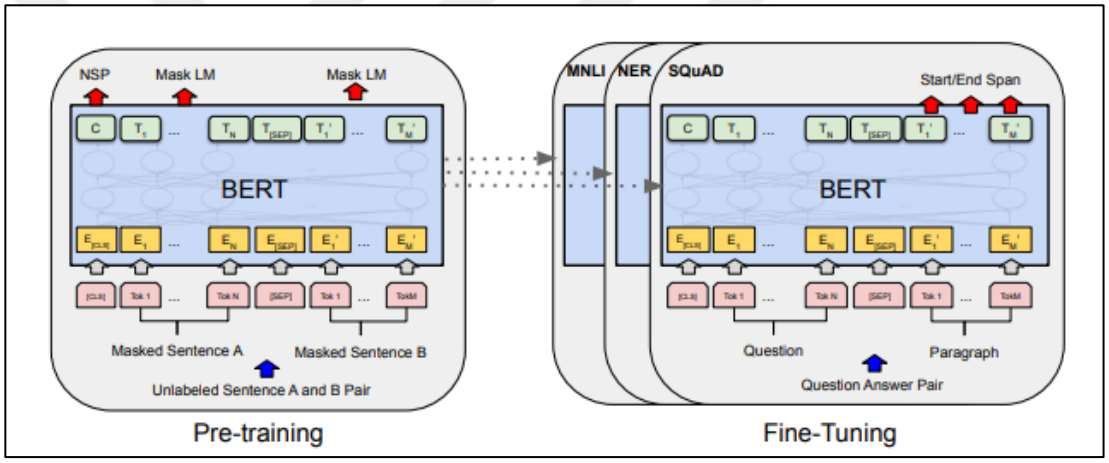
BERT'i daha küçük hesaplama ortamlarında (cep telefonları ve kişisel bilgisayarlar gibi) kullanmak için daha küçük BERT modellerine olan talep artmaktadır. Mart 2020'de 23 daha küçük BERT modeli piyasaya sürüldü. DistilBERT, BERT'in daha hafif bir versiyonunudur. BERT performansının %95'inden fazlasını korurken %60 daha hızlı çalışır. (Sanh, Debut, Chaumond ve Wolf, 2019)

4.10.1 Maskelenmiş Dil Modeli

BERT modeli, temelde maskelenmiş dil modeli mantığını kullanır. Maskelenmiş dil modelinde, rastgele bazı belirteçler girişten maskelidir ve amaç, sadece bağlamsal yani içerik olarak, maskeli kelimenin orijinal kelime kimliğini tahmin etmektir (Bao vd., 2020). Bir başka deyişle maskelenmiş dil modellemesi, bir cümledeki bir kelimeyi maskeleyerek yani gizleyerek ve BERT'i maskeli kelimeyi tahmin etmek için örtülü

kelimenin her iki tarafındaki kelimeleri çift yönlü olarak kullanmaya zorlayarak metinden çift yönlü öğrenmeyi sağlar.

Örneğin, Türkçe'de ki "yaz" kelimesi hem fiil hem de isim anlamına gelebilmektedir. Model, buradaki ayrımı yaparak "yaz" kelimesinin anlamına göre bir vektör ortaya çıkarmaktadır. Yani kelime mevsim olan "yaz" ise ayrı, fiil olan "yaz" kelimesi ise ayrı bir vektör oluşturmaktadır. Ayrıca, sağdan sola dil modelinin aksine BERT modeli, hedef kelimenin hem sağından hem solundan örnekleri birleştirerek çift yönlü derin dönüştürücülerin (bidirectional deep transformers) eğitimine izin verir. Bunlara ek olarak, BERT modeli bir sonraki cümle tahmini için de kullanılabilmektedir (Bao vd., 2020).

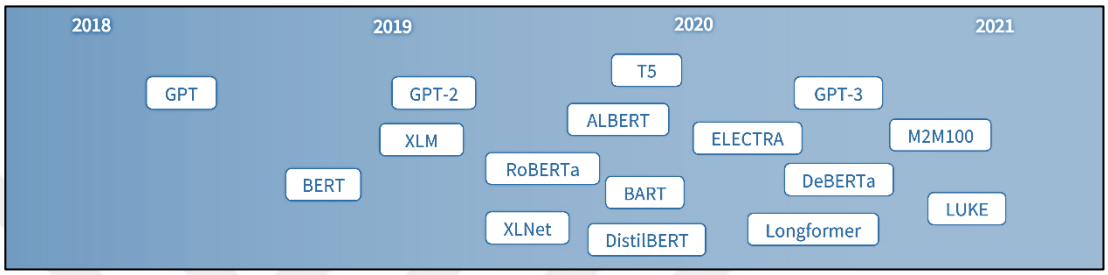


Şekil 4.16 BERT için genel ön eğitim ve ince ayar prosedürleri (Devlin vd., 2018)

Şekil 4.16'da, BERT için genel ön-eğitim ve ince ayar prosedürleri gösterilmektedir. Çıktı katmanlarının yanı sıra, aynı mimariler hem eğitim öncesi hem de ince ayarlarda kullanılmaktadır. Aynı önceden eğitilmiş model parametreleri, farklı aşağı akış görevleri için modelleri başlatmak için kullanılır. Şekilde, [CLS], her giriş örneğinin önüne eklenen özel bir semboldür ve [SEP], özel bir ayırıcı belirteçtir. Örneğin, soruları ve cevapları iki ayrı gruba ayırır.

4.10.2 Transformatörler

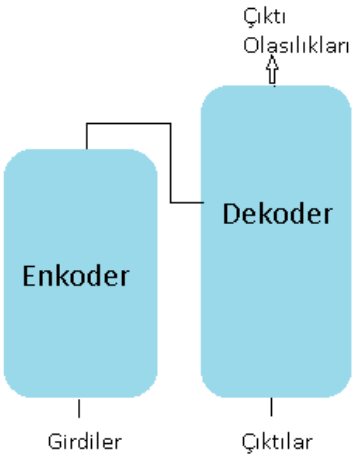
Transformatör mimarisi, makine öğrenimi eğitimini son derece verimli bir şekilde paralel hale getirmeyi mümkün kılar. Bu nedenle, büyük paralelleştirme, BERT'i nispeten kısa bir süre içinde büyük miktarda veri üzerinde eğitmeyi mümkün kılar (Lin, Nogueira ve Yates, 2021). Transformatörler, kelimeler arasındaki ilişkileri gözlemlemek için bir dikkat (attention) mekanizması kullanır (Lin vd., 2021). Şekil 4.17'de popüler transformatör modeli sürümlerinin zaman çizelgesi gösterilmiştir.



Şekil 4.17 Popüler transformatör modeli sürümlerinin zaman çizelgesi

Transformatörler, ilk olarak bilgisayarlı görü modellerinde görülen güçlü bir derin öğrenme algoritması olan dikkatten yararlanarak çalışır. İnsanların dikkat yoluyla bilgiyi işleme şeklimizden o kadar da farklı değil. İnsanlar tehdit oluşturmayan veya yanıt gerektirmeyen sıradan günlük girdileri unutmak veya görmezden gelmek konusunda gayet başarılıdır (Lin vd., 2021). Örneğin, yolda yürürken gördüğünüz ve duyduğunuz her şeyi hatırlamayız. Beynimizin hafızası sınırlı ve değerlidir. Önemsiz girdileri unutma yeteneğimiz hatırlamamıza yardımcı olur.

Benzer şekilde, makine öğrenimi modellerinin, ilgisiz bilgileri işleyen hesaplama kaynaklarını boşa harcamadan yalnızca önemli olan şeylere nasıl dikkat edileceğini öğrenmesi gerekir (Lin vd., 2021). Transformatörler, bir cümledeki hangi kelimelerin daha sonraki işlemler için en kritik olduğunu gösteren farklı ağırlıklar yaratır (Lin vd., 2021).



Şekil 4.18 Transformatörlerde enkoder-dekoder yapısı[2]

Bir transformatör bunu, genellikle kodlayıcı (enkoder) olarak adlandırılan bir transformatör katmanları yığını aracılığıyla bir girişi art arda işleyerek yapar (Jwa, Oh, Park, Kang ve Lim, 2019). Gerekirse, bir hedef çıktıyı tahmin etmek için başka bir transformatör katmanı yığını kod çözücü (dekoder) kullanılabilir. Ancak BERT bir kod çözücü kullanmaz. Transformatörler, milyonlarca veri noktasını verimli bir şekilde işleyebildikleri için denetimsiz öğrenme için benzersiz bir şekilde uygundur yapar (Jwa vd., 2019).

