BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ TEZLİ YÜKSEK LİSANS PROGRAMI

GÖRÜNTÜ ARŞİVLERİ İÇİN DERİN SİNİR AĞLARI KULLANILARAK HECELERE DAYALI GÖRÜNTÜ ALT YAZILAMA MODELİ

HAZIRLAYAN

YAĞMUR KAYA

YÜKSEK LİSANS TEZİ

ANKARA - 2023

BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ TEZLİ YÜKSEK LİSANS PROGRAMI

GÖRÜNTÜ ARŞİVLERİ İÇİN DERİN SİNİR AĞLARI KULLANILARAK HECELERE DAYALI GÖRÜNTÜ ALT YAZILAMA MODELİ

HAZIRLAYAN

YAĞMUR KAYA

YÜKSEK LİSANS TEZİ

TEZ DANIŞMANI

DR. ÖĞR. ÜYESİ TÜLİN ERÇELEBİ AYYILDIZ

ANKARA - 2023

BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı çerçevesinde Yağmur KAYA tarafından hazırlanan bu çalışma, aşağıdaki jüri tarafından Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Savunma Tarihi: 11 /08/2023

Tez Adı: Görüntü Arşivleri İçin Derin Sinir Ağları Kullanılarak Hecelere Dayalı Görüntü Alt Yazılama Modeli

Tez Jüri Üyeleri **İmza**

Doç. Dr. Emre SÜMER, Başkent Üniversitesi

Dr. Öğr. Üyesi Tülin ERÇELEBİ AYYILDIZ, Başkent Üniversitesi

Dr. Öğr. Üyesi Damla TOPALLI, Atılım Üniversitesi

ONAY

Prof. Dr. Faruk ELALDI
Fen Bilimleri Enstitüsü Müdürü

Tarih : … / … / 2023

BAŞKENT ÜNİVERSİTESİ

FEN BİLİMLER ENSTİTÜSÜ

YÜKSEK LİSANS TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

Tarih: 17 / 08 / 2023

Öğrencinin Adı, Soyadı : Yağmur KAYA

Öğrencinin Numarası : 22010617

Anabilim Dalı : Bilgisayar Mühendisliği

Programı : Bilgisayar Mühendisliği

Danışmanın Unvanı/Adı, Soyadı : Dr. Öğr. Üyesi Tülin Erçelebi Ayyıldız

Tez Başlığı : Görüntü Arşivleri İçin Derin Sinir Ağları Kullanılarak Hecelere Dayalı Görüntü Alt Yazılama Modeli

Yukarıda başlığı belirtilen Yüksek Lisans tez çalışmamın; Giriş, Ana Bölümler ve Sonuç Bölümünden oluşan, toplam 51 sayfalık kısmına ilişkin, 17 / 08 / 2023 tarihinde tez danışmanım tarafından Turnitin adlı intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezimin benzerlik oranı % 4'tür. Uygulanan filtrelemeler:

1. Kaynakça hariç
2. Alıntılar hariç
3. Beş (5) kelimeden daha az örtüşme içeren metin kısımları hariç

"Başkent Üniversitesi Enstitüleri Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Usul ve Esaslarını" inceledim ve bu uygulama esaslarında belirtilen azami benzerlik oranlarına tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Öğrenci İmzası:

ONAY

Tarih: 17 / 08 / 2023

Dr. Öğr. Üyesi Tülin ERÇELEBİ AYYILDIZ

Bu tezi sevgili dedem İhsan YALÇINER'e ithaf ediyorum.

TEŞEKKÜR

Yüksek lişans tez çalışmam süresince bana her türlü desteği veren, her sorunumda anlayışla çözüm bulan ve tecrübelerinden faydalanma imkânı sunan değerli hocam Dr. Öğr. Üyesi Tülin ERÇELEBİ AYYILDIZ'a,

Eğitim hayatımın başından itibaren desteklerini esirgemeyen ve her koşulda yanımda hissettiğim annem Zuhal KAYA, teyzem Müşfika YALÇINER, babam İdris KAYA, kardeşlerim Aşina SEVEN - Eren SEVEN, sevgili yeğenim Mehmet Yekta SEVEN'e,

Yüksek lisans eğitimim süresince ders aldığım tüm hocalarıma ve birlikte çalışmaktan keyif aldığım dönem arkadaşlarıma,

Bilgi ve tecrübelerinden her zaman faydalandığım arkadaşım Ersin AKSOY'a,

Tez çalışmamın veri etiketleme kısmında destek olan tüm çalışma arkadaşlarıma ve verilerin derlenmesinde katkı sağlayan Ekin ALTUĞ'a,

Yüksek lisans çalışmam süresince desteklerini hissettiğim çalışma arkadaşlarım Mehmet Erinç BAŞAR, Tuba DURMAZ KAYA'ya,

Bu süreçte beni yalnız bırakmadıkları için en içten teşekkürlerimi sunarım.

ÖZET

Yağmur KAYA

GÖRÜNTÜ ARŞİVLERİ İÇİN DERİN SİNİR AĞLARI KULLANILARAK HECELERE DAYALI GÖRÜNTÜ ALT YAZILAMA MODELİ

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

2023

Bir görüntüye ait içeriğin insan benzeri doğal bir dil kullanılarak otomatik olarak tasvir edilmesi görüntü alt yazılama olarak adlandırılmaktadır. Bilgisayarlı görü ve doğal dil işleme tekniklerinin bir arada kullanıldığı görüntü alt yazılama alanında İngilizce için tasarlanan birçok model bulunmaktadır. Ancak Türkçenin sondan eklemeli yapısından dolayı bu modellerin doğrudan Türkçeye uyarlanması mümkün değildir. Bu tez çalışmasında dilin yapısını daha iyi anlayabilmek için hecelerden faydalanılmış ve heceler üzerinde çalışan bir görüntü alt yazılama modeli önerilmiştir. Önerilen model encoder-decoder mimarisine sahip olup CNN ve LSTM birlikte kullanılmıştır. Mevcutta Türkçe dili için önerilen sözcük, alt kelime/kök tabanlı modeller ile önerdiğimiz hecelere dayalı modelin başarımını kıyaslamak için üç ayrı veri kümesi kullanılmıştır. Bunlardan ilk ikisi araştırmacıların açık erişimine sunulan Flickr8k ve Flickr30k bir diğeri ise kendi oluşturduğumuz veri kümesidir. Önerilen yaklaşımın performansı BLEU ölçüm metriği kullanılarak değerlendirilmiş her üç veri kümesinde de hecelere dayalı modelin diğer iki modele göre daha başarılı olduğu sonucuna ulaşılmıştır. Bilindiği kadarıyla, hecelere dayalı görüntü alt yazılama alanında yapılan herhangi bir çalışma bulunmamaktadır, bu sebeple sunulan çalışma bu konuda yapılan ilk çalışma olması sebebiyle önemlidir.

ANAHTAR KELİMELER: Görüntü alt yazılama, örüntü tanıma, görüntü işleme, doğal dil işleme, kodlayıcı-kod çözücü mimari, derin sinir ağları

ABSTRACT

Yağmur KAYA

SYLLABLE-BASED IMAGE CAPTIONING MODEL BASED ON DEEP NEURAL NETWORKS FOR IMAGE ARCHIVES

Başkent University Institute of Science and Engineering

Department of Computer Engineering

2023

Image captioning also known as automatic image description, refers to the process of automatically describing the content of an image by using natural language. In the field of image captioning, there are many models designed for English where computer vision and natural language processing techniques are combined. Direct adaptation of these models to Turkish is not possible due to the agglutinative structure of the Turkish language. In our study, we propose a syllable-based image captioning model to better understand the structure of the language. The proposed model follows an encoder-decoder architecture, utilizing both CNN and LSTM. Three seperate datasets were used to compare the performance of the proposed syllable-based model with word-level and baseword/subword-level models. The first and second datasets are the Flickr8k and Flickr30k dataset, which is publicly accessible for researchers, while the other one is a dataset that we created. The performance of the proposed approach was evaluated by using the BLEU metric, and we found that the syllable-based model outperformed the other two models in all datasets. To the best of our knowledge, there is no existing work on syllable-based image captioning, making our study significant as the first work.

KEYWORDS: Image captioning, pattern recognition, image processing, natural language processing, encoder-decoder architecture, deep neural network.

.

ÖNSÖZ

Kültürel miras olan görüntü arşivlerine hızlı ve doğru bir şekilde erişim sağlanabilmesi için görüntüyü betimleyen alt yazılara ihtiyaç duyulmaktadır. Ancak gelişen teknoloji ile birlikte artan veri miktarına, manuel veri girişi yapılarak yetişilmesi mümkün görünmemektedir. Bu tez çalışmasında gelişen bu ihtiyacı güçlü makineler yardımıyla daha hızlı bir şekilde karşılayabilmek amacıyla derin öğrenme yöntemlerinden faydalanılarak Türkçe diline özel bir görüntü alt yazılama modeli geliştirilmiştir. Yapılan çalışmalar ve elde edilen sonuçlar yeterli eğitim verisi ve güçlü makineler kullanılarak alt yazılama işlemlerinin insan gücüne ihtiyaç duyulmadan hızlı bir şekilde yapılabileceğini göstermektedir.

İÇİNDEKİLER

TABLOLAR LİSTESİ

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

SİMGELER VE KISALTMALAR LİSTESİ

BLEU	Bilingual Evaluation Understudy: İki Dilli Değerlendirme Çalıştırması
BPE	Byte Pair Encoding: Bayt Çifti Kodlama
CBOW	Continuous Bag of Words: Sürekli Kelime Torbası
CNN	Convolutional Neural Networks: Evrişimli Sinir Ağları
CV	Computer Vision: Bilgisayarlı Görü
IDF	Inverse Document Frequency: Ters Belge Frekansı
LSTM	Long Short Term Memory: Uzun Kısa Süreli Bellek
MSCOCO	Microsoft Common Objects in Context: Microsoft Ortak Nesne Tanıma
NLP	Natural Language Processing: Doğal Dil İşleme
OOV	Out-of Vocabulary Word: Sözlük Dışı Kelime
ReLU	Rectified Lineer Unit: Doğrultulmuş Doğrusal Birim
RNN	Recurrent Neural Networks: Yinelemeli Sinir Ağları
TF	Term Frequency: Terim Frekansı
VGG	Visual Geometry Group: Görsel Geometri Grubu

1. GİRİŞ

Gelişen teknoloji ve buna paralel olarak artan görüntü sayısı, sayısal görüntü arşivciliğinde iki temel sorunu beraberinde getirmiştir. Çeşitli kaynaklardan elde edilen görüntülerin güvenli bir ortamda saklanması ve istenildiğinde hızlı bir şekilde aranıp sorgulanabilir olması en temel sorunlar olarak öne çıkmaktadır. Sayısal görüntü arşivleme sistemlerinde görüntüler saklanırken, yüksek kapasiteli diskler veya teyp kütüphanelerinin kullanılıyor olması güvenli ortamda saklama sorununu büyük ölçüde çözmektedir [1]. Ancak hızlı bir arama sorgulama için görüntüyü betimleyen alt yazılara ihtiyaç duyulmaktadır. Bu işlemin manuel olarak insan eliyle yürütülmesi uzun yıllar alacağından, görüntülerin insan diline yakın bir şekilde otomatik olarak tasvir edilmesi önemli bir ihtiyaç haline gelmiştir [2].

Bilgisayarlı görü ve doğal dil işleme tekniklerini bir arada barındıran görüntü alt yazılama problemleri, bu bütünleşik yapısından dolayı zorlu problemler arasında sayılmaktadır. Görüntü alt yazılama problemiyle ilgili yapılan çalışmalar çoğunlukla İngilizce üzerinde yoğunlaşmıştır ve son yıllarda yapılan birkaç çalışma dışında Türkçe üzerinde yeterince çalışma bulunmamaktadır. Türkçenin sondan eklemeli bir dil oluşu sebebiyle kelimeye eklenen her bir ek kelimenin anlamını değiştirmektedir. Dilin bu yapısı zor olan görüntü alt yazılama problemini Türkçe için daha da zor hale getirmektedir ve İngilizce için geliştirilen modeller üzerine farklı yaklaşımlar uygulanması gerekliliğini doğurmaktadır.

Görüntü alt yazılama sayısal bir ortamda arşivlenen görüntülerin efektif bir şekilde aranıp sorgulanabilmesi için anahtar cümleler üretmenin yanında, görme engelli bireyler için önündeki manzarayı tasvir etme, sosyal medyada üretilen içeriklerin analiz edilerek raporlanması dahil birçok alanda kullanılabilmektedir. Daha önce de belirtildiği gibi Türkçenin sondan eklemeli bir dil oluşu sebebiyle İngilizce dili için tasarlanan görüntü alt yazılama modelleri doğrudan kullanılamamakta, farklı yaklaşımlara ihtiyaç duyulmaktadır. Bu çalışmada, görüntüler üzerinde Türkçe alt yazılama yapılırken kelimenin en önemli temel bileşeni olan heceler kullanılarak Türkçeye özel bir görüntü alt yazılama modeli önerilmiştir. Önerilen modelin başarımı hem sektörden toplanan verilerle oluşturulan yeni bir veri kümesi üzerinde hem de araştırmacıların açık erişimine sunulan Flickr8K, Flickr30K veri kümeleri üzerinde ölçülmüştür. Önerilen yöntemin sözcük veya alt kelime/kök yaklaşımlarına kıyasla gözle görülür bir başarıma ulaştığı her üç veri kümesi üzerinde de gözlemlenmiştir.

Bu tez çalışması toplam 5 bölümden oluşmaktadır. Bölüm 2'de görüntü alt yazılama konusu ile ilgili literatür taraması sunulmuştur. Bölüm 3'te kodlayıcı-kod çözücü mimari (encoder-decoder architecture), evrişimli sinir ağları (Convolutional Neural Networks-CNN), yinelemeli sinir ağları (Recurrent Neural Networks-RNN), doğal dil işleme (Natural Language Processing-NLP), kelime temsil yöntemleri gibi görüntü alt yazılama modellerini anlamak için gerekli olan materyal ve yöntemler sunulmuştur. Bölüm 4'te veri kümesi, veri ön işleme aşamaları, bu tez çalışmasında önerilen modelin detayları ve elde edilen deneysel sonuçlar sunulmuştur. Bölüm 5'te sonuç ve öneriler başlığı altında elde edilen deney sonuçları değerlendirilmiş, bu tez çalışmasının literatüre katkısından bahsedilerek gelecekte yapılması planlanan çalışmalar aktarılmıştır.

2. LİTERATÜR ÇALIŞMASI

Görüntü alt yazılama bilgisayarlı görü ve doğal dil işleme tekniklerini birleştirerek, verilen görüntülere ait doğal dile yakın tanımlayıcı açıklamalar üreten, son yıllarda popülaritesini artırmış bir konudur. Görüntü alt yazılama için geçmişte iki temel yaklaşım yaygın olarak kullanılmıştır. Bu yaklaşımlardan ilki erişim tabanlı (retrieval based), ikincisi ise şablon tabanlı (template based) yaklaşımlardır.

Erişim tabanlı modellerde bir görüntüye ait yeni bir alt yazılama yapılırken, veri kümesindeki benzer görüntülere ait tanımlamalar getirilerek bu tanımlamalar mevcut görüntüye uyacak şekilde değiştirilmektedir. Ferhadi ve diğerlerine göre görüntü ve görüntüyü temsil eden alt yazı arasındaki ilişki tanımlanırken nesne, eylem ve sahne üçlüsü göz önünde bulundurulmalıdır [3]. Gupta ve Davis nesneleri ve eylemleri daha iyi tanımlayabilmek için nesne ve eylemler arası ilişkinin doğru tanımlanmasının önemli olduğunu vurgulamışlardır [4][5]. Bir görüntü ve açıklama arasındaki puanın yüksek oluşu benzerliğin de bir o kadar fazla olduğu anlamına gelmektedir. Erişim tabanlı modeller genellikle büyük boyutlu görüntü özellik vektörleriyle çalıştıklarından en büyük dezavantajları yüksek hesaplama maliyetleri oluşturmalarıdır. Şablon temelli modellerde ise, görüntüye ait alt yazılama yapılırken önceden tanımlanan özne, nesne, yüklem benzeri bir dizilimden oluşan cümle şablonları kullanılmaktadır. Bu modellerde anlamlı alt yazılar üretilebilmektedir ancak cümle oluşturulurken önceden belirlenen şablonun dil bilgisel kurallarına katı bir şekilde bağlı olunduğundan oluşan cümlelerin ifadesi insan yazısı kadar akıcı olamamaktadır [6].

Erişim tabanlı ve şablon temelli yaklaşımlar görüntü alt yazılama problemlerinde belirli bir başarımı yakalamıştır ancak alt yazı oluştururken, önceden tanımlı cümle yapılarının dışına çıkamamakta ve daha önce hiç karşılaşılmayan nesneleri tanıyamamaktadırlar. Derin öğrenme modellerinin her geçen gün daha da gelişmesiyle birlikte görüntü alt yazılamada kullanılan bu iki yaklaşım yerini derin öğrenme modelleri kullanılarak görüntü alt yazılama yaklaşımlarına bırakmıştır [7]. Görüntü alt yazılama problemleri, çoğunlukla makine çevirisi problemleriyle benzer şekilde değerlendirilmektedir. Bu sebeple makine çevirisinde kullanılan kodlayıcı ve kod çözücü (encoder-decoder) mimarisi görüntü alt yazılama problemlerinde de kullanılmaktadır. Görüntülere ait özniteliklerin çıkarılması için kodlayıcı aşamasında Evrişimli Sinir Ağları kullanılırken, kod çözücü kısmında ise Yinelemeli Sinir Ağları kullanılmaktadır. Xu ve

diğerleri ise kodlayıcı-kod çözücü mimarisi üzerine dikkat mekanizmasını dahil ederek bu çalışmadaki başarımı artırmışlardır [8]. Dikkat mekanizması dil modelinin her aşamasında bir sonraki kelimenin oluşturulması için hangi görüntü özelliklerine odaklanacağını göstermektedir. Bu özellik dil modelinin görüntü özelliklerinin farklı bölgelerine dikkatini yönlendirmesine olanak sağlamaktadır [9].