Model temel olarak iki bloktan oluşmaktadır. Şekil 4.18'de modelin enkoder dekoder yapısı gösterilmiştir. Kodlayıcı bir girdi alır ve özelliklerinin bir temsilini oluşturur. Bu, modelin girdiden anlayış elde etmek için optimize edildiği anlamına gelir yapar (Jwa vd., 2019). Kod çözücü, bir hedef dizi oluşturmak için kodlayıcının gösterimini (özelliklerini) diğer girişlerle birlikte kullanır. Bu, modelin çıktı üretmek için optimize edildiği anlamına gelmektedir yapar (Jwa vd., 2019).

[2] https://huggingface.co/course/chapter1/4

Tablo 4.1 Transformatör mimarisinin parçaları ve tanımları

Model Mimarisinin Parçaları	**Tanım**
Parametreler	Model için kullanılabilen öğrenilebilir değişkenlerin/değerlerin sayısı
Transformatör Katmanları	Transformatör bloklarının sayısı. Bir transformatör bloğu, bir dizi sözcük
Gizli Boyut	Girdi ve çıktı arasında yer alan, istenen sonucu elde etmek için ağırlıkları
Dikkat Başları	Bir transformatör bloğunun boyutu
İşleme	Modeli eğitmek için kullanılan işlem birimi
Eğitim uzunluğu	Modeli eğitmek için geçen süre

Tablo 4.1'de model mimarisinin parçaları ve tanımı gösterilmiştir. Enkoder ve dekoder yapılarından her biri, göreve bağlı olarak bağımsız bir şekilde kullanılabilir. Yalnızca kodlayıcı modeller cümle sınıflandırması ve AVT gibi girdinin anlaşılmasını gerektiren görevler için iyidir yapar (Jwa vd., 2019). Yalnızca kodlayıcı modelleri, cümle sınıflandırması ve AVT gibi girdinin anlaşılmasını gerektiren görevler için iyidir (He, Zhao, Yang, Zhang ve Li, 2019). Yalnızca kod çözücü modeller, metin oluşturma gibi üretken görevler için iyidir (He vd., 2019). Kodlayıcı ve kod çözücü modelleri veya diziden diziye modeller (sequence to sequence), çeviri veya özetleme gibi girdi gerektiren üretken görevler için iyidir yapar (Jwa vd., 2019).

Transformatör modellerinin önemli bir özelliği, dikkat katmanları adı verilen özel katmanlarla oluşturulmuş olmalarıdır (Liu ve Lapata, 2019). Bu katman, modele, sıradaki cümledeki belirli kelimelere özellikle dikkat etmesini ve her kelimenin temsili ile uğraşırken diğerlerini az çok görmezden gelmesini söylemektedir (Liu vd., 2019). Tablo 4.2'de temel ve geniş BERT modellerinin özellikleri gösterilmiştir.

Tablo 4.2 Temel ve geniş BERT modellerinin sahip olduğu özellikler

	Transformatör Katmanları	Gizli Katman	Dikkat Başları	Parametreler	İşleme	Eğitim Uzunluğu
BERTbase	12	768	12	110 milyon	4 TPU	4 gün
BERTlarge	24	1024	16	340 milyon	16 TPU	4 gün

Bunu daha iyi bir bağlama oturtmak için, metni İngilizceden Fransızcaya çevirme görevini ele alalım. "Bu kursu beğendin" girdisi göz önüne alındığında, "beğen" kelimesinin doğru çevirisini elde etmek için bir çeviri modelinin bitişikteki "Sen" kelimesine de katılması gerekecektir, çünkü Fransızcada beğenmek fiili duruma bağlı olarak farklı şekilde çekimlenebilir. Bununla birlikte, cümlenin geri kalanı o kelimenin çevirisi için kullanışlı değildir (Liu vd., 2019). Aynı şekilde, bu kelimesini çevirirken modelin kurs kelimesine de dikkat etmesi gerekecektir, çünkü bu, ilişkili ismin eril mi dişil mi olduğuna bağlı olarak farklı şekilde çevrilir (He vd., 2019).

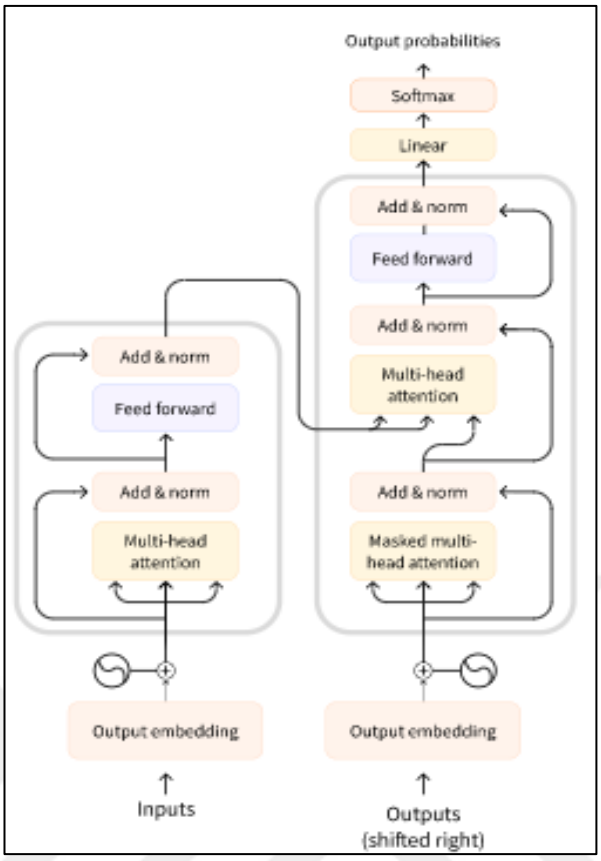
Yine cümledeki diğer kelimelerin bu kelimesinin tercümesi için önemi olmayacaktır. Daha karmaşık cümleler ve karmaşık dilbilgisi kuralları söz konusu olduğunda, modelin her bir kelimeyi doğru bir şekilde çevirmek için cümlede daha uzakta görünebilecek kelimelere özellikle dikkat etmesi gerekmektedir (Liu vd., 2019). Aynı kavram, doğal dil ile ilişkili herhangi bir görev için de geçerlidir: bir kelimenin kendi başına bir anlamı vardır, ancak bu anlam, çalışılan kelimeden önce veya sonra başka herhangi bir kelime veya kelime olabilen bağlamdan derinden etkilenir.

Transformatör mimarisi orijinal olarak diller arası çeviri için tasarlanmıştır. Eğitim sırasında kodlayıcı belirli bir dilde girdiler (cümleler) alırken, kod çözücü aynı cümleleri istenen hedef dilde alır (Lin vd., 2021). Kodlayıcıda, dikkat katmanları bir cümledeki tüm kelimeleri kullanabilir çünkü belirli bir kelimenin çevirisi cümlede ondan önce ve sonra olana bağlı olabilmektedir. Bununla birlikte, kod çözücü sıralı olarak çalışır ve yalnızca daha önce tercüme ettiği cümledeki kelimelere dikkat edebilir. Bu nedenle, yalnızca o anda üretilen kelimeden önceki kelimeler işleme alır

(Lin vd., 2021). Örneğin, çevrilen hedefin ilk üç kelimesini tahmin ettiğinde, bunlar kod çözücüye gönderilir ve kod çözücü daha sonra dördüncü kelimeyi tahmin etmeye çalışmak için kodlayıcının tüm girdilerini kullanır.

Eğitim sırasında işleri hızlandırmak için modelin hedef cümlelere erişimi olduğunda, kod çözücü hedefin tamamıyla beslenir, ancak gelecekteki kelimeleri kullanmasına izin verilmez (Lin vd., 2021). Örneğin, dördüncü kelimeyi tahmin etmeye çalışırken, dikkat katmanı yalnızca 1 ila 3 konumlarındaki kelimelere erişebilir. Orijinal transformatör mimarisi, kodlayıcı solda ve kod çözücü sağda olacak şekilde Şekil 4.19'da gösterilmiştir.

Bir kod çözücü bloğundaki birinci dikkat katmanının, kod çözücüye gelen tüm geçmiş girdilere dikkat etmektedir. Ancak ikinci dikkat katmanının kodlayıcının çıktısını kullanmaktadır. Böylece mevcut kelimeyi en iyi şekilde tahmin etmek için tüm giriş cümlesine erişebilmektedir. Bu, farklı dillerin kelimeleri farklı sıralara koyan gramer kurallarına sahip olabileceğinden veya cümlede daha sonra sağlanan bazı bağlamların belirli bir kelimenin en iyi çevirisini belirlemeye yardımcı olabileceğinden çok kullanışlı olabilmektedir (He vd., 2019).



Şekil 4.19 Ayrıntılı transformatör mimarisi

Dikkat maskesi, modelin bazı özel kelimelere dikkat etmesini önlemek için kodlayıcı veya kod çözücüde de kullanılabilir. Örneğin, cümleleri bir araya toplarken tüm girdileri aynı uzunlukta yapmak için kullanılan özel dolgu sözcüğü gibi bir yapı kullanmak mantıklı olacaktır (He vd., 2019).