Görüntü alt yazılama alanındaki çalışmalar incelendiğinde Türkçe dili özelinde yapılan çalışma sayısının yeterli miktarda olmadığı görülmektedir. Türkçe görüntü alt yazılama ile ilgili üzerinde çalışma yapılan ilk veri kümesi Ünal ve diğerleri tarafından önerilen TasvirEt veri kümesidir [10]. İngilizce için hazırlanan ve araştırmacıların açık erişimine sunulan Flickr8K veri kümesindeki görüntülere Türkçe alt yazılar toplanarak oluşturulmuştur. TasvirEt veri kümesi toplam 8091 adet görüntü ve her görüntü için 2'şer alt yazıdan oluşmaktadır. Ünal ve diğerleri bu çalışmalarında görsel olarak en yakın görüntünün alt yazılarını transfer etme ve konsensüs alt yazılama yöntemi olmak üzere iki yöntem geliştirmiş ve yaptıkları deneyler sonucu konsensüs yönteminin daha başarılı olduğunu ortaya koymuşlardır [10]. Türkçe için önerilen bir diğer veri kümesi, Samet ve diğerleri tarafından MSCOCO veri kümesindeki görüntülere ait açıklamaların Google çeviri yardımıyla otomatik olarak çevrilmesi sonucu elde edilmiştir [11]. Veri kümesi boyutunun TasvirEt'e kıyasla büyük olması başarımı artıran bir faktör olarak görünse de alt yazı elde edilirken otomatik tercüme kullanılması dilin yapısının tam olarak yansıtılamamasına ve semantik açıdan gürültülü veriler nedeniyle model başarımlarının düşük olmasına neden olmaktadır. Kuyu ve diğerleri Türkçenin dil yapısının karakteristik özelliklerini dikkate alarak alt yazılama yapmanın daha başarılı olacağını savunmuş ve kelimeler yerine alt sözcükler kullanan bir alt yazılama modeli geliştirmişlerdir [12]. Alt sözcük modelini uygularken eğitim kümesindeki kelimeler Byte Pair Encoding (BPE) [13] algoritması ile alt sözcüklere ayrıştırılmış ve bu alt sözcüklere dayalı sözlük temsilleri kullanılmıştır. MSCOCO ve Flickr30K veri kümelerine ait alt yazıların otomatik çevirisi yapılarak eğitim verisi olarak kullanılmış ve önerilen alt sözcük modelinin başarılı sonuçlar verdiği gösterilmiştir. Stefanini ve diğerleri alt sözcük tabanlı yöntemler ile literatürde "seyrek kelime" olarak geçen nadir kelimelerin bulunamaması probleminin çözülebileceğini belirtmişlerdir [14]. Videoların Türkçe alt yazılanması konusunda yapılan ilk çalışma Çıtamak ve diğerlerine aittir [15]. Bu çalışmada MSVD veri kümesinin bir kısmı alınarak otomatik çevirisi yapılmış ve oluşturulan bu veri kümesi (MSV-Türkçe) üzerinde derin sinir ağlarına dayalı yöntemler denenerek modellerin başarımı kıyaslanmıştır. Elde edilen sonuçların belirli bir başarıma ulaştığı ancak geliştirmelere açık olduğu belirtilmektedir.

Türkçe video alt yazılama konusundaki bu çalışma kendi çalışmam ile aynı tip veri kümesi üzerinde çalışmasa da probleme yaklaşım şekli ve kullanılan yöntemler yol gösterici olmuştur.

Literatürde görüntü alt yazılama problemiyle ilgili çok sayıda çalışma bulunmasına rağmen, çalışılan modeller çoğunlukla tek bir dil üzerine yoğunlaşmıştır ve İngilizce üzerinde yüksek başarımlar elde edilmiştir. Son yıllarda Türkçe üzerinde de yapılan çalışmalar olmasına karşın, bu çalışmaların sayısı bir hayli azdır. Türkçe gibi sondan eklemeli dillerde kelimeler aldıkları her bir ek ile yeni anlamlar kazanabilmektedir. Dilin yapısının daha iyi modellenebilmesi için kelimenin kök, çekim eki ve ekler gibi morfolojik özelliklerinin anlaşılmasına ihtiyaç duyulmaktadır ve heceler bu konuda yüksek katkı sağlamaktadır. Türkçe dili özelinde düşünüldüğünde; kelimelere eklenen ekler içerisinde tekillik/çoğulluk, olumluluk/olumsuzluk, sahiplik gibi birçok anlam barındırmaktadır, bu bilgilerin kaybedilmeden saklanıyor ve öğrenmeye dahil ediliyor oluşu üretilen alt yazıların insan diline yakın bir yapıda olmasına büyük katkı sağlamaktadır. Korpus boyutunun azaltılarak hesaplama maliyetinin düşürülmesi amacıyla kullanılan, alt kelime/kök modellerinde hecelerin barındırdığı değerli bilgiler kullanılmamaktadır. Bu tez çalışması bilgi kaybını en azda tutarak alt yazılama yapmayı hedeflemektedir ve bu sebeple diğer çalışmalardan ayrışmaktadır. Bu amaçla, bu tez çalışmasında hecelere dayalı bir görüntü alt yazılama modeli tasarlanmış, başarımı sözcük ve alt kelime/kök modelleriyle kıyaslanmış ve daha yüksek bir başarım elde edilmiştir. Bilindiği kadarıyla, literatürde heceler üzerinde çalışılan bir görüntü alt yazılama modeli bulunmamaktadır ve bu nedenle bu yönde yapılan ilk çalışmadır.

3. MATERYAL VE YÖNTEMLER

Bu bölümde, çalışmada kullanılacak olan temel yaklaşımlar ele alınmış olup görüntü alt yazılama modelinde kullanılan kodlayıcı-kod çözücü mimari, bilgisayarlı görü kavramı, doğal dil işleme, evrişimli sinir ağları, özyinelemeli sinir ağları, kelime temsil yöntemleri gibi konu başlıklarına yer verilmiştir.

3.1. Kodlayıcı Kod Çözücü Mimari (Encoder- Decoder Architecture)

Görüntü alt yazılama problemlerinin çözümünde kodlayıcı kod çözücü mimari yaygın olarak kullanılmaktadır. Kullanılacak kodlayıcı ve kod çözücü mimarisinin seçimi modelin başarımını doğrudan etkilemektedir. Kodlama esnasında, giriş verileri içerdikleri önemli özellikler kaybedilmeden daha küçük boyutlu özellik vektörleri olarak temsil edilmektedir [16]. Giriş verileri genellikle yüksek boyutlu ve karmaşık yapıda olabilirler, bu sebeple verilerin düşük boyutlu temsilleriyle işlem görüyor olması önemlidir. Görüntü alt yazılamada, VGG-Net, AlexNet, Inception, ResNet gibi CNN mimarilerinden biri kodlayıcı olarak seçilerek görüntülere ait öznitelikler çıkarılmaktadır. Hangi mimarinin seçildiği alt yazılama modelinin başarısını etkilemektedir. Kod çözücü ise oluşturulan öznitelik vektörlerini kullanarak açıklayıcı bir cümle oluşturmaktadır. Yaygın kullanılan kod çözücü yöntemlerinden biri dil modellemedir. Dil modeli, kendisine verilen n adet kelimeyi almakta ve önceden tanımlanmış kelime sözlüğündeki her bir kelime için bir sonraki kelime olma olasılığını hesaplamaktadır. Böylece görüntüyü temsil edebilecek alt yazı metni ardışık bir şekilde tahmin edilmektedir. Görüntü alt yazılamada kod çözücü olarak çoğunlukla RNN tabanlı LSTM mimarisi kullanılmaktadır. Kodlayıcı/kod çözücü mimarisi, makine çevirisi, görüntü alt yazılama, sesten metne çeviri, metin özetleme, duygu analizi gibi alanlarda yaygın olarak kullanılmaktadır. Örneğin Youtube ve benzeri platformlarda kullanıcılara sunulan altyazıların farklı dillere çevirisi özelliği, temelde ses verisinin metne dönüştürülmesi, sonrasında kodlayıcı kod çözücü mimari kullanılarak diller arası çeviri yapılmasına dayanmaktadır. Bu tür problemler görüntü alt yazılama modelleriyle benzer şekilde kodlayıcı/kod çözücü mimarisini kullanmaktadır.

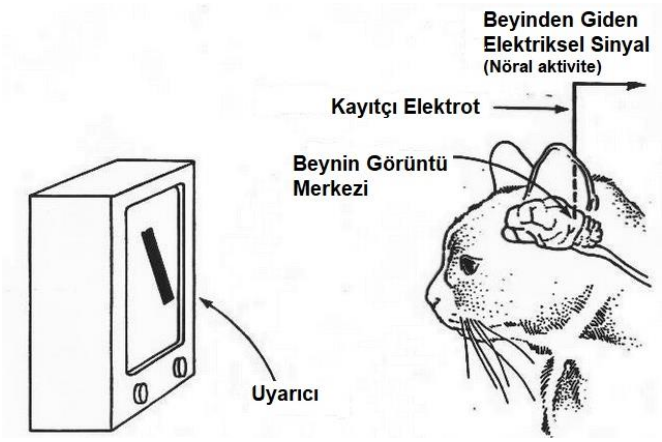
3.2. Doğal Dil İşleme (Natural Language Processing - NLP)

Doğal dil işleme bilgisayarların insan diliyle iletişim kurabilmesi için tasarlanmış bir yapay zeka alanıdır ve doğal dildeki ifadeleri yorumlamak, kelimeler arası ilişkileri anlamak, kelimelerin anlamını çıkarmak gibi işlemleri gerçekleştirmektedir. Alan Turing tarafından yapılan çalışmada "Makineler düşünebilir mi?" sorusunun cevabı aranmaktadır ve bu çalışma doğal dil işlemenin başlangıcı sayılmaktadır [17]. Doğal dil işleme yöntemlerinin gelişmesiyle birlikte bilgisayarların metin tabanlı soruları yorumlayarak cevaplayabilmesi, otomatik çeviriler yapabilmesi, metinleri analiz ederek sınıflandırabilmesi mümkün olmaktadır. Doğal dil işleme uygulamalarına geçilmeden önce metin üzerinde birtakım ön işleme adımları gerçekleştirilmekte sonrasında kelime vektörleri veya doküman vektörleri gibi yöntemler kullanılarak metinlere ait öznitelikler çıkarılmaktadır. Doğal dil işleme yapılırken dünya üzerindeki doğal dillerin farklı dilbilgisel özelliklere sahip olduğu bilgisi göz önünde bulundurularak her bir dil için ayrı ayrı analizler yapılmakta her bir dilin kendi yapısına uygun yöntemler geliştirilmektedir.

3.3. Bilgisayarlı Görü (Computer Vision - CV)

Bilgisayarlı görü temel olarak insanlardaki görsel algılama ve anlama becerisini bilgisayarlar üzerinde oluşturmayı amaçlayan, bilgisayarların görsel verilerden anlam çıkarmasını sağlayan tekniklere verilen isimdir. Bilgisayarlı görü kavramı özellikle 2012 yılında nesne sınıflandırma için yapılan, büyük ölçekli görsel tanıma (ImageNet) yarışmasında Alexnet modelinin elde ettiği başarı ile yaygınlaşmaya başlamıştır [18]. Ancak evrişimli sinir ağları tarafından desteklenen bilgisayarlı görü sistemlerinin tarihi 1950'lerin sonuna dayanmaktadır. Bilgisayarlı görü alanındaki en etkili makalelerden biri bilgisayar bilimleri ya da yazılım mühendisliği alanlarıyla doğrudan ilgisi bulunmayan iki nörofizyolog tarafından 1959'da yayınlanan "Receptive fields of single neurons in the cat's striate cortex" makalesidir. Bu çalışmada Hubel ve Wiesel, anestezi altındaki bir kedinin beyninin birincil korteks alanına elektrotlar yerleştirmiş ve kediye çeşitli görseller gösterirken o bölgede oluşan nöral aktiviteyi gözlemlemişlerdir. Araştırmaları sonucunda beynin birincil korteks alanında basit ve karmaşık nöronlar olduğunu, beyinde görselleştirmenin ilk etapta basit nöronlar aracılığıyla gerçekleştiğini ve nöronu öncelikle görsele ait kenarların hareketlendirdiğini fark etmişlerdir [19]. Bu keşif derin öğrenme modellerinin tasarımında ilham kaynağı olmuştur ve katmanlı yapıda öğrenme sırasında önce görsellere ait kenar

bilgilerinin daha sonra çok daha karmaşık özelliklerin öğrenildiği bilgisi üzerinden modeller geliştirilmiştir.



Şekil 3.1. Hubel ve Wiesel deneyi [20]

1982'de, İngiliz bir sinirbilimci olan David Marr, görmenin hiyerarşik olduğunu saptamıştır. [21]. Öncelikle düşük seviyeli algoritmalar ile kenarların ve köşelerin tespit edildiğini, sonrasında katmanlı bir şekilde üç boyutlu modellenen görsel verinin üst düzey bir şekilde anlaşıldığını ifade etmiştir.

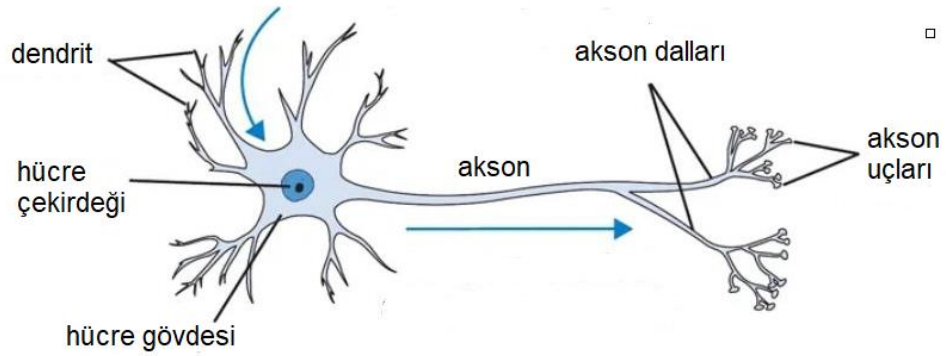
Bilgisayarlı görü alanında gerçekleşen bir diğer önemli çalışma, 1959 yılında Russell Kirsch ve ekibi tarafından resmi ızgaralara bölerek bilgisayarların anlayacağı şekilde ikili makine diline çeviren aparatın icadıdır. Bu sayede görüntüler dijital görüntüye çevrilerek işlenebilmektedir [22]. 1989'da, Fransız bilim adamı Yann LeCun ve diğerleri geri yayılım (backpropagation) içeren bir öğrenme algoritması uygulamış ve evrişimli sinir ağı LeNet'i kullanıma sürmüşlerdir [23].

1990'ların sonlarında bilgisayarlı görü alanındaki çalışmalar başka bir yöne evrilmiştir. O yıllara kadar genel kabul gören yöntem nesnelerin 3 boyutlu modellerini oluşturarak tespitinin yapılması iken, 1999 yılı itibariyle öznitelik çıkarımı yapılarak nesne tespiti yapılmaya başlanmıştır. 2001 yılında, gerçek zamanlı olarak çalışan ilk yüz algılama uygulaması Paul Viola ve Michael Jones tarafından tanıtılmıştır [24]. Yüz algılamada görüntü işlenirken görüntüdeki kişiye ait öznitelikler öğrenilmekte ve öznitelikler arası benzerlik/aynılık esasına bakılarak yeni gelen görüntüdeki kişinin aynı kişi olup olmadığına karar verilmektedir. Viola ve Jones tarafından tanıtılan yüz detektörü günümüzde yaygın olarak kullanılmaktadır.

Bilgisayarlı görü alanında çalışmalar arttıkça üzerinde çalışılacak güçlü ve yüksek boyutlu veri setlerine ihtiyaç duyulmaya başlanmıştır. 2006 yılında nesne sınıflandırması için standartlaştırılmış bir veri kümesi sağlayan Pascal VOC projesi başlatılmıştır [25]. 2010 yılında ImageNet veri kümesi üzerinde çalışan ILSVRC (ImageNet Large Scale Visual Recognition Challenge) yarışması düzenlenmiştir. Bu yarışma ile ImageNet veri kümesi üzerindeki farklı nesne sınıflarının tanınması beklenmektedir. ILSVRC yarışması bilgisayarlı görü ve derin öğrenme alanlarında önemli bir ilerletici unsur olarak öne çıkmaktadır.

3.4. Yapay Sinir Ağları

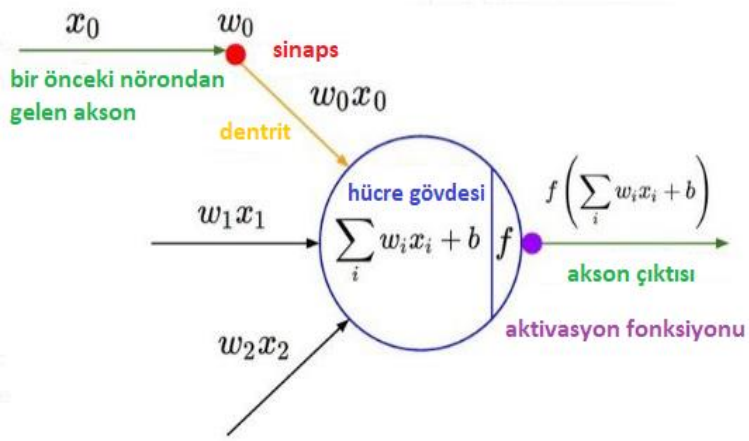
Yapay sinir ağları insan beyninin öğrenme yöntemini taklit eden ve beyindeki nöronların matematiksel olarak modellenmesi sonucu ortaya çıkmış bir teknolojidir [26]. Lineer bir fonksiyon ile ifade edilebilen nöronlar yapay sinir ağlarının en temel hesaplama birimi olarak kabul edilmektedir ve Rosenblatt tarafından tanımlanmıştır (Rosenblatt, 1960). Canlılarda dentrit uçlarından gelen elektrokimyasal uyarı hücre gövdesine girerek ağırlıklandırılmakta ve akson boyunca iletilerek diğer sinir ucuna aktarılmaktadır. Canlılara ait sinir hücresinin yapısı Şekil 3.2'de gösterilmektedir.



Şekil 3.2. Sinir hücresinin biyolojik gösterimi [28]

Canlılardaki bu yapıya benzer bir yapı kullanılarak yapay sinir ağları tasarlanmıştır ve bilgi iletimi bu yapı sayesinde sağlanmaktadır. Yapay sinir ağları temelde giriş katmanı, gizli katman/katmanlar ve çıktı katmanından oluşmaktadır. Giriş katmanına gelen veriler gizli katmanlarda birçok işlemden geçtikten sonra çıktı katmanına verilmektedir. Yapay sinir ağlarında çıktı tahmin edilirken önceden belirlenmiş bir fonksiyon ya da algoritma bulunmamaktadır. Tahmin edilen çıktı değeri ile gerçek değerin kıyaslanması sonucu

hesaplanan bir kayıp fonksiyonu ve bu fonksiyonun çıktısına göre güncellenen ağırlık değerleri elde edilmekte, dolayısıyla gelen her girdi ile sürekli bir öğrenme gerçekleşmektedir. Yapay sinir ağına ait hücreler ise girdi katmanı, ağırlık değerleri, toplama fonksiyonu, aktivasyon fonksiyonu ve çıktılardan oluşmaktadır. İnsanlardaki sinir hücresinin matematiksel modellemesi Şekil 3.3'te gösterilmektedir.



Şekil 3.3. Sinir hücresinin matematiksel modellemesi [29]

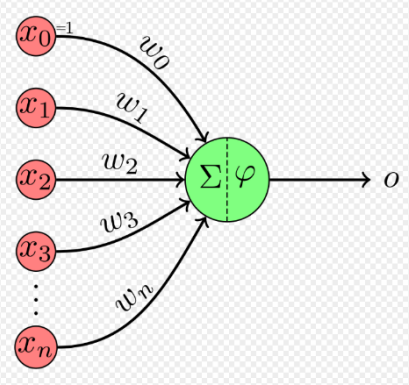
Yukarıdaki şekilde x değerleri, ağa giren veriye karşılık gelmektedir. Bu veriler ilk girdi olabileceği gibi başka bir ağın çıktısı da olabilmektedirler. Nöronlara ait her bir girdi, w ile ifade edilen ve rastgele bir başlangıç değeri atanan ağırlıklar ile çarpılmaktadır. Bu ağırlıklar modelin eğitimi süresince güncellenerek en iyi öğrenmeyi sağlayacak nihai değerlerine ulaşmaktadır. Yapay sinir ağı modellerine öğrenmenin sağlanabilmesi için aktivasyon fonksiyonları kullanılmaktadır. Aktivasyon fonksiyonları her bir nöronunun işlenmesini sağlayan önemli bileşenlerdir. Her bir nöron, bir diğer nöronun çıktısını girdi sinyali olarak almakta ve bu sinyali aktivasyon fonksiyonu ile işleyerek sonuç üretmektedirler. Aktivasyon fonksiyonları, iletilen sinyallerin işlenebilmesi için belirlenen bir eşik değerinin aşılması koşulunu kontrol ederek nöronların aktivasyonunu düzenlemektedir. Bu sayede sinyallerin etkili bir şekilde iletimi ve sinir ağındaki bilgi akışının sağlanması mümkün olmaktadır. Yapay sinir ağının karmaşık gerçek dünya problemleri ile uyumlu çalışabilmesi ve doğrusal olmayan ilişkileri öğrenebilmesi amacıyla nöronların çıktılarına aktivasyon fonksiyonları ile doğrusal olmayan özellikler kazandırılarak tekrar düzenlenmektedir.

Aktivasyon fonksiyonlarının doğru bir şekilde seçilmesi ve uygulanması yapay sinir ağlarının performansını artırmakta ve daha karmaşık veri kümeleri üzerinde başarılı sonuçlar elde edilmesini sağlamaktadır. Yapay sinir ağları içerisinde barındırdığı gizli katman

sayısına göre tek katmanlı (perceptron) veya çok katmanlı sinir ağları olarak sınıflandırılmaktadır.

3.4.1. Tek katmanlı sinir ağları (Perceptron)

Tek katmanlı sinir ağları içerisinde tek bir nöron barındırmaktadır, bu sebeple doğrusal olmayan problemlerin çözümünde yetersiz kalmaktadırlar. Şekil 3.4'te perceptron modeli görülmektedir. Burada x değerleri girdiyi, w değerleri ise girdilerle ilişkili ağırlıkları temsil etmektedir.



Şekil 3.4. Tek katmanlı sinir ağı gösterimi [49]

$$f(x) = w.x + b \tag{3.1}$$

b: bias

w: ağırlık

x: giriş değeri

f(x): çıkış değerini ifade etmektedir.

(3.1) numaralı formülde yapay sinir ağlarında kullanılan fonksiyonların en temel gösterimi bulunmaktadır. Öncelikle girdiler ağırlık değerleri ile çarpılmakta, daha sonrasında yanlılık (bias) değeriyle toplanmaktadır. Bias, nöron çıktılarını etkilemek ve sinir ağının çıktılarını esnek hale getirmek amacıyla kullanılmaktadır ve girdiler toplamına eklenen sabit bir değeri ifade etmektedir. Ayrıca aktivasyon fonksiyonunun eşik değerinin belirlenmesinde ve buna bağlı olarak çıktının ne derecede aktive olacağının kararının verilmesinde rol almaktadır. Bias değeri ve ağırlıklandırılmış girdi değerlerinin toplamı, aktivasyon fonksiyonundan geçirilerek çıktının doğrusal olması engellenmektedir.

3.4.2. Çok katmanlı yapay sinir ağları

Tek katmanlı ağlar doğrusal olmayan problemleri başarılı bir şekilde çözememektedir. Bu tip problemlerde girdi, çıktı ve birden fazla ara katmandan oluşan çok katmanlı yapay sinir ağları kullanılmaktadır. Birden çok gizli katmana sahip olan bu tip yapay sinir ağlarına derin sinir ağları denilmektedir ve gizli katman sayısı artırılarak derinlik sağlanmaktadır. Derin sinir ağlarının Evrişimli Sinir Ağları, Tekrarlayan Sinir Ağları gibi farklı derin öğrenme mimarileri bulunmaktadır. Görüntü, ses, video işleme gibi problemlerde Evrişimli Sinir Ağları daha başarılı olurken, metin tabanlı problemlerde veya hava durumu tahmini gibi ardışık verilerin olduğu problemlerin çözümünde Tekrarlayan Sinir Ağları daha başarılı olmaktadır [30].

Katmanların her birinde, içerisinde bilgiyi barındıran nöronlar ve nöronları birbirine bağlayan ağlar bulunmaktadır. Her bir katmanda verilere ait farklı özellikler ve temsiller öğrenilmektedir ve bu özellikler yardımıyla verilerin karmaşıklığı modellenebilmektedir. Her bir nöron doğrusal olmayan bir matematiksel fonksiyon olan aktivasyon fonksiyonunu kullanarak diğer katmanlardaki nöronlar ile bilgi paylaşımı yapmaktadırlar.

Çok katmanlı yapay sinir ağlarında öğrenme için ileri yayılım (forward propagation) ve geri yayılım (back propagation) yöntemleri kullanılmaktadır. İleri yayılım sırasında girdi değerleri ve başlangıçta rastgele atanan ağırlıklar çarpılmakta ve elde edilen bu değer bias değeriyle toplanarak nöronların çıktısı hesaplanmaktadır. Bu işlem ağın tüm katmanları boyunca devam etmekte ve sonunda tahmin değerleri elde edilmektedir. Elde edilen tahmin değerleriyle gerçek değerler arasındaki fark hesaplanarak kayıp fonksiyon değeri belirlenmektedir. Gerçek değer ve tahmin değeri arasında önemli bir fark bulunması durumunda geriye yayılım (back propogation) algoritması kullanılarak ağırlık değerleri güncellenmektedir. Bu işlem hata fonksiyonu sonucu minimum değerine ulaşıncaya kadar devam etmektedir. Minimum değere ulaşıldığında kullanılan ağırlık değerlerinin öğrenme açısından en uygun değerler olduğu kabul edilmektedir.

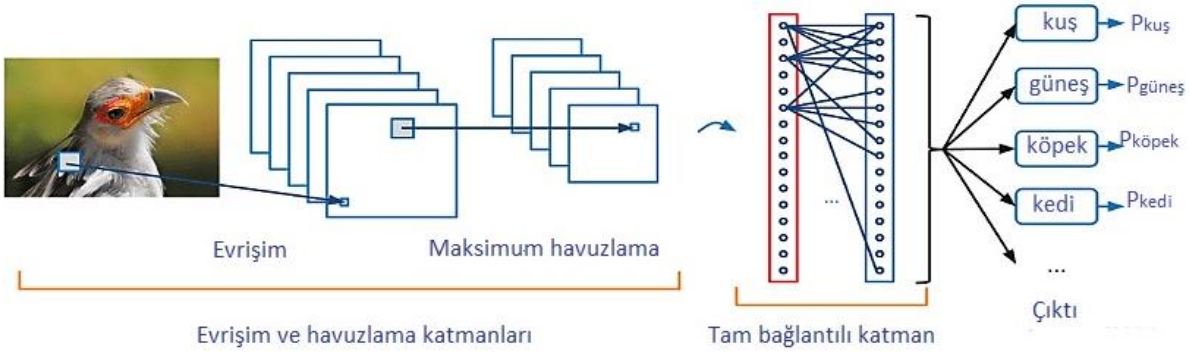
3.5. Derin Öğrenme

Derin öğrenme, derin sinir ağlarını kullanarak veri kümesi üzerinde öğrenme gerçekleştiren bir makine öğrenmesi yöntemidir. Derin öğrenmede, beyindeki sinir hücrelerinin çalışma prensibini modelleyen yapay sinir ağları daha karmaşık ve daha derin bir mimaride kullanılmaktadır. Derin sinir ağlarının içerisinde barındırdığı katman ve nöron sayısı arttıkça derin öğrenme yönteminin de başarımı artmaktadır.

3.6. Evrişimli Sinir Ağları (Convolutional Neural Network - CNN)

CNN, çoğunlukla görüntü işleme problemlerinde kullanılan ve görsel materyalleri girdi olarak alıp işleyen bir derin öğrenme mimarisidir. Bu mimari, makine öğrenmesi algoritmalarının aksine verilerin etiketlerini veya özniteliklerini önceden bilmek zorunda olmadan öğrenme yeteneğine sahiptir. Başka bir deyişle CNN, içerisinde bulunan katmanlar sayesinde girdi olarak verilen görsellere ait öznitelikleri yakalayabilmekte ve sınıflandırabilmektedir. Daha önceki bölümlerde CNN mimarisi tasarlanırken Hubel ve Wiesel tarafından yapılan çalışmada öne çıkan canlılarda görmenin katmanlı şekilde gerçekleştiği bilgisinden faydalanılarak; önce temel bileşenlerin sonrasında ise daha karmaşık özelliklerin tanındığı bir modelleme yapıldığından bahsedilmişti. Buna göre ilk katmanlarda görüntüye ait kenarlar tespit edilirken ilerleyen katmanlarda daha belirgin şekiller ve görüntüde tanınması daha zor olan el, yüz gibi karmaşık öznitelikler belirlenmektedir.

Evrişimli sinir ağları Şekil 3.5.'te gösterildiği şekilde, Convolutional Layer (Evrişim Katmanı), Pooling (Havuzlama Katmanı) ve Fully Connected (Tam Bağlantı) katmanlarından oluşmaktadır. Bu katmanlardan geçen görseller, ön işleme işlemlerine tabi tutularak derin öğrenme modeline hazırlanmış olmaktadır. Verilere ait özniteliklerin CNN modeli içerisinde yakalanması, veri ön işleme sürecini makine öğrenme yöntemlerine göre çok daha kolay hale getirmektedir.

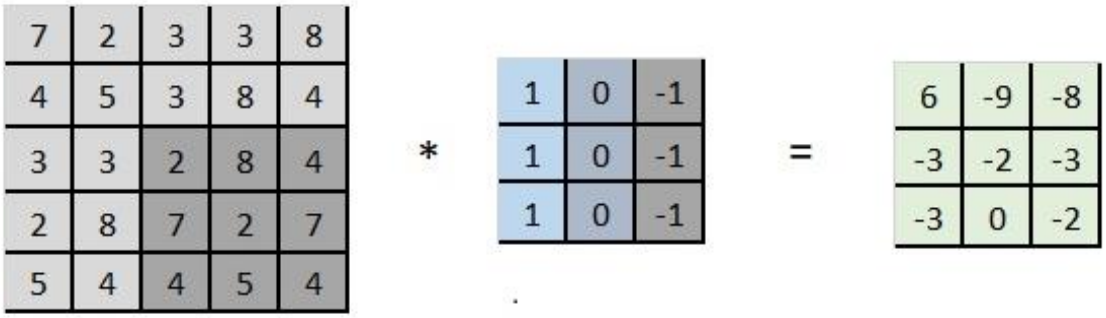


Şekil 3.5. Evrişimli Sinir Ağları Mimarisi [50]

3.6.1. Evrişim Katmanı

Evrişim katmanı evrişimsel sinir ağlarının temelini oluşturmaktadır ve bu katmanda amaç giriş verilerinden girdinin özellik haritasının (feature map) çıkarılmasıdır. Görsellere ait öznitelikleri (kenar bulma, nesne bulma vb.) tespit etmek için görseller üzerinde birtakım filtreler uygulanmaktadır. Her filtre bir özellik tespiti için kullanılmaktadır dolayısıyla

birden fazla özellik tespiti için birden fazla filtreye ihtiyaç duyulmaktadır. Şekil 3.6.'da evrişim işleminin uygulaması yer almaktadır.



Şekil 3.6. Evrişim Katmanı Uygulaması

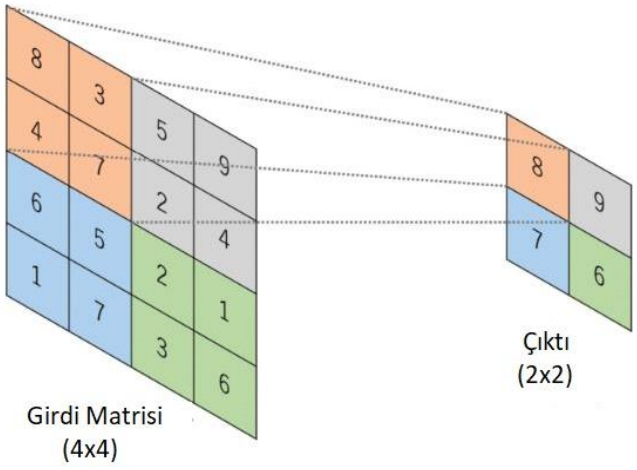
Şekil 3.6'da 5x5 boyutundaki ilk matris görüntüye ait piksel değerlerini barındıran görüntü matrisini temsil etmektedir. 3x3'lük ikinci matris ise filtreyi ifade etmektedir. Evrişim işlemi yapılırken filtre, görüntü matrisi üzerinde önce yatay daha sonra dikey şekilde kaydırılarak ilerlemekte ve matris çarpımı yapılarak daha küçük boyutlu bir matris elde edilmektedir. Elde edilen daha düşük boyutlu yeni matris ile birlikte görüntüye ait bir özellik keşfedilmiş olmaktadır. Evrişimsel sinir ağlarından önceki yöntemlerde filtreler kullanılmadan ham görüntü pikselleri üzerinde işlem yapılması büyük matrislerle çalışmayı gerektirirken, evrişim katmanı sayesinde görüntü üzerinde yerel bölgeler aranarak daha verimli bir yöntem keşfedilmiştir. Bu yöntemde bir sonraki katmandaki nöronlar, yalnızca önceki katmanın ilgili bölgesinden girdi almakta ve bağlantı boyutu önemli ölçüde azalmaktadır. Filtreyi temsil eden matrisin görüntü matrisi üzerinde ilerleme değeri *stride* olarak adlandırılmaktadır. Stride yani adım değeri ne kadar büyük olursa çıktıda ulaşılacak görüntü matrisinin boyutu da aynı oranda küçülecektir. CNN modeli tasarlanırken stride değerinin doğru belirlenmesi modelin performansını etkilemekte ve hesaplama süresini azaltmaktadır. Bu sebeple bu değer belirlenirken problemin özellikleri ve ihtiyaçları göz önünde bulundurulmalıdır.

Evrişim katmanı görüntüye ait özniteliklerin bulunmasında önemli faydalar sağlamaktadır ancak bu katmanın dezavantajları da mevcuttur. Evrişim işlemi esnasında girdi kenarlarındaki piksel değerleri kısmen kullanıldığından kenarlarda bilgi kaybı yaşanabilmektedir. Bunun önüne geçmek için *padding* işlemi uygulanmaktadır. Padding işlemi girdi çevresine fazladan piksel değerleri eklenmesidir, böylece kenarlardaki bilgi kaybı önlenebilecektir. Fazladan eklenen bu piksel değerleri, genellikle sıfır veya tekrar eden

piksel değerleri olarak seçilmektedir. Kenarlara sıfır değerinin eklenmesi literatürde *zero padding* olarak adlandırılmaktadır.

3.6.2. Pooling (Havuzlama) Katmanı

Bu katmanın amacı baskın özellikleri kaybetmeden parametre sayısını düşürerek modelin karmaşıklığını azaltmak ve sonraki katmanlardaki hesaplama maliyetlerini düşürmektir. Bu sebeple havuzlama katmanında boyut indirgeme işlemi yapılmaktadır. Bu işlemi gerçekleştirmek için yaygın olarak ortalama ortaklama (average pooling), minimum ortaklama (minimum pooling) ve maksimum ortaklama (maximum pooling) yöntemleri kullanılmaktadır. Şekil 3.6'da maksimum ortaklamaya ait bir örnek gösterilmektedir.



Şekil 3.7. Maksimum Ortaklama örneği [51]

Şekil 3.7'de maksimum ortaklama yapılırken matris boyutunun 2x2 olarak belirlendiği görülmektedir. Belirlenen bu matris görüntü matrisi üzerinde dolaştırılmakta ve kesiştiği noktadaki piksel değerleri arasından en büyük değer seçilmektedir. Seçilen bu değer yeni matrise yazılarak görüntü matrisinin boyutu azaltılmaktadır. Böylece içerisinde yeterli bilgiyi barındıran dolayısıyla ağın doğru kararlar vermesini sağlayacak aynı zamanda daha küçük boyutlu görüntü matrisleri elde edilmektedir.