4.11 Koşullu Rastgele Alanlar Algoritması

Koşullu rastgele alanlar (Conditional Random Fields-CRF), (Lafferty vd., 2001) koşullu bir yaklaşıma dayalı olarak sıralı verileri etiketlemek ve bölümlere ayırmak için olasılıklı bir algoritmadır. Bir CRF, belirli bir gözlem dizisi verilen etiket dizileri üzerinde tek bir logaritmik doğrusal dağılımı tanımlayan bir yönlendirilmemiş grafik model biçimidir (Wallach, 2004). CRF'lerin gizli Markov modellerine (Hidden Markov Model-HMM) göre birincil avantajı, izlenebilir çıkarımı sağlamak için HMM'lerin gerektirdiği bağımsızlık varsayımlarının gevşetilmesiyle sonuçlanan koşullu yapılarıdır.

Ek olarak, CRF'ler, maksimum entropi Markov modelleri (McCallum, Freitag ve Pereira, 2000) ve yönlendirilmiş grafik modellere dayalı diğer koşullu Markov modelleri tarafından sergilenen bir zayıflık olan etiket önyargı probleminden kaçınır (Lafferty vd., 2001). CRF'ler, bir dizi gerçek dünya dizi etiketleme görevinde hem maksimum entropi Markov modellerden hem de HMM'lerden daha iyi performans gösterir (Wallach, 2004).

Lafferty vd., (2001) belirli bir y etiket dizisinin, verilen gözlem dizisi x'in, potansiyel fonksiyonların normalleştirilmiş bir ürünü olma olasılığını tanımlarlar. Bir dizi (sequence) sınıflandırma modelinde nihai amaç, girdi dizi vektörlerine X'e göre verilen bir etiket dizisinin Y olasılığını bulmaktır. Bu, P(Y|X) olarak belirtilir.

- Eğitim seti: giriş ve hedef sıra çiftleri $\{(X_i\ ,\ y_i)\}$
- Vektörlerin i'inci giriş dizisi: $X_i = [x_1 \ldots\ x_l]$
- Etiketlerin i'inci hedef dizisi: $Y_i = [y_1 \ldots\ y_l]$
- ℓ sıra uzunluğudur.

Denklem 4.18'de P(Y|X) fonksiyonu gösterilmiştir. Bir örnek (X,Y) için, düzenli bir sınıflandırma probleminde, $1 \leq k \leq l$ olan dizideki k'inci pozisyondaki her bir ögenin olasılığını çarparak P(Y|X) 'i hesaplayabiliriz (Sutton ve McCallum, 2012):

$$P(Y|X) = \frac{\exp\left(\sum_{k=1}^{l} U(x_k, y_k)\right)}{\prod_{k=1}^{l} Z(x_k)} \tag{4.18}$$

U(x,y) emisyonlar veya tekli puanlar olarak adlandırılır. Bu sadece k. zaman adımında x vektörüne verilen y etiketi için bir puandır. Bunu bir LSTM'nin k. çıktısı olarak varsayılmaktadır. Uygulamada, x vektörü genellikle, kayan bir pencereden sözcük yerleştirmeleri gibi çevreleyen öğelerin birleşimidir. Her bir tekli faktör, modelde öğrenilebilir bir ağırlıkla ağırlıklındırlar.

Z(X) genellikle bölme işlevi olarak adlandırılır. Olasılıkları alınmak istendiği için bu bir normalleştirme faktörü olarak düşünülmektedir. Bu durumda her farklı etiketin puanları 1'e eşit olmalıdır. (Nielsen, Frank ve Ke Sun, 2021)

Emisyonlar veya tekli puanlar (U), bu skor değeri, x_k girdisi verildiğinde y_k'nin ne kadar olası olduğunu temsil eder. Bölme fonksiyonu (Z), sonunda bir olasılık elde etmek için normalleştirme faktörüdür (Nielsen vd., 2021)

BÖLÜM BEŞ

ÖN EĞİTİMLİ KÜTÜPHANELER İLE ADLANDIRILMIŞ VARLIK TANIMA YAPILMASI

Bu çalışma da Spacy ve Stanford kütüphanelerinin AVT modülleri kullanılarak varlık adlarının tespiti gerçekleştirilmiştir. Toplamda 47928 adet cümle içerisinde yapılan varlık ismi tanıma çalışmasında etiketli veri Kaggle.com sitesi üzerinden alınmıştır. Alınan veriler IOB2 etiketleme yapısına uygun olarak etiketlenmiştir. Cümleler tek tek ön eğitimli kütüphanelerin modüllerine verilmiştir. Çıkan sonuçların doğruluklarını kontrol edebilmek için etiketli verideki varlık grupları liste haline getirilmiştir. Etiketli varlık adları gruplarıyla ön eğitimli kütüphanelerin tahmin ettiği varlık adları karşılaştırılarak modüllerin varlık adlarını tanıma doğruluk oranları elde edilmiştir. Çıkan sonuçlar ayrıntılı olarak karşılaştırılmış ve değerlendirilmesi yapılmıştır.

5.1 Spacy Kütüphanesi

Spacy, Python programlama dilinde geliştirilmiş doğal dil işleme için ücretsiz, açık kaynaklı bir kütüphanedir. Çok fazla metinle çalışırken metin hakkında özet bilgi çıkarmak istenilebilir (Tutubalina ve Nikolenko, 2017). Örneğin, metin ne hakkındadır? Kelimeler anlamsal bağlamda ne anlama geliyor? Kim kime ne yapıyor? Hangi şirket ve ürünlerden bahsediliyor? Hangi metinler birbirine benziyor? Bu gibi kavramları keşfetmek için spacy kütüphanesinin açık kaynaklı kodları kullanılabilmektedir.[3]

Spacy özellikle üretim kullanımı için tasarlanmıştır ve büyük hacimli metinleri işleyen ve "anlayan" uygulamalar oluşturmanıza yardımcı olur. Bilgi çıkarma veya doğal dil anlama sistemleri oluşturmak veya derin öğrenme için metni önceden işlemek için kullanılabilir. (Orellana vd., 2020) Spacy modelleri genelde 3 tipten oluşur Bunlar; küçük (sm-small), orta (md-medium), büyük (lg-large) paketlerdir.[3]

[3] https://spacy.io/usage/spacy-101

Spacy kütüphanesi içerisinde pek çok özelliği barındırmaktadır. Bu özellikler alt modül olarak kodlanmıştır ve bunlardan bazıları şu şekildedir. Tokenization (jetonlaştırıcı), Part-of-speech (POS) Tagging (sözcük türü etiketleyici), Dependency Parsing (bağımlılık ayrıştırma), Lemmatization (kök çözümleme), Sentence Boundary Detection (cümle sınırları bulma), Named Entity Recognition (AVT) ve Entity Linking (varlık bağlama) vb. birçok modülü daha içerisinde barındırmaktadır. Yazımızın bir sonraki aşamasında Spacy kütüphanesinin AVT modülü olan Named Entity Recognition modülü anlatılacaktır.

AVT modülünün algoritması, geçiş tabanlı adlandırılmış bir varlık tanıma bileşenidir. Varlık tanıyıcı, örtüşmeyen etiketlenmiş belirteç aralıklarını tanımlamaktadır. Kullanılan geçiş tabanlı algoritma, geleneksel olarak adlandırılan varlık tanıma görevleri için etkili olan ancak her yayılma tanımlama problemi için uygun olmayabilmektedir. Spesifik olarak, kayıp fonksiyonu tüm varlık doğruluk değerleri için algoritmayı optimize eder, bu nedenle sınır belirteçleri üzerinde açıklayıcılar arası uyuşma düşükse, bileşen muhtemelen problem üzerinde düşük performans gösterecektir.[4]

Modülde ki geçiş tabanlı algoritma ayrıca varlıklar hakkında en belirleyici bilgilerin ilk belirteçlerine yakın olacağını varsaymaktadır. Varlık adları uzunsa ve ortasında belirteçlerle karakterize ediliyorsa, bileşen muhtemelen görev için uygun olmayacaktır.

5.2 Stanford Kütüphanesi

Stanford Üniversitesi tarafından kurulan doğal dil işleme grubu (The Stanford Natural Language Processing Group) bazı doğal dil işleme görevleri yapan kütüphaneler oluşturmuştur. Stanford doğal dil işleme grubu, Doğal Dil İşleme yazılımlarından bazılarını herkesin kullanımına sunmaktadır. İnsan dili teknolojisi ihtiyaçları olan uygulamalara dahil edilebilecek büyük hesaplamalı dilbilim sorunları için istatistiksel NLP, derin öğrenme NLP ve kural tabanlı NLP araçları

[4] https://spacy.io/api/entityrecognizer

sağlamaktadırlar. Bu paketler endüstride, akademide ve devlette yaygın olarak kullanılmaktadır.[5]

Desteklenen tüm yazılım dağıtımlarımız Java ile yazılmıştır. Yazılımların Ekim 2014'ten sonraki güncel sürümleri Java 8 ve üzeri versiyon gerektirmektedir. Dağıtım paketleri, komut satırı çağrısı, jar dosyaları, bir Java API'si ve kaynak kodu için bileşenleri içerir. Bir dizi yardımsever insan, diğer diller için ciltler veya çeviriler ile çalışmaları genişletilmiştir. Sonuç olarak, bu yazılımın çoğu Python (veya Jython), Ruby, Perl, Javascript, F# ve diğer .NET ve JVM dillerinden de kolaylıkla kullanılabilmektedir.