3.6.3. Düzleştirme Katmanı (Flattening Layer)

Görüntü işleme ile ilgili problemlerde derin sinir ağlarının giriş verisi, matrisler veya tensörler şeklinde temsil edilen görüntüler olmaktadır. Örneğin genişliği ve yüksekliği 224 pikselden oluşan renkli bir görüntü yükseklik, genişlik ve kanal sayısı olmak üzere 224x224x3 şeklinde bir matrisle temsil edilmektedir. Öğrenmenin gerçekleştiği tam bağlantı

katmanında görüntülerin işlenebilmesi için tek boyutlu bir vektör formuna dönüştürülmeleri gerekmektedir. Düzleştirme katmanının amacı evrişim ve havuzlama katmanından gelen çok boyutlu matris formundaki verileri tek boyutlu olacak şekilde bir diziye çevirerek tam bağlantı katmanına hazır hale getirmektir.

3.6.4. Tam Bağlantı Katmanı (Fully Connected Layer)

Evrişimsel Sinir Ağları'nda evrişim, ortaklama ve düzleştirme katmanlarından sonra tam bağlantılı katman gelmektedir. Tam bağlantılı katman her bir girdinin tüm nöronlara bağlı olduğu bir yapıdadır. Önceki katmanlarda çeşitli işlemlerden geçen çok boyutlu veri matrisi düzleştirme işlemi uygulanmış bir şekilde tam bağlantı katmanına girdi olmaktadır. Daha önceki aşamalarda çıkarılmış olan özellik haritaları işlenmekte ve üst düzey özellik haritaları kullanılarak görüntü sınıflandırma gerçekleştirilmektedir. Tam bağlantı katmanı, farklı derin öğrenme mimarilerinde farklı parametreler alarak öğrenme işlemini gerçekleştirmektedir.

3.7. Tekrarlayan Sinir Ağları (Recurrent Neural Networks- RNN)

Tekrarlayan Sinir Ağları, tıpkı evrişimli sinir ağları gibi derin öğrenmenin temel mimarilerinden biridir. Derin öğrenmede yapay sinir ağları ile insan beyni modellenmeye çalışılmaktadır. İncelemeler sonucu görülmüştür ki insan beyni yalnızca bir yöne doğru çalışan nöronlardan oluşmamaktadır. İnsan beyni herhangi bir konu hakkındaki bilgiyi öğrenebilmek için o konu hakkında daha önceden öğrendiği bilgiler ile yeni gelen bilgiyi birleştirmektedir. Eski bilgiler tecrübe şeklinde elde edilir ve bilgiler yanlış ise unutulmaya zorlanmaktadır. Bu yönden bakıldığında, CNN mimarisi insan beyninin veriyi işleme yöntemiyle kıyaslandığında son derece basit bir mimari olarak kalmaktadır. CNN'in eksik kaldığı durumlarda, özellikle sıralı ve birbiri arasında bağıntılar bulunan verileri modellemede, tekrarlayan sinir ağları (RNN) yaygın olarak kullanılan algoritmaların başında gelmektedir. Temelde tekrarlayan sinir ağları bir önceki adımın çıkışını mevcut adımın girişi olarak kullanan, önceki girişi ve mevcut girişi hatırlama üzerine kurulmuş bir mimaridir ve içerisinde bellek barındırma özelliği bulunmaktadır. Klasik derin öğrenme ağlarında gelen bir bilgi öğrenilirken ya da bir nesne tanınırken sadece o anda verilen giriş bilgileri kullanılmaktadır. Ağa daha öncesinde verilen bilgiler kullanılmamakta ve tahminler eski bilgiler kullanılmadan yapılmaktadır. Klasik derin öğrenme ağlarında giriş ve çıkışlar birbirinden bağımsız olarak çalıştığından ve dolayısıyla girdilerin ağa geliş sırasının bir

önemi olmadığından bir sonraki kelimeyi tahmin etme gibi temel Doğal Dil İşleme (NLP) problemlerinde başarısız olmaktadırlar. Tekrarlayan sinir ağları ise geçmiş bilgileri de hatırlayabildiği için doğal dil işleme, sesten metine ya da metinden sese çevirim, metin özetleme, otomatik çeviri (machine translation), anomali tespiti, kelime ve cümle tahmini, borsa tahmini, duygu analizi ve video analizi gibi konularda başarılı sonuçlar vermektedir [21]. Tekrarlayan sinir ağlarının amacı hata payını düşürerek klasik sinir ağlarına göre çok daha güvenilir ve sağlıklı sonuçlar elde etmemizi sağlamaktır. RNN'lerin teoride uzun dizilerde (long-sequence) başarılı sonuçlar vermesi beklenmektedir ancak kısa vadeli bir hafızası olmasından dolayı pratikte bu tür problemlerde yeterince başarılı çalışmadığı görülmektedir. Ayrıca geriye yayılım (backpropogation) sırasında gradyan yok olması problemi de yaşanmaktadır. Gradyan değerleri çok fazla küçüldüğünde RNN öğrenme problemi yaşamaya başlamaktadır.

3.8. Kaybolan Gradyan Problemi (Vanishing Gradient Problem)

Derin sinir ağlarının eğitiminde, gradyanlar ağın çıkış katmanından başlayarak önceki katmanlara yani geriye doğru iletilmektedir ve iletim sırasında her katmanda bir önceki katmandaki gradyanlarla çarpılarak ilerlemektedir. Gradyan çok değişkenli fonksiyonda, fonksiyonun tüm bağımsız değişkenleri üzerindeki kısmi türevlerin vektörel bir temsilidir. Ancak, gradyanların her adımda çok küçük değerlere yakınsaması önceki katmanlardaki ağırlıkların güncellenememesine veya çok yavaş bir şekilde güncellenmelerine sebep olacaktır.

Gradyanların bu şekilde kayboluyor oluşu özellikle derin sinir ağları için sorun teşkil eder çünkü bu ağlar genellikle çok sayıda katmandan oluşmaktadır ve gradyanlar bu katmanlar boyunca geçerken önemli ölçüde küçülmektedirler. Bunun sonucunda uzun süren eğitim süreleri ve düşük performanslı modeller ortaya çıkmaktadır.

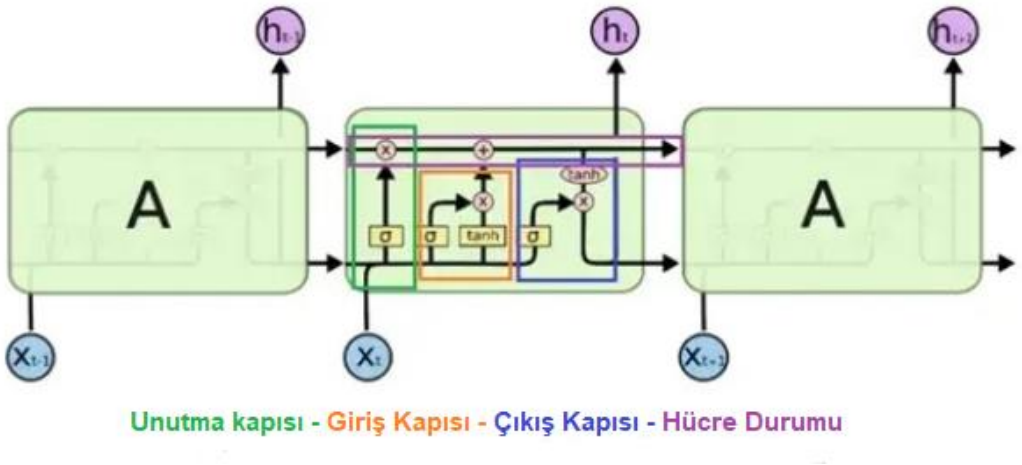
Gradyanların kaybolması, derin sinir ağlarının eğitimi sürecinde dikkate alınması gereken önemli bir husustur. Ağ mimarisinin ve aktivasyon fonksiyonlarının doğru seçilmesiyle gradyanların kaybedilmeden aktarılmasını sağlayacak yapılar elde edilmektedir.

3.9. Uzun Kısa Süreli Bellek (Long Short Term Memory – LSTM)

LSTM, veriler arasında zamana dayalı ya da ardışık türden ilişkilerin bulunduğu durumlarda sıralı verileri işlemek için kullanılan RNN tipinde bir yapay sinir ağı

mimarisidir. Uzun-kısa dönem belleklerin birleşiminden oluşan LSTM'ler Sepp Hochreiter ve Jürgen Schmidhuber tarafından 1997 yılında ortaya atılmıştır. Zamana dayalı verilerin işlenmesinde kullanılan RNN'ler kısa vadeli bir hafızaya sahip olduklarından bazı problemlerde yeterince iyi çalışamamaktadır. Örneğin RNN'e çevirisi yapılmak üzere uzun bir cümle girdi olarak verildiğinde geçmiş verileri hatırlamakta yetersiz kalacağından yeni bilgilerle ilişki kurup tahmin yapmada yetersiz kalacaktır. RNN'de ek olarak geriye yayılım (backpropagation) esnasında gradyan değeri çok fazla küçüleceğinden vanishing gradient problemi dolayısıyla öğrenme problemi yaşanacaktır. RNN'de yaşanan uzun vadede unutma problemleri nedeniyle LSTM ve GRU (Gated Recurrent Unit) gibi daha başarılı ağlar önerilmiştir. LSTM'ler RNN'lere kıyasla daha avantajlı olduğundan; konuşma tanıma, metinden sese çeviri, metinden anahtar kelime çıkarımı gibi NLP problemlerinde ve anomali tespiti, duygu analizi, videodan insan aktivitelerinin sınıflandırılması, müzik besteleri oluşturmak gibi alanlarda sıklıkla kullanılmaktadır. LSTM, zaman içerisinde verilerdeki uzun vadeli bağımlılıkları başarılı bir şekilde modelleyebildiğinden özellikle sıralı verilerin işlenmesinde tercih edilen bir yapay sinir ağı mimarisidir.

LSTM'in mimarisi, bir hafıza hücresi, bir giriş kapısı, bir unutma kapısı ve bir çıkış kapısından oluşmaktadır. LSTM içerisinde bulunan bu yapılarla bilgileri daha iyi saklamakta ve klasik RNN'lerin bellek problemini ortadan kaldırmayı hedeflemektedir. LSTM mimarisinde bulunan hafıza hücresi, ağ mimarisinde bulunan giriş, unutma ve çıkış kapıları sayesinde hangi verilerin tutulacağına ya da hangi verilerin unutulacağına/silineceğine karar vermektedir. LSTM'in yapısı Şekil 3.8'de gösterilmektedir.



Şekil 3.8. LSTM Mimarisi [31]

LSTM mimarisine ait temel bileşenler aşağıda açıklanmaktadır:

Hücre Durumu (Cell State)

Cell State, LSTM ağı içerisinde bulunan önemli bir yapıdır ve bilgiyi taşımak için kullanılmaktadır. Tahminler yapmak için anlamlı uzun süreli bağımlılıkları ve bilgileri hücreler boyunca taşımaktadır bu sebeple ağın hafızası olarak açıklanabilmektedir. Cell state sayesinde kısa süreli bellek (short-term memory) problemi çözülmekte, eski veriler ağ zinciri boyunca taşınabilmektedir. Cell State'in yolculuğu boyunca taşıması gereken bilgiler ise kapılar aracılığı ile belirlenmektedir. Bu kapılar hangi bilginin gerekli veya gereksiz olduğunu verileri 0 ile 1 aralığına sıkıştıran sigmoid aktivasyon fonksiyonunu kullanarak belirlemektedir. Sigmoid aktivasyonu sonucunda 0 olan bilgiler unutulmakta, 1 olan bilgiler ise Cell State ile ilerlemeye devam etmektedir.

Giriş Kapısı (Input Gate)

Giriş kapısı, LSTM ağı içinde önemli bir rol oynamaktadır ve mevcut hücre durumuna bağlı olarak yeni gelen bilgilerden hangisinin hücreye gireceğini kontrol etmektedir. Bu sayede LSTM ağının uzun vadeli bağımlılıkları modelleyebilmesi ve önemli bilgileri uzun süre saklayabilmesi sağlanmaktadır. Sigmoid aktivasyon fonksiyonu kullanılarak gelen bilgilerin ne kadar önemli olduğu tespit edilir ve 0'a yakın değerler önemsiz 1'e yakın değerler ise önemli kabul edilerek hücre durumuna eklenmektedir.

Unutma Kapısı (Forget Gate)

LSTM ağı içerindeki bir diğer önemli yapıdır ve önceki hücre durumunun mevcut hücre durumuna etkisini kontrol ederek hangi bilgilerin unutulması gerektiğine karar vermektedir. Unutma Kapısı, hücre durumunun güncellenmesi için kullanılır ve önceki hücre durumunun mevcut duruma olan katkısını kontrol ederek hangi bilgilerin unutulacağına karar verir. Sigmoid aktivasyon fonksiyonu kullanılarak, önceki hücre durumunun etkileri tespit edilir ve 0'a yakın değerler hafızadan silinecek bilgileri belirtirken, 1'e yakın değerler ise korunacak bilgileri belirtmektedir.

Çıkış Kapısı (Output Gate)

Çıkış Kapısı, LSTM ağında gizli durumun çıkışını belirleyen yapıdır. Hücrenin mevcut durumuna göre bir sonraki gizli durumun ne olacağına karar vermektedir. Sigmoid aktivasyon fonksiyonu kullanılarak, mevcut hücre durumuna bağlı olarak bir sonraki gizli durum belirlenir ve LSTM ağının sonraki adımda neyi dikkate alacağı ve neyi ihmal edeceği tespit edilmektedir.

3.10. Derin Öğrenme Hiper Parametreleri

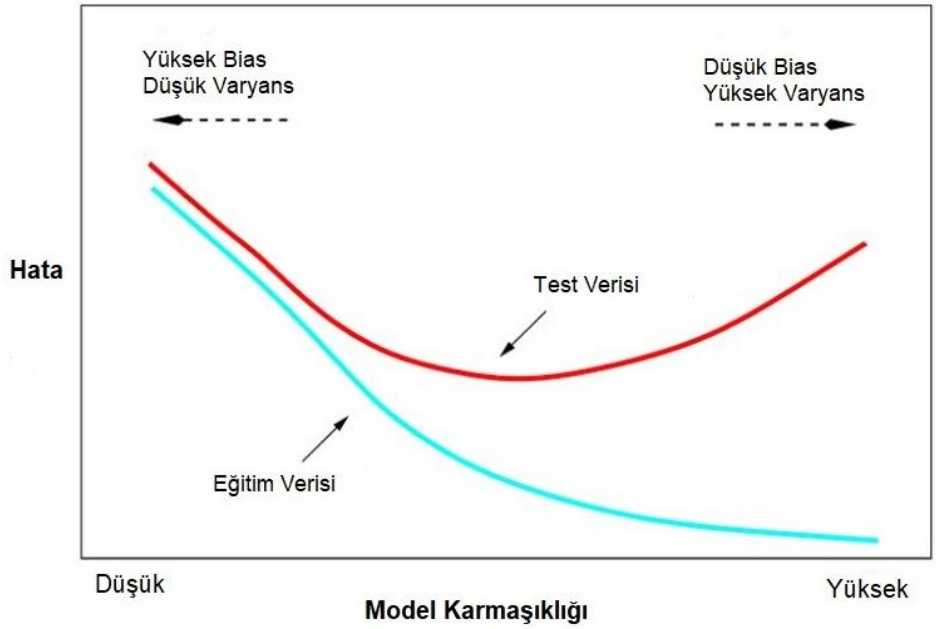
Derin öğrenme hiper parametreleri, derin öğrenme modellerinin yapısını ve davranışını belirleyen ve kullanıcı tarafından manuel olarak ayarlanması gereken değişkenlere verilen genel addır. Hiper parametreler seçilirken veri kümesinin özellikleri, eğitimin gerçekleştirileceği makinenin işlem gücü gibi birçok diğer parametre etkili olmaktadır. Hiper parametreler modelin başarımının artırılmasından doğrudan rol oynamaktadır. Başlıca hiper parametreler şunlardır:

3.10.1. Öğrenme hızı (Learning Rate)

Derin öğrenme modellerinde geri yayılım ile ağırlıklar güncellenmektedir ve yeni ağırlık değerleri hesaplanırken sabit veya değişken bir değer alabilen öğrenme hızı hiper parametresi kullanılmaktadır. Öğrenme hızının çok küçük seçilmesi durumunda modelin eğitim verilerini öğrenme süresi uzayacaktır. Öğrenme hızının çok büyük seçilmesi durumunda ise model ideal noktanın etrafında sürekli bir salınım yapacak ancak hedef noktaya ulaşamayacaktır. Bu durum modelin eğitim sürecini dengesizleştirmiş ve performansını düşürmüş olacaktır. Öğrenme hızının seçiminde yaygın yaklaşım başlangıçta büyük bir değer (0.001 gibi) ile başlamak ve iterasyon sayısı arttıkça bu değeri azaltmaktır. Böylece eğitim süreci başlangıçta hızlı bir şekilde ilerlemekte ve ilerleyen süreçlerde daha hassas bir öğrenmeye geçilmektedir. Bu parametrenin doğru seçilmiş olması modelin hızlı bir şekilde istenen sonuçlara ulaşmasını sağlamaktadır.

3.10.2. Tur sayısı (Epochs)

Tur sayısı, eğitim verilerinin tamamının ağa kaç kez sunulacağını belirlemektedir. Daha yüksek tur sayısı verilerin daha kapsamlı bir şekilde incelenmesine dolayısıyla daha doğru temsil edilmesine olanak sağlamaktadır. Ancak tur sayısının çok yüksek seçilmesinin bazı riskleri bulunmaktadır. Verilerin ağa çok kez sunuluyor olması ***overfitting*** denilen aşırı öğrenme sorununa sebep olabilir. Aşırı öğrenme, modelin eğitim verilerine çok iyi uyum sağlayarak bu verileri ezberlemesine ve hiç görmediği yeni veriler üzerinde çok kötü performans sergilemesine sebep olmaktadır. Aşırı öğrenme problemi olan modellerde yüksek varyans yani yüksek çeşitlilik durumu gözlenmektedir. Tur sayısının küçük seçilmesi de benzer şekilde ***underfitting*** denilen modelin yeterince öğrenememesine ve gerçek verilerle tahmin verileri arasında yüksek hataya (bias) sebep olmaktadır.