Kütüphane pek çok alt NLP modülü içermektedir. Bunlardan bazıları şu şekildedir. Stanford Parser (Bölücü), Stanford POS Tagger (Sözcük Türü Etiketleyici), Stanford Named Entity Recognizer (AVT modülü), Stanford RegexNER (kurallı ifadeler ile AVT), Stanford Coreference Resolution (eş referans çözümlemesi), Stanford Word Segmenter (Kelime Bölümleyici), Stanford Classifier (sınıflandırıcı), Stanford EnglishTokenizer (İngilizce jetonlaştima aracı) vb. Bu modüllerin bazıları kural tabanlı sistemler ile çalışırken bazıları derin öğrenme ve makine öğrenmesi kullanmaktadır. Yazımızı devam eden kısmında kullandığımız modül olan Stanford Named Entity Recognizer açıklanmıştır.

Adlandırılmış Varlık Tanıma, kişi ve şirket adları veya gen ve protein adları gibi şeylerin adları olan bir metindeki sözcük dizilerini varlık adı olarak tanımlar. Stanford NER, Adlandırılmış Varlık Tanıyıcının Java uygulamasıdır (Shelar, Kaur, Heda ve Agrawal, 2020). AVT için iyi tasarlanmış özellik çıkarıcılarla ve özellik çıkarıcıları tanımlamak için birçok seçenekle birlikte gelir. İndirmeye dahil olanlar, özellikle üç sınıf (Kişi, Organizasyon, Konum) için İngilizcede iyi olan adlandırılmış varlık tanıyıcılardır ve ayrıca CoNLL 2003 için eğitilmiş modeller de dahil olmak üzere farklı diller ve koşullar için çeşitli diğer modelleri de kullanıma sunmaktadır (Shelar vd., 2020).

[5] https://nlp.stanford.edu/software/

Kütüphane eğitilmiş farklı dil paketlerinden oluşmaktadır ve İngilizce temel dil paketi olarak sunulmaktadır. İngilizce dışındaki Almanca, Fransızca, İspanyolca, Çince ve Arapça dilleri içinde paketleri bulunmaktadır. Fakat bu paketlerin ayrıca indirilmesi gerekmektedir.[6]

Kütüphane İngilizce için 3 farklı modelden oluşmaktadır. Bunlar; üç adet sınıf (Konum, Organizasyon, Kişi) için eğitilen model, dört adet sınıf (Konum, Organizasyon, Kişi, Tür) için eğitilen model, yedi adet sınıf (Kişi, Organizasyon, Konum, Yüzde, Para, Tarih, Zaman) için eğitilen modeldir. 4 sınıflı model CoNLL 2003 konferansındaki İngilizce eğitim verisiyle eğitilmiştir. 7 sınıflı model MUC6 ve MUC7 konferanslarındaki veriler ile eğitilmiştir. 3 sınıflı model bu iki veri seti ve bunlara ek verilerle eğitilmiştir.

Stanford NER, CRF Sınıflandırıcısı olarak da bilinir. Yazılım (rastgele sırayla) doğrusal zincir Koşullu Rastgele Alan (Conditional Random Fields-CRF) dizi modellerinin genel bir uygulamasını sağlar (Yadav ve Bethard, 2019). Diğer bir deyişle, kendi modellerinizi etiketli veriler üzerinde eğiterek, bu kodu AVT veya başka bir görev için sıra modelleri oluşturmak için kullanılabilir.

5.3 Veri seti

Bu çalışmada kaggle.com sitesi üzerinden alınan İngilizce haber yazılarını içeren bir veri seti kullanılmıştır. Veri setinde 1354149 kelime ve 47928 adet cümle bulunmaktadır. Veri seti GMB (Groningen Meaning Bank) yapısına uygun olarak ve IOB2 (Inside, Outside, Beginning) etiketi kullanarak etiketlenmiştir. Veri seti insan eli ile etiketlenmiştir. Örnek veri setinin bir kısmı Tablo 1 ‘de gösterilmiştir. Tablo veri setinde ki her bir kelime bir cümleye ait olarak .csv formatı içerisinde etiketi ile birlikte verilmiştir. Etiketi verilen kelimeler söz öbekleri halinde alınmış ve cümleler ile Spacy ve Stanford kütüphaneleri ile AVT işlemi yapılmıştır.

[6] https://nlp.stanford.edu/software/CRF-NER.html

Külliyat yapısı Groningen Üniversitesi'nde geliştirilen Groningen Anlam Bankası ham ve simgeleştirilmiş formatta binlerce metin, konuşmanın bir bölümü için etiketler, adlandırılmış varlıklar ve sözcük kategorileri ve birinci dereceden mantıkla uyumlu söylem temsil yapılarından oluşmaktadır.[7]

Veri setinde her kelime tek tek etiketlendiği için kütüphanelerin girdi ve çıktı yapılarına uymamaktadır. İki kütüphane de cümle yapısında girdi aldığı için veri setinde ki kelimeler birleştirilmiş ve sonuçlar olarak cümleler halinde gruplanmış hale getirilmiştir. Sonra cümleler sırasıyla kütüphanelere varlık adlarını tahmin etmesi için verilmiştir. Tablo 5.1'de örnek etiketlenmiş veri setinin bir kısmı gösterilmiştir. Kütüphaneler çıktı yapıları da kendilerini özeldir. Çıkan sonuçların doğruluğunu kontrol edebilmek için varlık adları gruplanmış halde listelendi.

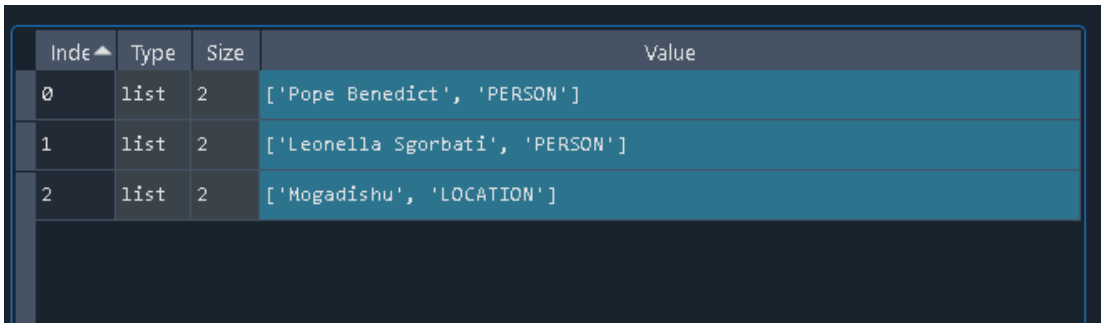
Tablo 5.1 Örnek etiketlenmiş veri seti

Cümle Numarası	Kelime	Etiket
1	thousand	O
1	of	O
1	demonstrator	O
1	have	O
1	march	O
1	through	O
1	london	B-geo

Örneğin üç kelimeden oluştuğu için ve hepsi bir varlık adını oluşturduğu için "Mustafa Kemal Atatürk" kelimeleri bir varlık adı olarak etiketlenmiş ve "PERSON " etiketi verilmiştir. Bir başka örnek ise kütüphaneler arası tahminleme biçimini anlamak içindir. Örneğin İzmir kelimesi Stanford Kütüphanesinde "LOCATION" olarak tahmin edilirken Spacy kütüphanesinde "GPE" olarak tahmin edilmektedir. Bu gibi benzeri tüm durumları karşılamak için veri setinde ki varlık adları ve etiketleri

[7] https://gmb.let.rug.nl/

kütüphanelere özel olarak düzenlenmiştir. Örnek düzenlemeler Şekil 5.1 ve 5.2'de gösterilmiştir.



Inde	Type	Size	Value
0	list	2	['Pope Benedict', 'PERSON']
1	list	2	['Leonella Sgorbati', 'PERSON']
2	list	2	['Mogadishu', 'LOCATION']

Şekil 5.1 Stanford kütüphanesi için düzenlenmiş etiketli veri

Inde	Type	Size	Value
0	list	2	['Iran', 'GPE']
1	list	2	['President Mahmoud Ahmadinejad', 'PERSON']
2	list	2	['Tuesday', 'DATE']
3	list	2	['European', 'GPE']
4	list	2	['Iran', 'GPE']
5	list	2	['Iranian', 'GPE']

Şekil 5.2 Spacy kütüphanesi için düzenlenmiş etiketli veri

5.4 Eğitim Sonuçları

Bu veri seti ile AVT uygulaması yapılırken kullanılan kütüphane modellerinin en büyük versiyonları kullanılmıştır. Spacy için "en_core_web_lg – 3.0.0" modeli, Stanford Kütüphanesi için ise "stanford-ner-4.0.0" kullanıldı. Modellerin başarısını ölçmek için elde edilmiş keskinlik, hassasiyet ve F1 skor değerleri aşağıda verilmiştir. Spacy modeli kullanılarak veri setinde geçen varlık adları, coğrafi varlık, organizasyon, kişi adları, jeopolitik varlık, önemli olay adları tahmin edildi. Stanford kütüphanesi ile organizasyon, kişi adları ve konum adları tahmin edildi.

Tablo 5.2 Stanford kütüphanesi ön eğitimli model sonuçları

	Precision	**Recall**	**F1-score**
Location	0,8933	0,8937	0,8935
Organization	0,8686	0,6884	0,7680
Person	0,7618	0,9405	0,8417
Accuracy			0,8547
Weighted avg	0,8563	0,8547	0,8512

Tablo 5.3 Spacy kütüphanesi ön eğitimli model sonuçları

	Precision	**Recall**	**F1-score**
Date	0,9920	0,9498	0,9704
Event	0,2825	0,5206	0,3663
Geo	0,8684	0,8468	0,8575
Gpe	0,9090	0,8913	0,9001
Organization	0,8015	0,6823	0,7371
Person	0,8355	0,8785	0,8564
Accuracy			0,8446
Weighted avg	0,8762	0,8446	0,8593

Tablo 5.4 Modeller için karşılaştırmalı f1-skor tablosu

	Spacy NER Modeli	**Stanford NER Modeli**
Organization (ORG)	0,7371	0,7680
Person (PER)	0,8564	0,8417
Geographical (GEO)	0,8575	0,8935
Geo-Political Entity (GPE)	0,9001	-
Date	0,9704	-
Event	0,3663	-

Modellerin keskinlik, hassasiyet ve F1 skor değerleri Tablo 5.2 ve Tablo 5.3'te verilmiştir. En son Tablo 5.4'te ise iki modelin F1 skor başarı değerleri karşılaştırmalı olarak gösterilmiştir.

5.5 Sonuçların Değerlendirilmesi

Organizasyon adı ve konum adlarında F1 skor bazında Stanford kütüphanesi %3-%4 civarında daha iyi sonuç verdi. Ayrıca konum adlarında hem keskinlik hem de hassasiyet bazında Stanford modeli daha iyi sonuç verdi. Bunun nedeni, Spacy modelinde konum için "geo" (coğrafi varlık) alanı kontrol edildi. Ama Spacy modeli anlamsal olaraktan yakın olan coğrafi varlık ile jeopolitik varlık (Örn: İngiltere ve İngiliz) arasında bazen karışıklık yaşadı. Bu yüzden Spacy kütüphanesi konum adı tahmininde Stanford kütüphanesine oranla daha az başarı sağladı. Organizasyon adı tahmini için keskinlik değeri bazında da Stanford kütüphanesi %6-7 civarında bir fark sağladı. Bu sonuca bakarak bu veri setinde Stanford modeli organizasyon adlarını diğer varlık adlarından ayırmada daha başarılıdır. Kişi adları tanımada keskinlik ve hassasiyet değerleri bazında Spacy modeli keskinlikte daha iyiyken, Stanford modeli hassasiyette daha iyi bir değer sergiledi. F1 skorunda da %1-%2 civarında Spacy modeli daha başarılı sonuç verdi. Spacy modeli kişi adı belirlemede keskinlik ve hassasiyet yakınlığı bakımından da daha iyi sonuç verdiği için bu veride daha tercih edilebilir durmaktadır.

Sadece Spacy kütüphanesinin modeli ile tahmin edilen varlık adları ise tarih, önemli olay adları ve jeopolitik varlık adlarıdır. Tarihte F1 skor başarısı %97 civarındayken, jeopolitik varlık adlarında başarı %90 civarındadır. Bununla birlikte Spacy modeli bu veri setinde önemli olay adlarının tanımada pek fazla başarı gösteremedi ve F1 skor başarısı %30 civarında kaldı. Ayrıca bu çalışma da 47928 cümle ele alındı. Eğitimler Intel i5 beşinci nesil 2.20GHz hızında bir işlemci ve 8 GB ram kullanılan 64 bit bir cihazda gerçekleştirilmiştir. Spacy modelinin toplam tahmin süresi 12 dakika civarında iken Stanford kütüphanesinin toplam tahmin süresi ise 28,9 saat civarındadır. Sonuç olarak bu çalışma da Stanford ve Spacy kütüphanelerinin AVT modülleri kullanılarak İngilizce haber metinlerinden oluşan yazıda varlık isimleri tanınmaya çalışılmıştır. Veri seti farklı varlık gruplarından oluşmaktadır. Bu yüzden her varlık grubu için elde edilen doğruluk sonucu ayrı olarak değerlendirilmiş ve karşılaştırması yapılmıştır. Elde edilen bulgular bu alanda hızlı bir şekilde varlık adı tanıma işlemi yapacak kişilere yol gösterme amacıyla yapılmıştır. Sonuçlar farklı veri gruplarında farklılık gösterebilir niteliktedir.

BÖLÜM ALTI

MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TÜRKÇE TWEETLER ÜZERİNDE KÜFÜR TESPİTİ YAPILMASI

6.1 Veri Seti ve Ön İşleme

Bu çalışmada, derin öğrenme tabanlı Bi-LSTM ve BERT modelleri kullanarak Türkçe tweet metinleri üzerinde küfür, hakaret veya aşağılayıcı kelime tespitine çalışılmıştır. Küfür tespitinin kelime bazında yapılması ve problemin ele alınırken adlandırlmış varlık tanıma problemi olarak ele alınmasına çalışılmıştır. Bu yüzden devam eden süreçte ön işlem etiketleme ve eğitim adımları AVT olarak ele alınmış ve buna göre süreç ilerletilmiştir.

Çalışmada kullanılan veri seti, Twitter'dan alınmış Türkçe tweetlerden oluşmaktadır. Veri seti aslında cümlede duygu analizi gerçekleştirmek için oluşturulmuş bir veri setidir. Ancak içerisinde Türkçe hakaret ve küfür içeren tweetler bulunmaktadır. Küfür tespitini AVT çalışması olarak yapmak için tüm cümlelerde ki duygu durumu etiketleri kaldırılmıştır. Duygu durum etiketleri kaldırıldıktan sonra her sözcük IOB2 etiketleme yapısına göre tek tek etiketlenerek AVT çalışmasına uygun hale getirilmiştir.

Veri setine bu adresten[8] ulaşılabilir. Veri setinde toplamda 15,110 cümle bulunmaktadır. Tablo 6.1'de, veri setindeki bazı örnek cümle verilmiştir. Cümle içerisinde geçen küfürlü kelimeleri tespit etmek amacıyla kelimeler IOB2 tagging yöntemine göre etiketlenmiştir. Tablo 6.2'de etiketlenmiş örnek cümleler gösterilmiştir. Ayrıca, cümleler içerisindeki kelimeleri etiketlemek için kullanılan küfürlü kelimelere de bu adresten[9] ulaşılabilir.

[8] https://github.com/ezgisubasi/turkish-tweets-sentiment-analysis/tree/main/data

[9] https://github.com/d35k/Turkish-Swear-Words/blob/master/swears.txt

Tablo 6.1 Veriden alınmış örnek tweetler

Örnek Cümle
Şerefsizlik, sözde sanatçıların vazgeçemediği bir değerdir
Kendisi de bilmiyordur çünkü beyinsiz
En uzun yolculuklar bile, tek bir adımla başlar. Geleceğin mimarları gençlerimiz için atılan ilk adımları var gücümüzle desteklemeye devam ediyoruz.
Merhaba, konuyla ilgili yardımcı olmak isteriz. İrtibat numaranızı direkt mesaj olarak bize iletir misiniz?
Çok güzel bir çalışma olduğuna inanıyoruz. Yazımızda iki bölümün de avantaj ve dezavantajlarını değerlendirmeye çalıştık. Unutmayınız, bu bir üstünlük konusu değil. İşimiz ve amacımız bilim.