Şekil 3.9. Overfitting Underfitting Grafiği [32]

Şekil 3.9'daki grafikte tur sayısının yüksek veya düşük seçilmesi sonucu modelin nasıl davrandığı gösterilmektedir. Grafiğin sağına doğru gidildiğinde tur sayısı dolayısıyla model karmaşıklığı artmaktadır. Bunun sonucunda Düşük Bias – Yüksek Varyans yaşanmaktadır. Yani model eğitim verilerini ezberlediğinden eğitim verileri ile yüksek uyum sağlamış ve eğitim verilerine göre şekil alarak sürekli bir varyans yani değişim göstermektedir. Bu durumda eğitim verileri üzerindeki hata oranı giderek azalırken hiç görmediği test verileri üzerindeki hata oranı artmaktadır.

Grafiğin soluna doğru gidildiğinde tur sayısı dolayısıyla modelin karmaşıklığı azalmaktadır. Bunun sonucunda Yüksek Bias – Düşük Varyans yaşanmaktadır. Burada da eğitim verileri üzerinde yeterince çalışamadığından model kararlı bir yapıda düşük varyans ile ilerlemektedir. Ancak model yeterince öğrenme sağlayamadığından gerçek verilerle arasındaki hata oranı artmaktadır.

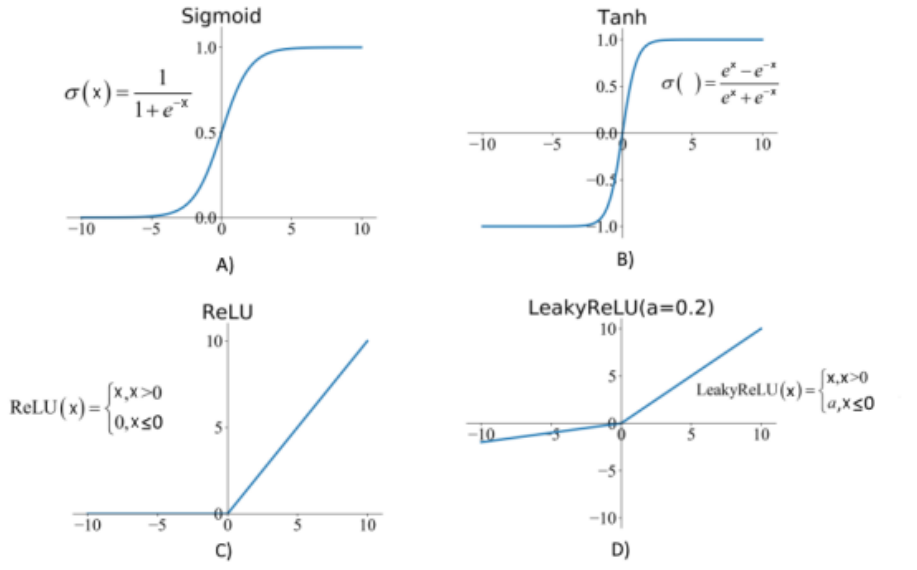
3.10.3. Mini grup boyutu (Mini Batch Size)

Mini grup boyutu, veri kümesinin alt kümelere ayrılması sonucu oluşan her bir alt kümedeki veri sayısını ifade eden bir terimdir. Bazı durumlarda modellerin eğitiminde kullanılacak veri kümesi boyutu hafızaya sığmayacak kadar büyük olabilir. Bu tür durumlarda bütün veri kümesinin aynı anda modele verilerek modelin eğitilmesi hem zaman hem de hafıza açısından oldukça zorlayıcı olmaktadır. Çözüm olarak, mini grup boyutu belirlenerek eğitim gerçekleştirilmektedir. Mini grup boyutunun doğru seçimi, eğitim

sürecinin hızını ve performansını önemli ölçüde etkilemektedir. Yaygın olarak kullanılan mini grup değerleri 4, 8, 16, 32, 64, 128 ve 256 gibi küçük kuvvetlerdir.

3.10.4. Aktivasyon fonksiyonları

Yapay sinir ağlarında ve derin öğrenme modellerinde, nöronların çıktılarını belirlemek için kullanılan matematiksel işlemlere aktivasyon fonksiyonları denilmektedir. Seçilen aktivasyon fonksiyonu modelin performansını etkilemektedir. Yanlış veya uygun olmayan bir aktivasyon fonksiyonunun kullanılması, aşırı öğrenme (overfitting) veya yetersiz öğrenme (underfitting) gibi eğitim süreci problemlerine neden olmaktadır. Overfitting, modelin eğitim verilerine çok iyi uyum sağlaması ancak daha önce hiç görmediği yeni veriler üzerinde kötü performans göstermesi anlamına gelmektedir. Bu durum, modelin eğitim verilerinde bulunan gürültülü değerleri de öğrenerek aşırı karmaşık bir yapı oluşturmasından kaynaklanmaktadır. Underfitting ise modelin eğitim verilerini yeterince öğrenememesi ve verilerdeki temel örüntüleri yakalayamaması durumuna denilmektedir. Aktivasyon fonksiyonları, özellikle çok katmanlı derin öğrenme modellerinde, gradyanların düzgün bir şekilde aktarılmasını ve modelin istenilen sonuca daha hızlı ve doğru bir şekilde ulaşmasını sağlamaktadır. Farklı problemler ve yapılar için farklı aktivasyon fonksiyonları kullanılarak daha doğru sonuçlar elde edilebilmektedir. Yaygın olarak kullanılan aktivasyon fonksiyonları arasında ReLU (Rectified Linear Unit), Leaky ReLU, tanh (Hyperbolic Tangent) ve sigmoid fonksiyonları yer almakta olup matematiksel ifadeleri ve grafikleri Şekil 3.10.'da gösterilmiştir.

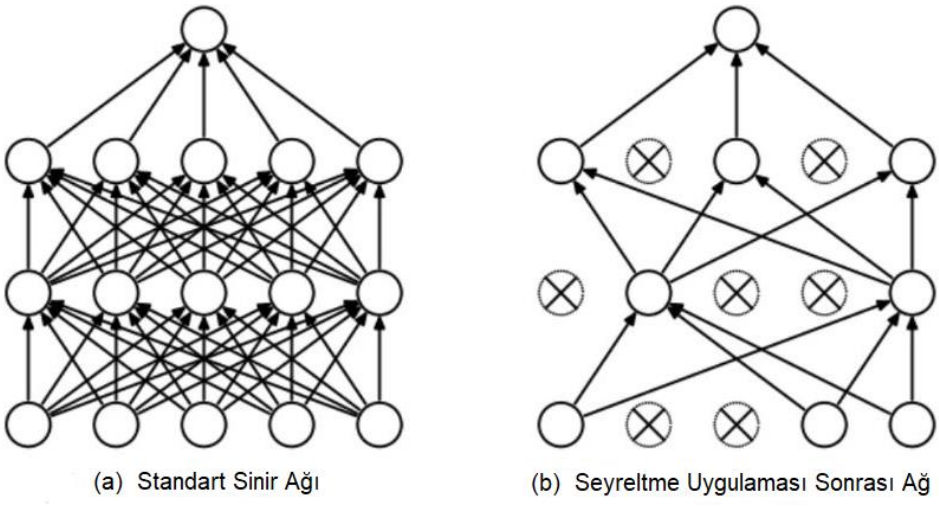


Şekil 3.10. Aktivasyon Fonksiyonları Grafiksel Gösterimi [33]

ReLU: Doğrultulmuş Doğrusal Birim (Rectified Lineer Unit) anlamına gelmektedir ve negatif olan girdi değerlerini çıkışa vermeden önce sıfır ile değiştirmektedir. Sigmoid fonksiyonu çıkış değerlerini 0 ile 1 arasına sıkıştırarak olasılık değerleri elde etmeye yardımcı olmaktadır. Tanh fonksiyonu ise girdi değerlerini -1 ile 1 arasında olacak şekilde sıkıştırmaktadır. Leaky ReLU fonksiyonu ise tıpkı ReLU fonksiyonundaki gibi pozitif değerleri ve 0 değerini değiştirmez ancak negatif değerleri 0,01 ile çarparak değerlerini korumaktadır. Aktivasyon fonksiyonlarının işlevleri incelendiğinde her birinin farklı davranışlar sergilediği görülmüştür. Bu sebeple aktivasyon fonksiyonlarının derin öğrenme modelinin amacına uygun olacak şekilde doğru seçilmiş olması, yapay sinir ağlarının performansını ve öğrenme becerisini önemli ölçüde artırmaktadır.

3.10.5. Seyreltme Oranı (Dropout)

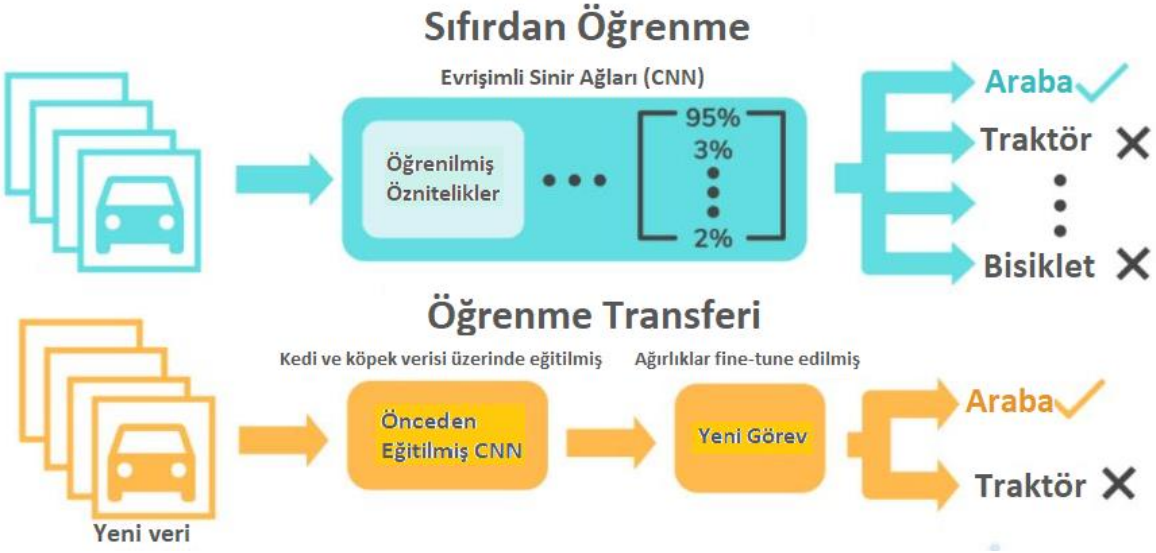
Yapay sinir ağlarında eğitim veri kümesinin aşırı öğrenilmesi (overfitting) model performansını olumsuz yönde etkileyen bir durumdur. Bunun önüne geçilebilmesi ve derin öğrenme modellerinde genelleştirmeyi artırmak amacıyla kullanılan yöntemlerden bir tanesi seyreltme (dropout) uygulamaktır. Seyreltme, her eğitim iterasyonunda uygulanmakta ve rastgele seçilen veya belirli bir eşik değerinin altındaki nöron bağlantılarının geçici olarak kapatılması şeklinde gerçekleştirilmektedir. Kapatılan nöronlar, modelin eğitimi sırasında katkıda bulunamamakta ve ağın çıktısını etkilememektedir. Bu sayede model veriler arasındaki örüntüleri daha doğru bir şekilde yakalayabilmektedir. Seyreltme oranı %20, %50 veya %75 gibi rastgele değerler seçilerek modelin en performanslı hali elde edilmeye çalışılmaktadır. Şekil 3.11'de seyreltme işlemi uygulaması öncesi/sonrası ağın durumu gösterilmektedir.



Şekil 3.11. Seyreltme (Dropout) Modeli [34]

3.11. Öğrenme Aktarımı (Transfer Learning)

Modellerin başarılı bir şekilde eğitilebilmesi için büyük veri kümelerine ve bu verileri işleyebilecek güçlü makinelere ihtiyaç duyulmaktadır. Veri kümesinin büyük seçilmesi model başarımını artırmakla beraber uzun eğitim sürelerine neden olmaktadır. Bu sorunun üstesinden gelmenin en iyi yollarından biri büyük veri kümeleri üzerinde önceden eğitilmiş modellerin aktarılarak yeni problemlerin çözümünde kullanılmasıdır. Öğrenme gerçekleştirilirken sıfırdan öğrenme yapmak yerine hazır modeller kullanılması işlemine *Öğrenme Aktarımı* denilmektedir. Bu işlem insanlardaki tecrübe kavramıyla eşleştirilebilmektedir. İnsan beyni bir bilgiyi öğrenirken geçmiş tecrübelerinden faydalanarak ve çok daha az emek harcayarak daha hızlı bir öğrenme yaşamaktadır. Transfer öğrenmenin avantajlarından biri, büyük veri kümelerine ihtiyaç duymadan, önceden eğitilmiş bir modeli yeni bir görev için uygun bir şekilde ayarlayarak, öğrenme sürecini hızlandırması ve daha az işlem gücü gerektirmesidir. Ayrıca eğitim için yeterli veriye sahip olunamayan durumlarda bile, Transfer Öğrenme sayesinde başarılı sonuçlar elde edilebilmektedir. Şekil 3.12'de Transfer öğrenme ve sıfırdan eğitim yapılarak öğrenme modellerinin karşılaştırılması sunulmuştur. Önceden eğitilmiş modeli yeni probleme uygun hale getirmek için modelin bazı katmanlarını veya parametrelerini değiştirmek gerekmektedir. Böylece modelin yeni veri kümesine adapte olması ve yeni görevinde daha başarılı performans göstermesi sağlanmış olacaktır.



Şekil 3.12. Transfer öğrenme ve sıfırdan öğrenme karşılaştırılması [52]

3.12. Kelime Temsil Yöntemleri

Kelime temsil yöntemleri, doğal dil işleme alanının önemli başlıklarından biri olup metin tabanlı verilerin makine öğrenimi ve derin öğrenme modelleri tarafından işlenebilir hale getirilmesi için kullanılan tekniklere verilen isimdir. Metinler içerisinde önemli bilgiler barındırmaktadır ve nicel verilerden elde edilemeyen bazı bilgilere erişim metinler üzerinden gerçekleşebilmektedir. Bu yöntemlerde metinler sayısal bir vektör olarak temsil edilmekte, metinlerin matematiksel işlemlerle analiz edilebilmesi ve derin öğrenme modellerine giriş olarak kullanılması sağlanmaktadır. Böylece metinlerdeki kelimelerin semantik ve yapısal özellikleri daha iyi yansıtılarak, derin öğrenme algoritmalarının metin verilerini daha doğru işlemesine ve modelin performansının artmasına yardımcı olmaktadırlar. Kelime temsil yöntemleri, dil modelleme, metin sınıflandırma, duygu analizi, soru cevaplama, belge organizasyonu, metin filtreleme, makine çevirisi gibi doğal dil işleme ve metin madenciliği alanlarında yaygın olarak kullanılmaktadır. Doğru kelime temsil yönteminin seçimi, metin verilerinin doğru bir şekilde temsil edilmesini böylece modelin daha iyi bir performans göstermesini sağlamaktadır. Kelimelerin sayısal veriler üzerinden temsil edilebilmeleri için birçok yöntem geliştirilmiştir. Kelime temsil yöntemleri veri ön işleme, öznitelik çıkarma ve öğrenilen kelime temsillerinin sınıflandırılması şeklinde 3 ana bölüm altında incelenebilir.

3.12.1. Veri ön işleme aşaması

Metinsel veriler modele girdi olarak verilmeden önce büyük harften küçük harfe dönüştürme, noktalama işaretleri ve rakamların çıkarılması gibi bazı işlemlerden geçirilerek gürültüden arındırılmaktadır. Sonrasında metinsel veri tokenization (kelimeleri küçük parçalara ayırma), lemmatization (kök çözümleme) veya stemming (gövdeleme) işlemleri ile daha küçük parçalarına ayrılmaktadır.

3.12.2. Öznitelik çıkarma teknikleri

Metinlerden öznitelik çıkarma, metinlerin makine öğrenimi modellerine uygun hale getirilmesi amacıyla sayısal verilere dönüştürmesi için kullanılan bir tekniktir. Metinler çoğunlukla düzensiz yapıda bulunmaktadır bu sebeple makine öğrenimi modellerine verilmeden önce anlamlı özniteliklerin çıkarılması gerekmektedir. Böylece metinlerin içerdiği bilgiler daha kolay işlenebilir ve analiz edilebilir hale gelmektedir. One-Hot Encoding, TF-IDF ve Word Embedding (kelime gömme) gibi çeşitli kelime temsil yöntemleri bulunmaktadır.

One - Hot Encoding

Metinlerdeki kelimeleri bir vektör olarak temsil etmek için kullanılan bir yöntemdir. Bu yöntemde kelimeleri vektör olarak ifade edebilmek için bir kelime sözlüğü oluşturulmaktadır. Sözlükteki her bir kelime benzersiz/tekil bir indekse atanmakta ve kelime metin içerisinde yer alıyorsa kelime indeksine karşılık gelen değer 1, yer almıyorsa 0 olarak işaretlenmektedir. Bu yöntem ile bir kelimenin anlamı hakkında fikir edinilememekte sadece kelimenin metin içerisinde varlığı veya yokluğu tespit edilebilmektedir. Bu yöntemin en büyük dezavantajı listedeki kelime sayısının artmasıyla paralel olarak kelimeyi temsil eden vektör boyutunun da artmasıdır. Ek olarak vektör içerisinde çok fazla sıfır değeri barındıracağından seyrek (sparse) vektörler elde edilecektir. Bu sebeple daha farklı yöntemlere ihtiyaç duyulmuştur.