Tablo 6.2 Etiketlenmiş örnek cümle

Kelime	Etiket
şerefsizlik	B-Profanity
sözde	O
sanatçıların	O
vazgeçemediği	O
bir	O
değerdir	O

Veri ön hazırlık aşamasında aşağıdaki işlemler yapıldı:

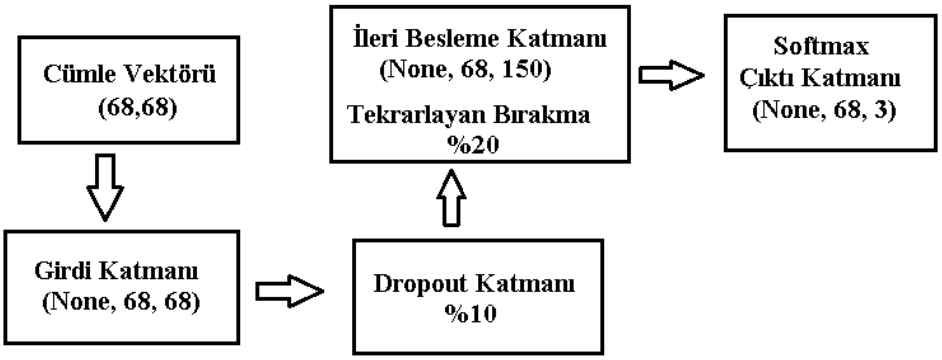
- Cümle içerisindeki tüm noktalama işaretleri kaldırıldı,
- Tüm harfler küçük hale getirildi,
- "@" işareti ile başlayan ve twitterda kişi etiketleme için kullanılan bahsetmeler kaldırıldı.

Veri etiketleme aşamasında aşağıdaki işlemler yapıldı:

- Ön hazırlıktan geçen kelimeler jetonlaştırılarak ayrı jetonlar haline getirildi,
- Her jeton ve sonrasında gelen kelime bir küfür veya hakaret içeren söze eşit mi diye kontrol edilerek eldeki küfür sözlüğüyle jetonlar eşlendi. Eşleşme işlemi IOB2 etiketleme sistemine göre gerçekleştirildi.
- Etiketlenmeyen veya sözlükte bulunmayan kelimeler manuel olarak kontrol edildi ve eşlenme sağlandı.

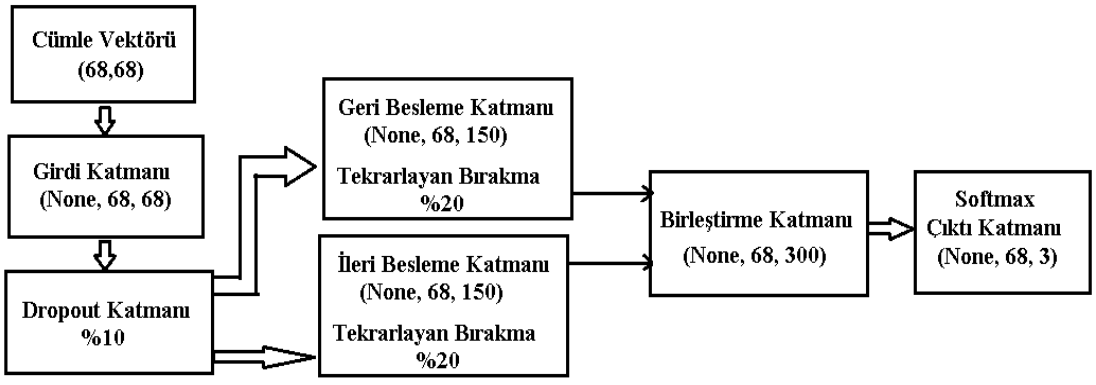
6.2 Model Mimarisi

Model mimarisi oluşturulurken tüm cümleler en uzun cümle ile aynı boyda vektörlere dönüştürüldü. Vektör boylarını eşitlemek için vektörler sıfır ile dolduruldu. Cümle içerisinde ki kelimeler vektörleştirilirken kategorik vektörleştirme uygulandı. Her benzersiz sözcük eşsiz bir tek sayı ile temsil edildi. Oluşturulan cümle vektörleri modellerin giriş katmanına en uzun cümlenin botu kadar yeni 68 sayı ile verilmiştir.



Şekil 6.1 Tek yönlü modellerin mimarisi

Girdi katmanından sonra ezberlemeyi engellemek için yüzde 10'luk bir bırakma (dropout) katmanı ilave edilmiştir. Bu katmandan sonra iki farklı durum söz konusudur. Bir tanesi ileri yönde beslemesi olan modeller, bir tanesi de iki yönde beslemeye sahip olan çift yönlü modellerdir. İleri yönde beslemeli sinir ağı modelleri Şekil 6.1'de gösterilmiştir. Çift yönde beslemeli sinir ağı modelleri ise Şekil 6.2'de gösterilmiştir.

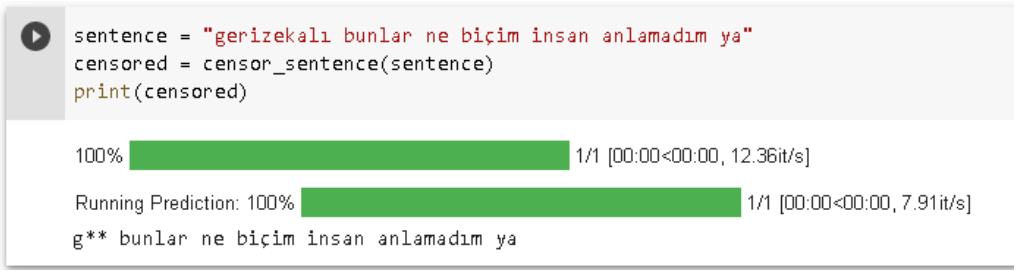


Şekil 6.2 Çift yönlü modellerin mimarisi

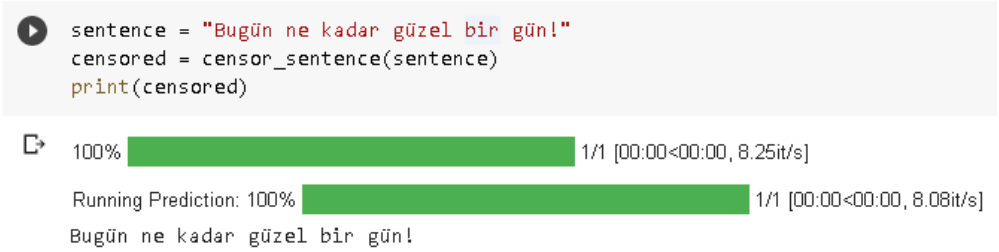
LSTM, RNN ve GRU gibi ağlar hafıza yapısına sahip oldukları için ezberlemeyi önlemek için sinir ağı katmanına %20 tekrarlayan bırakma (recurrent dropout) eklenmiştir. Çift Yönlü beslemeli sinir ağı katmanından sonra da birleştirme katmanı eklenmiştir. İleri ve geri yönde ki sinir ağları 150 adet nörondan oluşturulmuştur. Bu sayede birleştirme katmanı tek yönlü katmanlarına iki katına, 300 adet nörona çıkarılmıştır. Daha sonra çıktı katmanına softmax aktivasyon fonksiyonu eklenmiştir. Bu katman ise 68 satır 3 sütundan oluşan tahmin vektörünü döndürmektedir.

6.3 Modelleri Eğitilmesi

Kullanılan modellerin eğitimi, Google Colab üzerinde sağlanan GPU (Graphical processor Unit) Tesla K80 isimli sanal makine ile gerçekleştirildi. Sanal makine, 33 MB CPU (Central Processing Unit) ve 1.4 GB GPU'ya sahiptir. Eğitim ve test setleri %80'e %20 olacak şekilde ayarlanmıştır. Şekil 6.3 ve 6.4'te, örnek model çıktıları gösterilmiştir. Eğitilmiş modele verilen cümleler önce tokenleştirilmiştir. Sonra da modelin tahminleme yapması sağlanmış ve yapılan tahminlere göre sansürlenmiş cümle şekline döndürülmüştür.



Şekil 6.3 Hakaret içeren cümle ve sansürlenmiş model tahmin çıktısı



Şekil 6.4 Normal bir cümle ve model tahmin çıktısı

Tablo 6.3 ve 6.9 arasında ki tablolarda modellerin çalışma sonuçları gösterilmiştir. RNN, çift yönlü RNN, GRU, çift yönlü GRU, LSTM, çift yönlü LSTM modelleri hem eğitimde hem de test aşamasında %99'a yakın doğruluk oranı sergilemiştir. BERT modeli ise eğitim aşamasında %99 civarında eğitim başarısı gösterirken, test başarı %95 civarında gözlemlenmektedir. Bert modeli özelinde test başarının eğitim başarısına göre düşüş nedenlerinden birisi eğitim yapılırken seçilen epoch sayısının azlığı olarak düşünülmektedir. Ayrıca iki model eğitim süreleri açısından karşılaştırmak gerekirse; hafıza temelli modeller her adımı yaklaşık 840 milisaniye civarında gerçekleştirirken, BERT modeli her adımı 2,2 saniye civarında tamamlamıştır. Hafıza temelli modellerin yaklaşık olarak 3 kat daha hızlı olduğu görülmektedir.