Terim Frekansı - Ters Belge Frekansı (Term Frequency - Inverse Document Frequency, TF - IDF)

TF-IDF bir kelimenin metin içerisinde kaç kere geçtiği (TF) ve tüm dokümanlarda ne ölçüde yaygın olduğu (IDF) bilgisini dikkate alarak; o kelimenin ne kadar önemli olduğu hakkında bilgi vermekte ve kelimeye ait bir temsil vektörü oluşturmaktadır. TF-IDF değeri TF ve IDF değerlerinin çarpımı sonucu elde edilmektedir. TF değerini ifade eden formül Eş. 3.2'de, TF-IDF değerinin elde edildiği formül ise Eş. 3.3'de verilmektedir.

$$TF = \frac{\text{terimin dokümanda geçme sıklığı}}{\text{dokümandaki toplam terim sayısı}} \tag{3.2}$$

$$w_{i,j} = tf_{i,j} \times log\left(\frac{N}{df_i}\right) \tag{3.3}$$

N: Toplam doküman sayısı

df_i: Terimi içeren doküman sayısı

IDF değeri 0'a ne kadar yakınsa, kelime o kadar yaygın demektir, aksi takdirde bu sayı 1'e yaklaşacaktır. Toplam doküman sayısının 100 olduğunu ve aranan kelimenin tüm bu dokümanlarda geçtiğini varsayalım. Böyle bir örnekte log(100/100) değeri 0 olacaktır. Bunun anlamı aranan kelimenin tüm dokümanlarda geçen yaygın bir kelime olduğudur.

Aynı toplam frekansa sahip iki kelime, TF-IDF ile ifade edilerek farklı dokümanlarda ne sıklıkta geçtikleri ve ne derece önemli oldukları vurgulanmaktadır. Bir kelimenin yalnızca

tek bir dokümanda 50 kez geçtiği, diğer bir kelimenin ise 10 farklı dokümanda her biri 5 kez olmak üzere toplamda 50 kez geçtiği bir örneği ele alalım. Bu durumda, TF-IDF, ikinci seçenekteki kelimeye yani daha çok dokümanda geçen kelimeye daha yüksek önem atamaktadır. Çünkü, bir kelimenin farklı dokümanlarda geçmesi, bu kelimenin daha genel ve yaygın bir anlama sahip olduğunu göstermekte, dolayısıyla da metindeki önemini artırmaktadır.

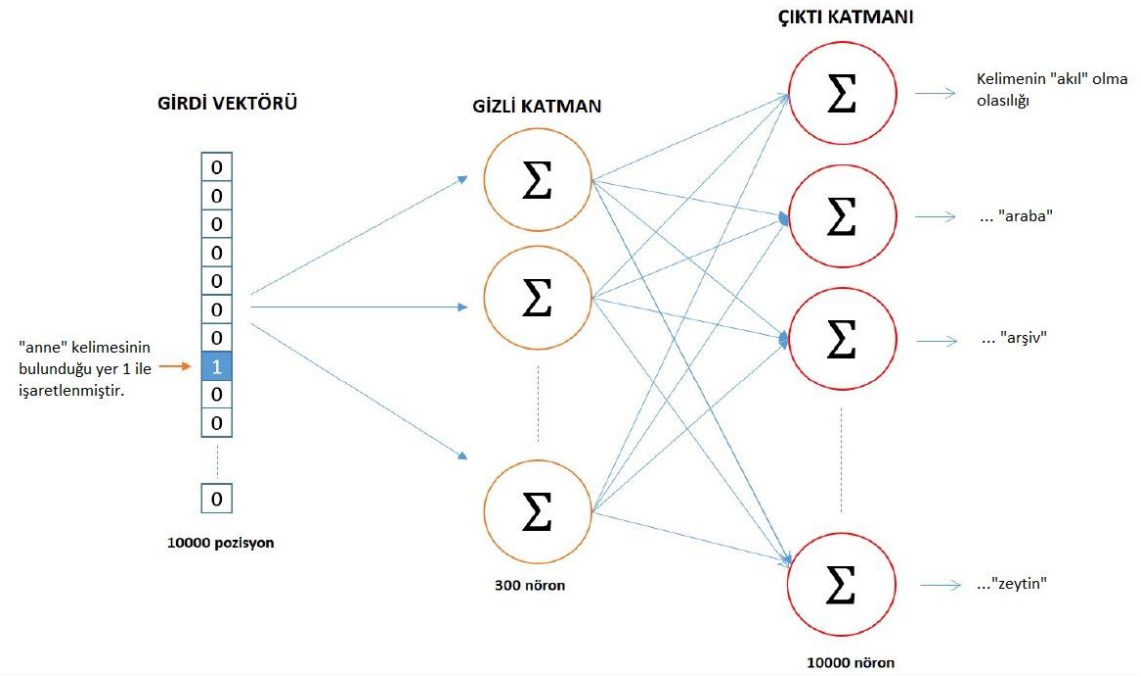
TF-IDF belge sınıflandırması, metin özetleme, içerik önerisi sunma, doküman gruplama, kelime ilişkisi ile anlam analizi ve durdurma kelimesi filtreleme (stop-word filtering) dahil olmak üzere birçok doğal dil işleme (NLP) algoritmasında kullanılmaktadır.

Kelime Gömmeleri (Word Embedding)

Bir önceki başlıkta dezavantaj olarak bahsedilen yüksek boyutlu vektör temsillerinin üstesinden gelmek amacıyla derin öğrenme temelli kelime temsil yöntemleri ortaya atılmıştır. Kelime gömmeleri, benzer anlamdaki kelimelerin benzer şekilde temsil edilmesine yarayan ve kelimeler arası kosinüs uzaklığını kullanan derin öğrenme tabanlı bir yöntemdir [35]. Mikolov ve diğerleri tarafından 2013 yılında geliştirilmiş olan Word2vec en yaygın kullanılan kelime temsil yöntemlerindendir [36].

- **Word2vec (Kelime Vektörleştirme)**

Mikolov ve diğerleri tarafından geliştirilmiş olan Word2vec çalışmalarında, anlamsal olarak birbirine benzeyen kelimelerin vektör uzayında kosinüs uzaklığı hesaplandığında birbirlerine yakın oldukları ve kelimelerin birden fazla benzerlik derecesi barındırabileceği görülmüştür [16]. Word2vec temelde giriş katmanı, gizli katman ve çıkış katmanı olacak şekilde üç katmandan oluşan yapay sinir ağı mimarisini kullanmaktadır. Kelime kaç boyutlu vektör ile temsil ediliyorsa gizli katmanda da o sayı kadar nöron bulunmaktadır. Bu ağın girdisi kelimeye ait one-hot vektör olarak adlandırılan, veri kümesindeki kelimeleri temsil eden sözlük içindeki her kelimenin sıfırlardan oluşan vektörüdür. Eğitim sırasında, stokastik gradyan inişi ve geri yayılım gibi teknikler kullanılarak ağın ağırlıkları eğitilmekte ve kelimelerin vektör temsilleri oluşturulmaktadır. Word2vec mimarisinin amacı, kelimeleri anlamsal olarak benzer kelimelerle aynı bölgede temsil etmek ve dildeki kelime ilişkilerini düşük boyutlu vektörlerle matematiksel bir uzayda göstermektir. Bu sayede kelime vektörleri arasındaki benzerlikler ve ilişkiler yakalanabilmektedir. Şekil 3.13'te Word2vec mimarisi ve ağın katmanları gösterilmiştir.

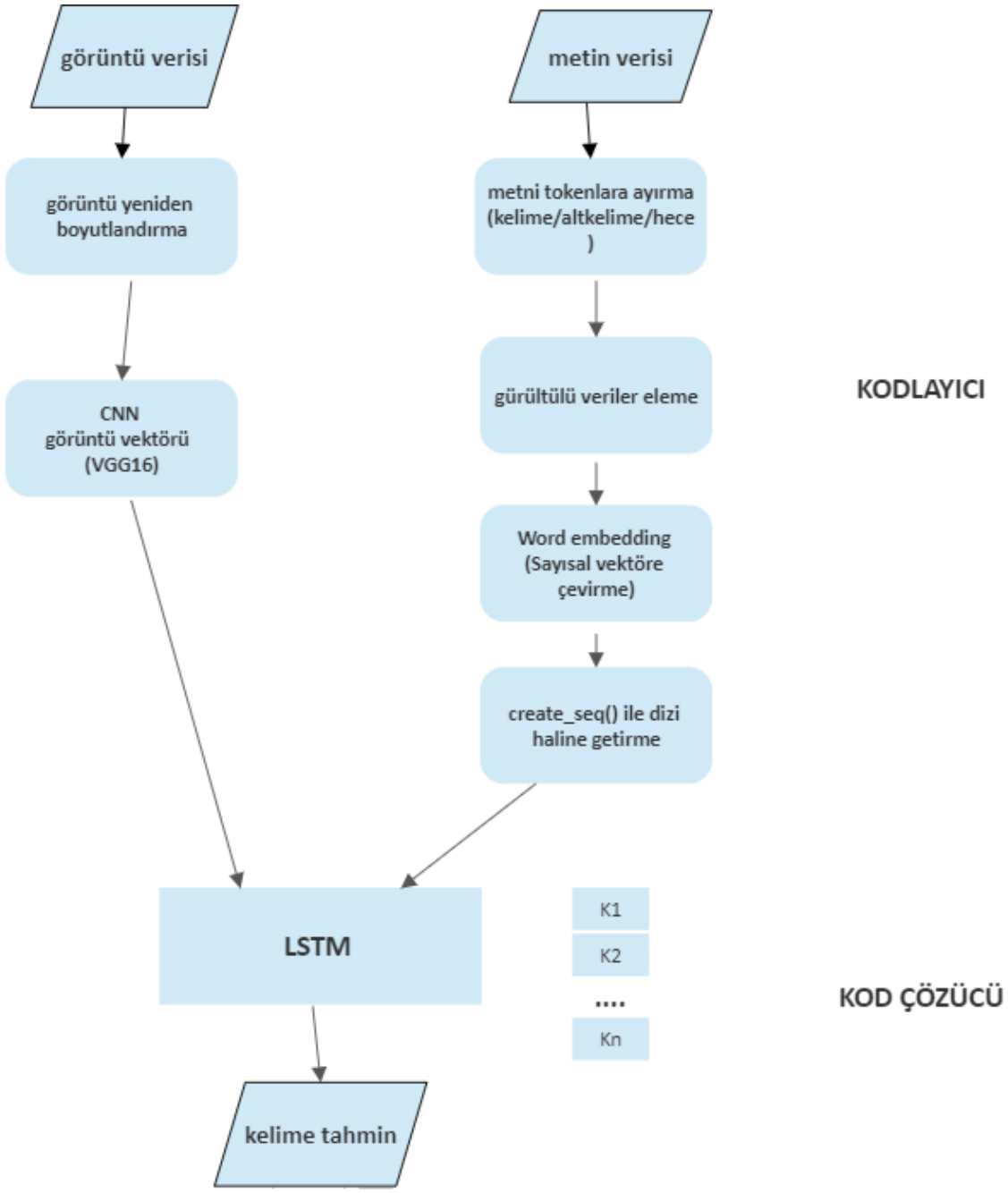


Şekil 3.13. Word2vec Mimarisi [53]

Mikolov ve diğerleri tarafından CBOW (Continuous Bag of Words) ve Skip-gram olmak üzere iki farkli Word2vec yöntemi geliştirilmiştir. Skip-gram yönteminde modele girdi olarak tek kelime verilir ve çıktının bu kelimeye anlamsal olarak yakın olan kelimelerden oluşması beklenmektedir. Giriş ve çıkış katmanlarının yer değiştirmesi sonucunda bu iki yöntem ortaya çıkmaktadır. CBOW yönteminde ise modele girdi olarak birden fazla kelime verilir ve bu kelimelere anlamsal olarak en yakın kelimenin model çıktısı olması beklenmektedir. CBOW, Skip-gram yöntemine kıyasla daha az işlemci gücü gerektirirken birden fazla anlamı olan kelimelerin öğrenilmesinde Skip-gram daha başarılıdır.

4. YAPILAN ÇALIŞMA

Tez kapsamında gerçekleştirilen çalışmada görüntü arşivlerinde bulunan görsellerin hızlı bir şekilde aranıp sorgulanabilmesi için gerekli olan üst veriler başka bir deyişle görüntüyü betimleyen alt yazılar tahmin edilmeye çalışılmıştır. Görüntü alt yazılama yapılırken kelimenin en önemli temel bileşeni olan hecelerden faydalanılarak alt yazı oluşturan bir model önerilmiştir. Önerilen model veri kümesindeki metin verisinin kelimelerine ve alt kelime/köklerine ayrılarak öğrenme modeline verildiği iki ayrı modelle kıyaslanmıştır. Tüm modeller seçilen veri kümeleri üzerinde uygulanmış ve başarımları değerlendirilmiştir. Önerilen modelin başarımı araştırmacıların açık erişimine sunulan hazır veri kümeleri (Flickr8k, Flickr30k) üzerinde denendikten sonra bir kamu kuruluşundan elde edilen arşiv görüntüleri üzerinde de test edilmiştir. Böylece önerilen modelin, üzerinde hiç çalışılmamış bir veri kümesi üzerindeki başarımı ölçülebilmiştir. Görüntü alt yazılama modellerinin eğitim veri kümesi hem görsel veriler hem de o görseli tasvir eden metin verilerinden oluşmaktadır. Bu sebeple veri kümesi üzerinde eğitime başlanmadan önce hem görseller üzerinde hem de metinler üzerinde ön işleme yapılmasına ihtiyaç duyulmuştur. Her üç veri kümesi de %80 eğitim, %20 test olacak şekilde ikiye ayrılarak görüntü alt yazılama modeli eğitilmiş olup önerilen modele ait akış diyagramı Şekil 4.1.'de paylaşılmaktadır.



Şekil 4.1. Modelin Akış Diyagramı

4.1. Veri Kümesi

Görüntü alt yazılama problemlerinin popüler olmasıyla birlikte bu alanda kullanılmak üzere literatüre kazandırılan birçok veri kümesi bulunmaktadır. Bu veri kümeleri çoğunlukla kitle kaynaklı (crowd-sourced) yaklaşımlar kullanılarak oluşturulmuştur. Oldukça maliyetli olmasına rağmen veri seti oluşturmada bu yöntemin tercih edilmesinin temel sebebi, üretilen açıklamaların görüntünün içeriğiyle fazlasıyla

tutarlı olması ve az gürültü içeriyor olmasıdır. Hecelere dayalı görüntü alt yazılama modelinin tanıtıldığı bu çalışmada, araştırmacıların açık erişimine sunulan Flickr8k ve Flickr30k veri kümelerinin yanı sıra bir kamu kuruluşuna ait arşiv görüntüleri kullanılmıştır. Görüntüler toplanırken arşiv taraması yapılmış çok tanınan kişileri veya yerleri içermemesine özen gösterilerek içinde maksimum bilgiyi barındıracak görüntülerin seçilmesine dikkat edilmiştir. Bir videoda her 5 saniyede anlamlı bir değişim olacağı varsayılarak, her 120 karede (frame) bir kare çekecek şekilde bir kod parçacığı yazılmıştır. Toplanan bu frame kümesi üzerinde manuel alt yazılama yapmak üzere 10 kişiden oluşan bir ekip gönüllü olarak çalışmıştır. Görüntü kümesinde toplam 2000 adet fotoğraf bulunmaktadır ve görüntü kümesindeki her bir fotoğraf için üretilen alt yazı sayısı 2 ile 5 arasında değişmektedir. Yazılan her bir alt yazı en az 2 en çok 12 kelimeden oluşmaktadır. Veri kümesiyle ilgili istatistiki bilgi Tablo 4.1.'de verilmektedir.

Tablo 4.1. Veri Kümesi İstatistikleri

Görüntü Sayısı	2000
Açıklamalardaki Kelime Sayısı	En az 2, en çok 12
Görüntü Başına Üretilen Açıklama Sayısı	En az 2, en çok 5

Not. Görüntülere ait açıklamalar toplam 10 kişi tarafından 2 ayda yazılmıştır.

Bu çalışmada önerilen hecelere dayalı alt yazı oluşturma modelinin literatürde yaygın olarak kullanılan [37][38] veri kümeleri üzerindeki başarımını ölçebilmek amacıyla bu çalışma için oluşturulan veri kümesine en yakın sayıda veriye sahip, görüntü alt yazılama problemleri için hazırlanmış öncü veri kümeleri olan Flickr8k [39] ve Flickr30k [40] seçilmiştir. Flickr30k yaklaşık 32 bin görüntüden oluşmaktadır ve Flickr8k veri kümesindeki görüntüleri de içinde barındıran daha büyük ölçekli bir veri kümesidir. Flickr8k veri kümesi eğitim, doğrulama ve test verilerine ayrılmış şekilde araştırmacıların kullanımına sunulmuştur. Ancak Flickr30k'da veri kümesindeki resimler Flick8k'nın aksine eğitim, doğrulama, test şeklinde işaretlenmemiştir ve eğitim öncesi ayırma işleminin bir oran belirlenerek yapılması araştırmacılara bırakılmıştır. Flickr8k ve Flickr30k veri kümelerinde her görüntü için beş adet alt yazı mevcuttur [10]. Türkçeye özgü bir çalışma yürütüldüğünden Flickr8k ve Flickr30k veri kümesindeki alt yazıların Türkçeye çevrilmesi ihtiyacı oluşmuştur. Samet ve diğerleri Flickr30k ve MSCOCO üzerinde yaptıkları çalışma sonrası "Otomatik tercüme ile görüntü alt yazılama için iyileştirmelere açık bir veri kümesi oluşturulabileceği" sonucunu ortaya koymuşlardır [11]. Bu çalışma sonucundan yola çıkılarak Flickr8k ve Flickr30k veri kümesindeki alt yazılar Google Translate uygulaması

yardımıyla Türkçeye çevrilmiştir. Bir önceki paragrafta bahsedilen bu çalışma için manuel oluşturulan veri kümesiyle eşit sayıda veri elde edebilmek amacıyla Flickr8k'dan rastgele 2000 adet görüntü seçilmiştir. Böylece önerilen modelin eşit sayıda veri içeren iki farklı veri kümesi üzerindeki başarımlarının kıyası yapılabilecektir. Rastgele görüntülerin ve onlara ait açıklamaların seçilmesinde kullanılan kod parçacığına ait sözde kod Tablo 4.2.'de gösterilmiştir.