Tablo 6.3 RNN modelinin çalışma performansı

RNN	**Loss**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Train	0,0023	0,9994	0,9994	0,9994	0,9994
Test	0,0065	0,99868	0,9987	0,9987	0,99867

Tablo 6.4 Bidirectional RNN modelinin çalışma performansı

Bi-RNN	**Loss**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Train	0,0012	0, 9996	0, 9996	0, 9997	0,9996
Test	0,0043	0,99914	0,9991	0, 9991	0,9996

Tablo 6.5 GRU modelinin çalışma performansı

GRU	**Loss**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Train	0,0017	0,9996	0,9996	0,9996	0,9996
Test	0,0039	0,99922	0,99926	0,99931	0,99922

Tablo 6.6 Bidirectional GRU modelinin çalışma performansı

Bi-GRU	**Loss**	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Train	0,0011	0,9996	0,9996	0,9996	0,9996
Test	0,0045	0,999042	0,99920	0,999052	0,999033

Tablo 6.7 LSTM modelinin çalışma performansı

LSTM	Loss	Accuracy	F1-Score	Precision	Recall
Train	0,0022	0,9994	0,9994	0,9994	0,9994
Test	0,0071	0,99877	0,99920	0,99886	0,99918

Tablo 6.8 Bidirectional LSTM modelinin çalışma performansı

Bi-LSTM	Loss	Accuracy	F1-Score	Precision	Recall
Train	0,0010	0,9997	0,9997	0,9997	0,9996
Test	0,00363	0,999192	0,99865	0,99921	0,99844

Tablo 6.9 BERT modelinin çalışma performansı

BERT	Loss	Accuracy	F1-Score	Precision	Recall
Train	0,002642	0,9884	0,988485	0,9876256	0,989347
Test	0,031639	0,9523	0,933693	0,922315	0,945355

6.4 Sonuç

Bu çalışmada Türkçe tweetler kullanılarak hakaret veya küfür içeren kelime ve kelime öbeklerinin tespiti problemi ele alındı. Bu probleme, NER problemi olarak yaklaşıldı ve çözümü için derin öğrenme tabanlı RNN, çift yönlü RNN, GRU, çift yönlü GRU, LSTM, çift yönlü LSTM ve BERT modelleri kullanıldı. Modellerin karşılaştırmalı sonuçları analiz edildi. Hafıza temelli modeller hem eğitimde hem de test aşamasında %99'a yakın doğruluk oranı sergiledi. BERT modeli ise eğitim aşamasında %99 civarında eğitim başarısı gösterirken, test başarı %95 civarında olduğu gözlemlendi. Çalışma hızı açısından, hafıza temelli modeller BERT modelinden yaklaşık olarak 3 kat daha hızlı olduğu görüldü.

Çalışmada kullanılan veri seti, genel olarak sosyal medya içeriklerinden oluşmaktadır. Fakat bu çalışma farklı uygulama alanları, yani dergi, kitap veya gazete gibi diğer basın kuruluşlarından veri alarak geliştirilebilir. Ayrıca, ileri çalışmalar olarak, kullanılan metnin yanında diğer ek özelliklerinin de dikkate alınarak daha yüksek performanslı algoritmaların geliştirilmesi düşünülebilir.

KAYNAKLAR

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A. ve Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11), e00938.

Adafre, S. F. ve de Rijke, M. (2005, Haziran). Feature engineering and post-processing for temporal expression recognition using conditional random fields. *Proceedings of the ACL workshop on feature engineering for machine learning in natural language processing* (9-16).

Adam, E. E. B. (2020). Deep learning based NLP techniques in text to speech synthesis for communication recognition. *Journal Of Soft Computing Paradigm* (JSCP), *2*(04), 209-215.

Alaparthi, S. ve Mishra, M. (2021). BERT: A sentiment analysis odyssey. *Journal of Marketing Analytics*, 9(2), 118-126.

Bahdanau, D., Cho, K. ve Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473.*

Bao, H., Dong, L., Wei, F., Wang, W., Yang, N., Liu, X. ve Hon, H. W. (2020, Kasım). Unilmv2: Pseudo-masked language models for unified language model pre-training. *International Conference on Machine Learning* (642-652). PMLR.

Berglund, M., Raiko, T., Honkala, M., Kärkkäinen, L., Vetek, A. ve Karhunen, J. T. (2015). Bidirectional recurrent neural networks as generative models. *Advances in neural information processing systems*, 28.

Brants, T. (2000). TnT-a statistical part-of-speech tagger. *arXiv preprint cs/0003055.*

Brill, E., (1992). A simple rule-based part of speech tagger. *3rd Conference on Applied Natural Language Processing* (ANLC), Trento, Italy,152–155

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational linguistics, 21*(4), 543-565.

Brunette, E. S., Flemmer, R. C. ve Flemmer, C. L. (2009, Şubat). A review of artificial intelligence. *2009 4th International Conference on Autonomous Robots and Agents* (385-392). Ieee.

Bowden, K. K., Wu, J., Oraby, S., Misra, A. ve Walker, M. (2018). SlugNERDS: A named entity recognition tool for open domain dialogue systems. *arXiv preprint arXiv:1805.03784*.

Cho, K., Van Merriënboer, B., Bahdanau, D. ve Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Chung, J., Gulcehre, C., Cho, K. ve Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Çelik, A. ve Yıldırım, B. (2020, Kasım). Turkish profanity detection enhanced by artificial intelligence. *2020 28th Signal Processing and Communications Applications Conference (SIU)* (1-4). IEEE.

Deepak, G., Teja, V. ve Santhanavijayan, A. (2020). A novel firefly driven scheme for resume parsing and matching based on entity linking paradigm. *Journal of Discrete Mathematical Sciences and Cryptography, 23*(1), 157-165.

Deshmukh, R. D. ve Kiwelekar, A. (2020, Mart). Deep learning techniques for part of speech tagging by natural language processing. *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)* (76-81). IEEE.

Devlin, J., Chang, M. W., Lee, K. ve Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dos Santos, C. ve Zadrozny, B. (2014, Haziran). Learning character-level representations for part-of-speech tagging. *International Conference on Machine Learning* (1818-1826). PMLR.

Grishman, R. (1995). *The NYU System for MUC-6 or Where's the Syntax?*. Newyork University NY Deptartment Of Computer Scıence.

Guo, J., Xu, G., Cheng, X. ve Li, H. (2009, Temmuz). Named entity recognition in query. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval* (267-274).

Güneş, A. ve Tantuğ, A. C. (2018, Mayıs). Turkish named entity recognition with deep learning. *2018 26th Signal Processing and Communications Applications Conference (SIU)* (1-4). IEEE.

Han, J., Sun, A., Cong, G., Zhao, W. X., Ji, Z. ve Phan, M. C. (2017). Linking fine-grained locations in user comments. *IEEE Transactions on Knowledge and Data Engineering*, 30(1), 59-72.

He, J., Zhao, L., Yang, H., Zhang, M. ve Li, W. (2019). HSI-BERT: Hyperspectral image classification using the bidirectional encoder representation from transformers. *IEEE Transactions on Geoscience and Remote Sensing*, 58(1), 165-178.

Heck, J. C. ve Salem, F. M. (2017, Ağustos). Simplified minimal gated unit variations for recurrent neural networks. *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)* (1593-1596). IEEE.

Hochreiter, S. ve Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

Jones, K. S. (1994). Natural language processing: a historical review. *Current issues in computational linguistics: in honour of Don Walker*, 3-16.

Joseph, S. R., Hlomani, H., Letsholo, K., Kaniwa, F. ve Sedimo, K. (2016). Natural language processing: A review. *International Journal of Research in Engineering and Applied Sciences*, 6(3), 207-210.

Jwa, H., Oh, D., Park, K., Kang, J. M. ve Lim, H. (2019). exbake: Automatic fake news detection model based on bidirectional encoder representations from transformers (bert). *Applied Sciences*, 9(19), 4062.

Krause, S., Li, H., Uszkoreit, H. ve Xu, F. (2012, Kasım). Large-scale learning of relation-extraction rules with distant supervision from the web. *International Semantic Web Conference* (263-278). Springer, Berlin, Heidelberg.

Krupka, G. (1995). SRA: Description of the SRA system as used for MUC-6. *Sixth Message Understanding Conference (MUC-6): Kolombiya Maryland'de düzenlenen konferans tutanakları, 6-8, 1995.*

Laboreiro, G. ve Oliveira, E. (2014, Ekim). What we can learn from looking at profanity. *International Conference on Computational Processing of the Portuguese Language* (108-113). Springer, Cham.

Lafferty J., McCallum A., Pereira F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the eighteenth international conference on machine learning*, ICML, 1, 282–289, 2001.

LeCun, Y., Bengio, Y. ve Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436-444

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E. ve Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems* (396-404).

LeCun, Y., Bottou, L., Bengio, Y. ve Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

Lee, H. S., Lee, H. R., Park, J. U. ve Han, Y. S. (2018). An abusive text detection system based on enhanced abusive and non-abusive word lists. *Decision Support Systems*, 113, 22-31.

Lin, J., Nogueira, R. ve Yates, A. (2021). Pretrained transformers for text ranking: Bert and beyond. *Synthesis Lectures on Human Language Technologies*, 14(4), 1-325.

Liu, Y. ve Lapata, M. (2019). Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Luo, F., Xiao, H. ve Chang, W. (2011, Ekim). Product named entity recognition using conditional random fields. 2011 *Fourth international conference on business intelligence and financial engineering* (86-89). IEEE.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S. ve McClosky, D. (2014, Haziran). The Stanford CoreNLP natural language processing toolkit. *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations* (55-60).

McCallum, A., Freitag, D. ve Pereira, F. C. (2000, Haziran). Maximum entropy Markov models for information extraction and segmentation. *Icml* (17, No. 2000, 591-598).

Meng, X., Wei, F., Liu, X., Zhou, M., Li, S. ve Wang, H. (2012, Ağustos). Entity-centric topic-oriented opinion summarization in twitter. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (379-387).

Mikolov T., Chen K., Corrado G. ve Dean J., Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781, 2013.*

Mikolov T., Sutskever I., Chen K., Corrado G.S. ve Dean J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 3111–3119, 2013.

Miller, D. (2019). Leveraging BERT for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.

Nasiboglu, R. ve Gencer, M. (2021). Comparıson of spacy and stanford libraries'pre-trained deep learning models for named entity recognition. *Journal of Modern Technology and Engineering*, 6(2), 104-111.

Orellana, M., Trujillo, A., Lima, J. F., Acosta, M. I. ve Peña, M. (2020). An evaluation methodology of named entities recognition in spanish language: *ECU 911 case study*.

Özkaya, S. ve Diri, B. (2011, Nisan). Named entity recognition by conditional random fields from Turkish informal texts. *2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU)* (662-665). IEEE.

Pawar, S., Srivastava, R. ve Palshikar, G. K. (2012, Ocak). Automatic gazette creation for named entity recognition and application to resume processing. *Proceedings of the 5th ACM COMPUTE Conference: Intelligent & scalable system technologies* (1-7).

Pinto, D., McCallum, A., Wei, X. ve Croft, W. B. (2003, Temmuz). Table extraction using conditional random fields. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (235-242).

Pradhan S., Moschitti A., Xue N., Ng H.T., Björkelund A., Uryupina O., Zhang Y. ve Zhong Z. (2013). Towards robust linguistic analysis using ontonotes. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning* (143-152).

Ramshaw, L. A., & Marcus, M. P. (1999). Text chunking using transformation-based learning. *Natural language processing using very large corpora* (157-176). Springer, Dordrecht.

Ratadiya, P. ve Mishra, D. (2019, Kasım). An attention ensemble based approach for multilabel profanity detection. *2019 International Conference on Data Mining Workshops (ICDMW)* (544-550). IEEE.

Rau L.F., "Extracting company names from text," in Artificial Intelligence Applications, 1991. *Proceedings., Seventh IEEE Conference on Artificial Intelligence Application*, 1991, 1, 29-32: IEEE.

Ravanelli, M., Brakel, P., Omologo, M. ve Bengio, Y. (2018). Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2), 92-102.

Russell, S., ve Norvig, P. (2005). *AI a modern approach. Learning*, 2(3), 4.

Sang, E. F. ve De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Sang, E. F. ve Veenstra, J. (1999). Representing text chunks. *arXiv preprint cs/9907006*.

Sanh, V., Debut, L., Chaumond, J. ve Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sarı, Ö. C. ve Aktaş, Ö. (2018), Türkçe ders metinleri için özelleştirilmiş bir varlık ismi tanıma yapısı. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 11(2), 52-68.

Satheesh, K., Jahnavi, A., Iswarya, L., Ayesha, K., Bhanusekhar, G. ve Hanisha, K. (2020). Resume ranking based on job description using SpaCy NER model. *International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395–0056*, *7*(05).

Schiersch, M., Mironova, V., Schmitt, M., Thomas, P., Gabryszak, A. ve Hennig, L. (2020). A german corpus for fine-grained named entity recognition and relation extraction of traffic and industry events. *arXiv preprint arXiv:2004.03283.*

Schmid, H. (1994). Part-of-speech tagging with neural networks. *arXiv preprint cmp-lg/9410018.*

Schmid, H. (1999). Improvements in part-of-speech tagging with an application to German. *Natural language processing using very large corpora* (13-25). Springer, Dordrecht.

Schuster, M. ve Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11), 2673-2681.

Sha, F. ve Pereira, F. (2003). Shallow parsing with conditional random fields. *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (213-220).

Shelar, H., Kaur, G., Heda, N. ve Agrawal, P. (2020). Named entity recognition approaches and their comparison for custom NER model. *Science & Technology Libraries*, 39(3), 324-337.

Shen, Y., Yun, H., Lipton, Z. C., Kronrod, Y. ve Anandkumar, A. (2017). Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928.*

Sienčnik, S. K. (2015, Mayıs). Adapting word2vec to named entity recognition. *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)* (239-243).

Sood, S. O., Antin, J. ve Churchill, E. (2012, Mart). Using crowdsourcing to improve profanity detection. *2012 AAAI Spring Symposium Series*.

Sood, S., Antin, J. ve Churchill, E. (2012, Mayıs). Profanity use in online communities. *Proceedings of the SIGCHI conference on human factors in computing systems* (1481-1490).

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S. ve Tsujii, J. I. (2012, Nisan). BRAT: A web-based tool for NLP-assisted text annotation. *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics* (102-107).

Subramaniam, L. V., Faruquie, T. A., Ikbal, S., Godbole, S. ve Mohania, M. K. (2009, Mart). Business intelligence from voice of customer. *2009 IEEE 25th International Conference on Data Engineering* (1391-1402). IEEE.

Su, H. P., Huang, Z. J., Chang, H. T. ve Lin, C. J. (2017, Ağustos). Rephrasing profanity in chinese text. *Proceedings of the First Workshop on Abusive Language Online* (18-24)

Sukthanker, R., Poria, S., Cambria, E. ve Thirunavukarasu, R. (2020). Anaphora and coreference resolution: A review. *Information Fusion, 59*, 139-162.

Tao, Q., Liu, F., Li, Y. ve Sidorov, D. (2019). Air pollution forecasting using a deep learning model based on 1D convnets and bidirectional GRU. *IEEE access*, 7, 76690-76698.

Tenney, I., Das, D. ve Pavlick, E. (2019). BERT rediscovers the classical NLP pipeline. *arXiv preprint arXiv:1905.05950.*

Turing, A. M., ve Haugeland, J. (1950). *Computing machinery and intelligence. The Turing Test: Verbal Behavior as the Hallmark of Intelligence*, 29-56.

Tutubalina, E. ve Nikolenko, S. (2017). Combination of deep recurrent neural networks and conditional random fields for extracting adverse drug reactions from user reviews. *Journal of healthcare engineering*, 2017.

Wallach, H.M., "Conditional Random Fields: An Introduction", University of Pennsylvania *CIS Technical Report*, 2004.

Wang, H., Ma, C. ve Zhou, L. (2009, Aralık). A brief review of machine learning and its application. *2009 international conference on information engineering and computer science* (1-4). IEEE.

Wang, Z., Cui, X., Gao, L., Yin, Q., Ke, L. ve Zhang, S. (2016). A hybrid model of sentimental entity recognition on mobile social media. *EURASIP Journal on Wireless Communications and Networking*, 2016(1), 1-12.

Yadav, V. ve Bethard, S. (2019). A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470.*

Yang, J., Liang, S. ve Zhang, Y. (2018). Design challenges and misconceptions in neural sequence labeling. *arXiv preprint arXiv:1806.04470.*

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K. ve Lin, J. (2019). End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*.

Yi, M., Lim, M., Ko, H. ve Shin, J. (2021). Method of Profanity Detection Using Word Embedding and LSTM. *Mobile Information Systems*, 2021.

Yin, W., Kann, K., Yu, M. ve Schütze, H. (2017). Comparative study of CNN and RNN for natural language processing. *arXiv preprint arXiv:1702.01923*.

Yoo, S. ve Jeong, O. (2020). An intelligent chatbot utilizing BERT model and knowledge graph. *Journal of Society for e-Business Studies*, 24(3).

Zhou, Q. ve Wu, H. (2018, Ekim). NLP at IEST 2018: BiLSTM-attention and LSTM-attention via soft voting in emotion classification. *Proceedings of the 9th workshop on computational approaches to subjectivity, sentiment and social media analysis* (189-194).