Tablo 4.2. Rastgele veri kümesi oluşturulmasına ait sözde kod

```
karistir(Flickr8k_veri_kümesi)
secilen_goruntu_isimleri = Flickr8k_veri_kümesi[:2000]
yeni_metin_veri_kumesi = []
for goruntu_isim in secilen_goruntu_isimleri:
        aciklamalar = aciklamalari_cek(goruntu_isim)
        yeni_metin_veri_kumesi.extend(aciklamalar)
yeni_dizin = "yeni_goruntuler/"
for goruntu_isim in secilen_goruntu_isimleri:
        goruntu_kopyala(goruntu_isim, yeni_dizin)
rastgele_veri_kumesi = rastgele_sec(yeni_metin_veri_kumesi, 2000)
for veri in rastgele_veri_kumesi:
        goruntu = goruntu_cek(veri)
        metin = metni_cek(veri)
        model_girdi_olarak_ver(goruntu, metin)
```

Modelin eğitiminde kulanılacak veri kümeleri ön işleme aşamalarından geçirildikten sonra sırayla önerilen modele girdi olarak verilmişlerdir. Görüntülere ait metin verileri modellere girdi olarak verilmeden önce token_yarat metodu ile daha küçük parçalarına ayrılmıştır. Tokenlarına ayırma aşamasında eğitim verisi ilk model için boşluk karakteri kullanılarak kelimelere, ikinci model için alt kelime/köklerine, son model için ise hecelerine ayrılmıştır. Sonrasında sayısal vektörlere çevrilerek her bir parçaya benzersiz/tekil bir sayı atanmıştır. Böylece metin verileri öğrenme modeline girdi olmadan önce sayısal vektörlerle ifade edilmiştir. Model girdi olarak görüntülere ait öznitelikler ile sayısal vektörlerle ifade edilmiş metin dizisini almakta ve çıktı olarak bir sonraki kelimeyi tahmin etmektedir. Bir sonraki kelime tahmin edilirken model, sözlükteki tüm kelimeler için olasılıksal bir değer hesaplamakta ve bu değere göre çıktıya karar vermektedir.

4.2. Önerilen Model

Görüntü alt yazılama problemlerinde Türkçe dil yapısını gözeten az sayıda çalışma bulunmaktadır. Bu tez çalışmasında Türkçe dilinde kelimenin en temel bileşeni olan hecelerin içerisinde taşıdığı bilgiyi öne çıkaracak bir model önerilmiştir. Türkçeye özel oluşturulan hecelere dayalı alt yazılama modelinin geliştirilmesi için Tensorflow kütüphanesi kullanılmıştır. TensorFlow, Google tarafından geliştirilen, 2015 yılında piyasaya sürülen açık kaynak bir makine öğrenimi ve derin öğrenme kütüphanesidir. Yapay zeka, doğal dil işleme, görüntü işleme gibi çeşitli makine öğrenimi ve derin öğrenme projelerinde kullanılmak üzere tasarlanmıştır. Python, C++, Java, Go gibi dillerde kullanım desteği bulunmaktadır. Görüntü verisi üzerinde gerçekleştirilen ön işleme aşamaları temel olarak görüntünün seçilen CNN mimarisi girişine uygun boyuta getirilmesi ve öznitelik vektörünün çıkarılmasıdır. Metin verisinin ön işleme aşamasında metnin hecelere ayrılması için Python, alt kelime/köklerine ayrılması için Java dili kullanılmıştır. Önerilen hecelere dayalı görüntü alt yazılama modelinin başarımı literatürde kullanılan kelimelere dayalı ve alt kelime/köklere dayalı modeller ile kıyaslanmıştır. Kelimelere dayalı modelde eğitim verisindeki her bir açıklama boşluk karakteri kullanılarak kelimelerine ayrılmaktadır. Alt kelime/kök modelinde ise Zemberek kütüphanesinden faydalanılarak kelimeler kök ve eklerine ayrıştırılmış sonrasında anlamlı en büyük kelime alt kelime olarak belirlenmiştir. Heceleme işleminin nasıl gerçekleştirildiği bir sonraki başlıkta detaylı olarak verilmektedir. Veri kümesindeki metin ve görseller üzerinde veri ön işleme aşamaları tamamlandıktan sonra vektör haline getirilen veriler modele girdi olarak verilmiştir.

4.2.1. Heceleme yöntemi

Eğitim veri kümesindeki cümlelerin anlaşılabilmesi için öncelikle metin içerisindeki kelimeler tespit edilmektedir. Kelimeler tespit edilirken boşluk karakterinden faydalanılmıştır. Kelimeler, metin içerisinde tespit edildikten sonra iki şekilde daha ayrıştırılabilir. Birincisi kelimenin hecelenmesi ikincisi ise kelimenin ek ve köklerine morfolojik olarak ayrıştırılması işlemidir. Ayrıştırma işlemi kelimenin yanlış yazılıp yazılmadığının anlaşılmasında önemli bir bilgi sağlamaktadır. Çünkü hecelerine ve/veya ek-köklerine ayrılamayan bir kelimenin Türkçe dil kurallarına uygun olmadığı veya yanlış yazıldığı anlaşılmaktadır. Ayrıca ekler ve kökler ile ilgili ses olayları hece düzeyinde gerçekleştiği için morfolojik ayrıştırma öncesinde kelimenin hecelerine ayrılabilmesi önemlidir.

Önceki bölümlerde belirtildiği üzere görüntü alt yazılama problemlerinde İngilizce için birçok model geliştirilmiş olsa da Türkçe dilinin yapısal özellikleri nedeniyle bu modeller doğrudan alınıp uygulanamamaktadır. Dil modelleri önceden belirlenen sabit bir sözlük içerisinden sözcükler seçerek alt yazı üretmektedirler. Dolayısıyla sözlüklerinde bulunmayan ve öncesinde hiç karşılaşılmamış sözcükleri (out-of vocabulary word-OOV) alt yazı oluştururken kullanmaları olası değildir. Heceler sayesinde OOV kelimelerin bileşenlerini anlamak ve daha tanıdık kelimelerle ilişkilendirmek mümkün olmaktadır. Ayrıca heceler Türkçe dilinde bir kelimenin kök, çekim ekleri ve ekler gibi morfolojik özelliklerinin anlaşılmasında katkı sağlamaktadır. Dolayısıyla hecelerin kullanılması, dil işleme modellerinin dilbilgisi kurallarını daha doğru uygulamasına olanak sağlayacaktır. Literatürde bulunan önceki çalışmalarda Türkçe için metin verisi doğrudan kullanılarak veya metinde geçen kelimeleri köklerine ayırarak görüntü alt yazılama modelleri denenmiştir. Bu çalışmada, sayılan diğer iki yöntemde karşılaşılması olası "çok anlamlılığı yitirme" sorununa çözüm olabilecek hecelere dayalı alt yazılama modeli önerilmiştir. Bu kısımda belirtilen "çok anlamlılık" kavramı hecelerin sıralanışına göre kelimelerin farklı anlamlar kazanması şeklinde açıklanabilir. Örneğin "ka-me-ra" kelimesi ile "ma-ka-ra" kelimesindeki heceler fonetik olarak ve içerdikleri karakterler bakımından benzerdir ancak hecelerin sıralanışı kelimelere farklı anlamlar yüklemektedir. Heceler bilgiyi kodlamaktadır dolayısıyla hecelerin öğrenme modellerinde kullanılıyor olması hecelerin farklı kombinasyonundan oluşan kelimelerin yakalanmasında önemli bir etmen olmaktadır.

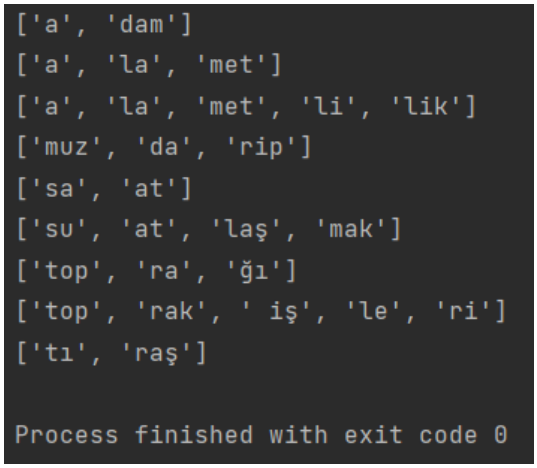
Hece, dilin ses birimlerinin en küçüğü olup bir araya gelerek kelimeleri oluşturan ve ağızdan tek bir çıkışta tamamlanabilen ses modeli olarak tanımlanabilmektedir [54]. Dilin yapısı gereği Türkçede heceler bilginin kodlandığı yerdir ve önemli bir role sahiptir. Hecelerde kodlanan bilgi, dilin yapısını ve anlamını kavramak açısından son derece önemlidir. Türkçede heceler çoğunlukla en az bir, en çok dört harften oluşmakta ve içerisinde en fazla bir sesli harf bulundurmaktadır [54]. Yabancı dillerden Türkçeye geçen kelimelerde ise heceyi oluşturan harf sayısı artabilmektedir. (Örnek: Prens, kramp gibi) Yabancı dillerden Türkçeye geçen kelimeler de Türkçenin hece yapısına göre hecelerine ayrılmaktadır [55]. Bazı dillerde bir sesli ve sekiz sessiz olmak üzere toplam dokuz harften oluşan heceler bulunmaktadır [41]. Bir kelimenin kök, çekim ekleri ve ekler gibi morfolojik özelliklerinin anlaşılmasında heceler önemli bir rol oynamaktadır. Bu çalışmada veri kümesindeki kelimeler hecelerine ayrılırken literatürde bulunan heceleme algoritmaları incelenmiş ve Tablo 4.3.'te sözde kodu paylaşılan bir kod yazılmıştır.

Tablo 4.3. Heceleme algoritmasına ait sözde kod

```
fonksiyon hece_bul(metin):
    heceler = boş liste
    kelimeler = metin.split()
    for kelime in kelimeler:
        hece = ''
        unluler_bulundu = False
        i = len(kelime) – 1
        while i >= 0:
            harf = kelime[i]
            hece = harf + hece
            if harf in unlu_harfler:
                unluler_bulundu = True
                i -= 1
                while i >= 0 and kelime[i] in unsuz_harfler:
                    hece = kelime[i] + hece
                    i -= 1
                if i >= 0 and kelime[i] in unlu_harfler:
                    continue
            elif harf in unsuz_harfler and unluler_bulundu:
                heceler.append(hece)
                hece = ''
            i -= 1
return heceler
```

["adam", "alamet", "alametlilik", "muzdarip", "saat", "suatlaşmak", "toprağı", "toprak işleri", "tıraş"] kelimelerine karşılık bulunan heceler Şekil 4.2.'de gösterilmiştir. ***"alamet"*** kelimesi üzerinden heceleme algoritmasının işleyişi ele alınırsa, ilk olarak hece isimli boş bir liste oluşturulmaktadır. Sonrasında kelimenin sağından başlanarak birer birer kelimeye ait karakterler alınmaktadır. Bu durumda karşılaşılan ilk harf "t" olacaktır. Bu harf ünsüz harfler listesinde bulunduğundan hece isimli boş listeye alınmakta bir sonraki harfi almak üzere bir adım sola kayılmaktadır. İkinci olarak elde edilecek harf olan "e" ünlü harfler listesinde bulunduğundan hece listesine dahil edilir ve bir adım sola kayılır. Kelimeye ait heceler bulunurken içerisinde sadece bir ünlü harf barındıran en küçük karakter kümesi aranmaktadır. Bir sonraki harf olan "m" ünsüz harfler listesinde bulunduğundan hece listesine dahil edilmektedir. Bir adım daha sola kayıldığında gelen harf "a" olacaktır ve ünlü bir harf geldiğinden ve bir hece sadece bir tek ünlü harf barındırabileceğinden, hece bir

önceki haliyle “met” şeklinde sonlandırılacaktır. İki ünsüz arasına sıkıştırılan bir ünlü harf modeli yakalandığında ilk hece tamamlandı adayı olarak işaretlenmektedir. Kelime içerisindeki bir sonraki heceyi bulmak üzere yinelemeli olarak sola kaymaya devam edilmektedir. Benzer şekilde “la” hecesi de bir ünlü harfin ünsüz harf ile kapatılması şeklinde elde edilmektedir. Son olarak gelen “a” harfinden sonra gelecek bir harf bulunmadığından son hece “a” harfinden oluşturularak kelimeye ait hece bulma işlemi sonlandırılmaktadır. Bu durumda “alamet” kelimesine ait heceler “a”, “la”, “met” olarak bulunmaktadır. Özetle kelime üzerinde yinelemeli bir şekilde kayılırken ünlü bir harf bulunana kadar ilerlenmekte ve ünlü harf ünsüz harfler arasına sıkıştırılarak hece yakalanmaktadır. Ünlü harf bulunana kadar gelen ünsüz harfler birlikte bir hece oluşturmak üzere yan yana depolanmaktadır.



Şekil 4.2. Heceleme algoritmasının çıktısı

Önerilen hecelere dayalı öğrenme modelinin işleyişini ve diğer yöntemlere göre avantajını daha iyi açıklayabilmek için “okula” kelimesini örnek olarak ele alalım. Alt kelime/kök modelinde bu kelime eklerinden ayrılarak “okul” şeklinde bir ayrıştırma yapılmaktadır. Aynı kelimeye heceleme yöntemi uygulandığında “o-ku-la” şeklinde bir sonuç elde edilmektedir. Daha önceki bölümlerde dil modellerinin önceden belirlenen sabit bir sözlük içerisinden sözcükler seçerek alt yazı ürettiğinden bahsedilmişti. “okul” kelimesi özelinde düşünüldüğünde heceleme yöntemi kullanılarak sözlüğe “o”, “ku”,”la” heceleri eklenecek ve n-gramlar yardımıyla bu hecelerin birlikte kullanılma kombinasyonundan öğrenme sağlanabilecektir. Heceleme modelinde aynı kökten türeyen fakat dilin yapısı gereği ünlü düşmesi, ünsüz sertleşmesi gibi işlemlere maruz kalan benzer kelimeler yakalanabilmektedir. Ayrıca “okula” kelimesindeki “la” ekinin barındırdığı “yönelme” fonksiyonu kaybedilmeyecektir. Görüldüğü gibi heceler içerisinde daha fazla bilgi

barındıracağından daha esnek, dil yapısına daha uygun metinler oluşturabilecektir. Sayılan bu sebeplerle modeli eğitmeye başlamadan önce veri setindeki alt yazılar üzerinde bir veri ön işleme gerçekleştirilerek kelimeler hecelerine ayrılmış, böylece elde edilen hecelerin birlikte kullanılma şekilleri ve dilin yapısı gereği maruz kaldıkları ünsüz sertleşmesi, ünlü düşmesi gibi kurallar öğrenme sürecine dahil edilebilmiştir. Flickr8k veri kümesi üzerinde uygulanan heceleme algoritması sonucunda oluşan alt yazılar Tablo 4.4.'te gösterilmektedir.

Tablo 4.4. Flickr8k metin verisi üzerine heceleme algoritması uygulaması

Orijinal metin	**Göstergeli ve gözlüklü bir adam Blitz şapka takıyor.**
Hecelerine ayrılmış hali	Gös ter ge li ve göz lük lü bir a dam Blitz şap ka ta kı yor.
Orijinal metin	**Pembeli küçük bir kız parkta bir ip köprüye tırmanıyor.**
Hecelerine ayrılmış hali	Pem be li kü çük bir kız park ta bir ip köp rü ye tır ma nı yor.
Orijinal metin	**Boston teriyeri çimlerde koşuyor .**
Hecelerine ayrılmış hali	Bos ton te ri ye ri çim ler de ko şu yor .
Orijinal metin	**Pembe elbiseli bir çocuk giriş yolunda bir dizi merdiveni tırmanıyor.**
Hecelerine ayrılmış hali	Pem be el bi se li bir ço cuk gi riş yo lun da bir di zi mer di ve ni tır ma nı yor.
Orijinal metin	**Küçük bir çocuk, arkasında bir adam ve metal bir direkle taş döşeli bir sokakta yürüyor.**
Hecelerine ayrılmış hali	Kü çük bir ço cuk, ar ka sın da bir a dam ve me tal bir di rek le taş dö şe li bir so kak ta yü rü yor.
Orijinal metin	**Beyaz köpek yeşil bir alanda sarı bir oyuncakla oynuyor.**
Hecelerine ayrılmış hali	Be yaz kö pek ye şil bir a lan da sa rı bir o yun cak la oy nu yor.

Tablodan da görüleceği gibi veri kümesindeki kelimeler başarılı bir şekilde hecelerine ayrılmıştır. Türkçeye yabancı dillerden geçen kelimeler veya "Blitz" gibi yabancı kelimeler de yukarıda sözde kodu verilen heceleme algoritması ile doğru bir şekilde hecelerine ayrılmaktadır. Kullanılan heceleme algoritmasının başarımına ait bilgiler, veri kümesinden rastgele seçilen ve toplam 852 kelimeden oluşan bir örnek küme üzerinde Tablo 4.5'te gösterilmektedir. Kelimelerin her biri en kısa 3, en uzun 17 kelimeden oluşan 100 cümleden alınmış olup içerisinde hem Türkçe kökenli hem de yabancı dillerden Türkçeye geçen kelimeleri barındırmaktadır.

Tablo 4.5. Heceleme Algoritmasının Örnek Veri Kümesi Üzerinde Hesaplanan Başarımı

Toplam Kelime Sayısı:	852
Hatalı Hecelenen Kelime Sayısı	10
Doğru Hecelenen Kelime Sayısı	842
Doğruluk oranı	**%98.8**

Veri kümesindeki tüm kelimeler hecelerine ayrıldıktan sonra doğal dil işleme konusunda bir diğer önemli başlık olan kelime ön işleme yöntemlerine tabi tutularak içerdikleri gürültüden arındırılmış ve sonrasında sayısal vektörlerle ifade edilmiştir. Kelime temsil yöntemlerinde kelimeler derin öğrenme modeline girdi olmadan önce, tıpkı görüntüler gibi ön işlemden geçirilmekte ve kelime vektörü denilen sayısal karşılıklarıyla temsil edilmektedir. Kelime temsil yöntemleri veri ön işleme, öznitelik çıkarma ve öğrenilen temsilleri uygun bir sınıflandırıcı yardımıyla sınıflandırma olmak üzere 3 temel başlığa ayrılabilmektedir.

4.2.2 Metin Ön İşleme (Text Preprocessing)

Metin tabanlı veriler, içerisinde çok fazla bilgi taşımakla birlikte verilerin toplanma şekliyle ilintili olarak bünyesinde gereksiz kelimeleri, bazı yazım hatalarını veya argo kelimeleri barındırabilmektedir. Bu tür gürültülü veriler metinlerin işlenmesi aşamasında zorluk yaratacağı gibi başarısız bir sınıflandırma sürecine de neden olmaktadırlar. Bu sebeple metin tabanlı veriler modellere girdi olmadan önce üzerlerinde ön işleme yapılarak bu tür gürültülerinden arındırılmalıdır. Bu çalışmada da hem kendi oluşturduğumuz veri kümesindeki metinler hem de hazır veri kümelerine ait metinler aşağıda sayılacak işlemlere tabi tutularak gürültülerinden temizlenmiştir. Barındırdığı gürültülerden temizlenen metin verisi görüntü ismi (Örn: resim_01) ve karşılığında açıklaması olacak şekilde dictionary nesnesine dönüştürülmüş ve "aciklama.txt" isimli dosyaya kaydedilmiştir.

- **Kelimelerin küçük harfe dönüştürülmesi:**

Bir kelimenin büyük harf içeren versiyonu ile küçük harf içeren versiyonu makineler tarafından farklı kelimelermiş gibi algılanarak birden fazla kopya şeklinde tutulabilir. Bu davranış öğrenme başarımını olumsuz yönde etkilemektedir bu sebeple metinler modellere girdi verilmeden önce küçük harflere dönüştürülmektedir. Bu işlem için aciklama.txt dosyasındaki tüm kelimeler lower() metodu kullanılarak küçük harfe dönüştürülmüştür.

- **Noktalama işaretlerinin kaldırılması:**

Noktalama işaretleri, üzerlerinde özel bir çalışma gerçekleştirilmiyorsa makineler tarafından anlamlandırılabilen veriler değildir. Bu sebeple veri kümesi boyutunu küçültmek ve modelin başarımını olumsuz etkilememeleri amacıyla noktalama işaretleri metinsel verilerden çıkarılmıştır.

- **Numerik değerlerin kaldırılması:**

Tıpkı noktalama işaretleri gibi numerik değerler de makineler için anlam ifade etmemektedir. Bu sebeple word.isalpha() metodu yardımıyla metinsel veriler numerik değerlerinden arındırılmıştır.

- **Sık tekrarlayan gereksiz kelimelerin çıkarılması:**

İngilizce dilinde "a", "the", "an" gibi kelimeler, Türkçede ise "veya", "ve", "de", "daha" gibi kelimeler metin içinde önemli bir anlama sahip olmayan fakat sık tekrar eden kelimelerdir. Metnin anlamına kritik bir katkıda bulunmayan fakat metin içinde sıklıkla geçen bu kelimelerin temizlenmesi veri kümesinin küçülmesine ve daha hızlı işlem yapılabilmesine olanak tanımaktadır. Bu kelimelerin çıkarılması esnasında genellikle belirli bir sayı (Örneğin:1,2) eşik değeri kabul edilmekte ve bu eşik değerinin altındaki boyuta sahip kelimeler otomatik olarak veri kümesinin dışına atılmaktadır. Bu tez çalışmasında sözcük ve alt kelime/kök modellerine ait metinlerde bu temizleme işlemi yapılmıştır. Ancak hecelemeye dayalı modelde asıl hedef veri kümesindeki bilginin kaybedilmeden öğrenmeye dahil edilmesi olduğundan bilgi kaybına sebep olmamak amacıyla bu işlem yapılmamıştır. Çünkü heceler yapısı gereği düşük sayıda karakterlerle ifade edilmektedir ve belirlenen eşik değerinin altında kalmaları durumunda barındırdıkları bilgi de kaybolacaktır.

- **Kelimeleri küçük parçalara ayırma (Tokenization):**

Doğal dil işleme problemlerinde yaygın olarak kullanılan tokenization, metin içerisindeki cümleleri, kelimeleri veya sembolleri ayrıştırarak daha küçük parçalarına yani "token"larına ayırma işlemine denilmektedir. Tokenization sayesinde metinler makine öğrenimi ve doğal dil işleme problemlerinde daha etkili şekilde kullanılabilmektedir. Bu çalışmada kelime modeli kullanılırken veri kümesindeki metinler, boşluk karakteri dikkate alınarak kelimelerine ayrıştırılmış ve modele kelime halinde verilmiştir. Benzer şekilde alt kelime/kök modeli için de anlamlı en büyük kelime döndürülmüştür. Hecelere dayalı modelde ise tokenlarına ayırma işlemi kelimenin hecelerine ayrılması şeklinde olmaktadır.

- **Gövdeleme (Stemming):**

Gövdeleme, doğal dil işleme alanında kullanılan metin işleme tekniklerinden biridir ve metin içerisinde kelime köklerini bulmayı hedeflemektedir. Aynı kökten gelen fakat aldığı eklerle farklı şekillere dönüşen kelimeler veri kümesi boyutunu istenmeyen şekilde artırmakta ve metinlerin temsilini karmaşık hale getirmektedir. Örneğin, "gidiyordu", "gidelim", "gitmeli" gibi farklı fiil türevleri stemming uygulanarak "git" kelimesine dönüştürülebilmekte ve metindeki kelime sayısı azaltılabilmektedir. Stemming tekniğinde kelimeye ait kökler bulunmaya çalışılırken eklerin yeri değiştirilmekte veya ekler tamamen atılmaktadır. Ancak bu yöntem her zaman başarılı sonuçlar vermemekte hatalı kökler bulunmasına sebep olmaktadır.

- **Kök Çözümleme (Lemmatization):**

Lemmatization da tıpkı stemming gibi aynı kökten gelen kelimeleri tespit edebilmek için kelimeye ait kökleri bulmayı hedeflemektedir. Stemming yönteminde kökler bulunmaya çalışılırken basitçe ekleri kesme işlemi yapılmaktaydı. Lemmatization ise dilbilgisel kuralları daha çok dikkate alan bir teknik olduğundan stemming yöntemine göre daha başarılı sonuçlar üretmektedir. Kelimelerin kökünü bulmaya çalışırken sözlük bilgisinden faydalanmakta ve kelimelerin temel formlarını bulmayı hedeflemektedir. Böylece aldığı ek ile birlikte ünlü düşmesi ünsüz benzeşmesi gibi işlemlere maruz kalan kelimelerin köklerini daha doğru bir şekilde bulabilmektedir. Bu çalışmada sözcüklere ait alt kelime/kökler bulunarak hecelere dayalı model ile başarısı kıyaslanacağından lemmatization yönteminden faydalanılmış ve kelimeyi temsil edecek alt kelimeler elde edilmiştir. Tablo 4.6.'da Flickr8k veri kümesindeki orijinal metinler ve lemmatization işlemi sonucu cümlelerin dönüştüğü form gösterilmiştir.

Tablo 4.6. Flickr8k Metin Verisi Üzerine Lemmatization Uygulaması

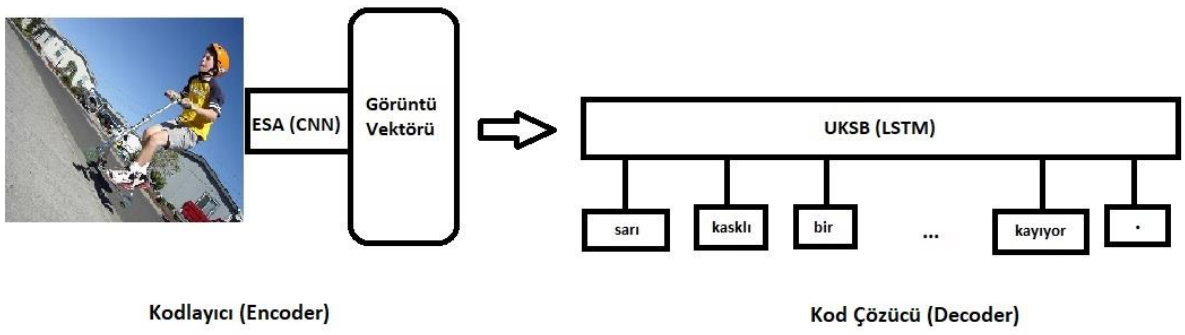
Orijinal metin	**Göstergeli ve gözlüklü bir adam Blitz şapka takıyor.**
Altkelime/kök hali	gösterge li ve gözlük lü bir adam blitz şapka tak ıyor
Orijinal metin	**Pembeli küçük bir kız parkta bir ip köprüye tırmanıyor.**
Altkelime/kök hali	pembe li küçük bir kız park ta bir ip köprü ye tırmanıyor
Orijinal metin	**Boston teriyeri çimlerde koşuyor** .
Altkelime/kök hali	boston teriyer i çim lerde koşu yor
Orijinal metin	**Pembe elbiseli bir çocuk giriş yolunda bir dizi merdiveni tırmanıyor.**
Altkelime/kök hali	pembe elbise li bir çocuk giriş yol unda bir dizi merdiven i tırmanıyor
Orijinal metin	**Küçük bir çocuk, arkasında bir adam ve metal bir direkle taş döşeli bir sokakta yürüyor.**
Altkelime/kök hali	küçük bir çocuk arka sında bir adam ve metal bir direk le taş döşeli bir sokak ta yürüyor
Orijinal metin	**Beyaz köpek yeşil bir alanda sarı bir oyuncakla oynuyor.**
Altkelime/kök hali	beyaz köpek yeşil bir alan da sarı bir oyuncak la oynuyor

Lemmatization kavramı, literatürde kelime ön işleme yöntemi olarak geçmektedir ve kelimelerin daha düşük boyutlu bir vektöre dönüştürülmesi işlemi öncesinde kelimelerin köklerine ayrıştırılması amacıyla kullanılmaktadır [42].

Kelimelerin sayısal vektörlere dönüştürülmesi kelime gömmeleri (word embedding) yöntemleri ile yapılmaktadır ve bu sayede birbirine anlam olarak yakın kelimelerin yakalanmasına olanak sağlanmaktadır. Örneğin mor, turuncu gibi kelimeler temelde renkleri ifade ettiklerinden vektör uzayında birbirlerine daha yakın görünmektedirler. Lemmatization işlemi ile kelimeler köklerine ayrıldıktan sonra word embedding kullanılarak kelime benzerliği korunabilmekte düşük boyutlu vektörlerle anlamlı temsiller sağlanabilmektedir.

4.2.3. Derin Öğrenme Modeli

Önceki bölümlerde, son yıllarda derin öğrenme modellerinde yaşanan gelişmeler sonucunda görüntü alt yazılama problemlerinin makine çeviri problemlerine benzer şekilde kodlayıcı-kod çözücü mimarisi kullanılarak çözüldüğünden bahsedilmiştir. Görüntü kodlayıcı olarak CNN kullanılmakta ve görüntüdeki nesneleri tanımak için görüntüye ait öznitelik vektörü çıkartılmaktadır. Burada elde edilen görüntü vektörünün maksimum bilgi içermesi ve görüntüyü en iyi şekilde temsil etmesi gerekmektedir [16]. Temelde Evrişimli Sinir Ağları sınıflandırma problemleri için tasarlanmıştır ve çıktı olarak bir sınıf tahmini yapmaktadır. Ancak en sondaki sınıflandırma katmanı çıkarılarak görüntüye ait öznitelik vektörünü elde etmek mümkündür. Sonrasında RNN tabanlı bir kod çözücü, dil modeli olarak çalışmakta ve CNN tarafından üretilen öznitelik vektörü ile metinlerden oluşturulan kelime vektörünü kullanarak görüntüyü açıklayan bir metin oluşturmaktadır. Kod çözücü her bir adımda, bir önceki adımda üretilen kelimeyi kullanarak bir sonraki kelimeyi tahmin etmektedir. Şekil 4.3'de kodlayıcı kod çözücü mimariye ait akış gösterilmektedir.



Şekil 4.3. Kodlayıcı Kod Çözücü Mimari

4.2.4. Görüntü özniteliklerinin çıkarılması

Literatürde kodlayıcı olarak kullanılabilecek birçok CNN modeli bulunmaktadır. Eğitim sırasında bu hazır modellerin kullanılması öğrenme aktarımı (transfer learning) olarak geçmektedir [43]. Öğrenme aktarımının avantajı sıfırdan bir eğitim gerçekleştirmek yerine daha önceden öğrenilen bilgileri kullanarak daha az veri ile daha hızlı ve yüksek başarım gösteren modeller elde ediliyor oluşudur [44]. Bu çalışmada, kodlayıcı olarak 2014 yılındaki ImageNet yarışmasında en iyi 5 modelden biri olan VGG16 mimarisi kullanılmıştır. oznitelik_cikar() isimli metod ile görüntüler son katmanı çıkarılan VGG16 modeline girdi olarak verilmiş, görüntüye ait öznitelikler çıkarılarak dictionary nesnesine dönüştürülmüştür. Veri kümesindeki görüntülere ait elde edilen öznitelikler "pkl" uzantılı dosyaya kaydedilmiştir. Her bir veri kümesi için öznitelik çıkarma işlemi bir kere yapılmakta ve görüntü özniteliklerini barındıran ".pkl" uzantılı dosya kelime, alt kelime/kök ve hece modellerinde doğrudan kullanılabilmektedir.

VGG16 Mimarisi

VGG16 modeli 1000 farklı sınıf ve 14 milyon görüntüden oluşan bir veri kümesi üzerinde eğitilmiştir. VGG16 mimarisi 224x224x3 boyutunda görüntüler üzerinde çalışmaktadır dolayısıyla veri ön işleme aşamasında veri kümesindeki tüm görüntüler bu boyuta gelecek şekilde yeniden boyutlandırılmıştır. Modelde overfitting yani aşırı uyum sorununu yaşamamak amacıyla 0,5 oranında dropout uygulanmıştır. Ek olarak VGG16 modelinde optimizasyon algoritması olarak Adam (Adaptive Moment Estimation) seçilmiştir. Optimizasyon algoritmaları derin öğrenme modellerinde geri yayılım (backpropagation) algoritmalarını kullanarak ağırlıkların nasıl güncelleneceğine karar vermektedir. Adam derin öğrenme hızını otomatik olarak ayarlayarak verimli bir gradyan inişi gerçekleştiren ve derin öğrenme modellerinde başarılı sonuçlar veren bir algoritmadır.

Metin üretme kısmında geleneksel yöntem n-gramlardan faydalanarak bir sonraki kelimenin tahmin edilmesidir. N değerinin 3 olarak belirlendiği n-gram yönteminde, sıradaki kelime tahmin edilirken kendinden 2 önceki kelimeye bakılmaktadır. ***"Bugün hava …"*** şeklinde başlayan bir cümle örneğinden gidilirse; veri setinde bulunan kelimelerden "sıcak", "kapalı", "soğuk" aday kelimeler olarak öne çıkmaktadır. Tüm veri seti içerisinde ***"Bugün hava sıcak"*** ifadesinin geçme sayısının ***"Bugün hava…"*** ifadesinin geçme sayısına bölünmesiyle bir sonraki kelimenin "sıcak" olma olasılığı hesaplanabilmektedir. Bu durumda n değerinin büyük seçilmesi daha doğru tahminler yapılmasını sağlamaktadır ancak yüksek bir hesaplama maliyeti getirmektedir. Derin sinir ağlarında yaşanan gelişmeler

sonrasında n-gramlar yerini hafıza hücrelerine sahip öğrenme modellerine bırakmıştır. Böylece hesaplama maliyeti yaşanmadan uzun cümleler üzerinde çalışılabilmektedir.

Bu çalışmada çıktıda anlamlı ve sıralı bir metin beklendiğinden bir önceki adımdaki bilgilerin tutuluyor olması önemlidir. Bu sebeple kod çözücü olarak RNN tabanlı bir mimari seçilmiştir. Temelde tekrarlayan sinir ağları bir önceki adımın çıkışını mevcut adımın girişi olarak kullanan, önceki girişi ve mevcut girişi hatırlama üzerine kurulmuş bir mimaridir ve içerisinde bellek barındırma özelliği bulunmaktadır. Teoride RNN'lerin uzun dizilerde (long-sequence) başarılı sonuçlar elde etmesi beklenmektedir ancak kısa vadeli bir hafızaya sahip olduğundan pratikte bu tür problemlerde yeterince başarılı çalışmadığı görülmektedir. Ayrıca geriye yayılım (backpropagation) sırasında Gradyan Kaybolması (Vanishing Gradient) problemi de yaşanmaktadır. Gradyan Kaybolması, geriye yayılım sırasında gradyan değerinin çok fazla küçülerek zamanla kaybolması sonucu öğrenme problemleri yaşanmasıdır. RNN'de yaşanan uzun vadede unutma problemlerinin önüne geçmek amacıyla LSTM modelleri önerilmiştir [45]. Bu sebeple çalışmanın kod çözücü kısmında ReLU aktivasyon fonksiyonu seçilerek RNN tabanlı LSTM mimarisi kullanılmıştır.

LSTM'in mimarisi, bir hafıza hücresi, bir giriş kapısı, bir unutma kapısı ve bir çıkış kapısından oluşmaktadır. LSTM içerisinde bulunan bu yapılarla bilgileri daha iyi saklamakta ve klasik RNN'lerin bellek problemini ortadan kaldırmayı hedeflemektedir. LSTM mimarisinde bulunan hafıza hücresi, ağ mimarisinde bulunan giriş, unutma ve çıkış kapıları sayesinde hangi verilerin tutulacağına ya da hangi verilerin unutulacağına/silineceğine karar vermektedir [46]. Cell State ise hücre içindeki taşınması gereken bilgileri alarak hücre sonuna, oradan da diğer hücrelere taşıyan bir yapıdır. Cell state'in taşıması gereken bilgileri belirlemek hücre içindeki kapıların görevidir. Bu kapılar hangi bilgilerin gerekli hangilerinin gereksiz olduğunu verileri 0 ile 1 arasına sıkıştıran sigmoid aktivasyon fonksiyonu yardımıyla belirlemektedir. Sigmoid fonksiyonu sonucu 0 olan bilgiler unutulurken 1 olan bilgiler ise Cell State ile birlikte bir sonraki hücreye ilerlemeye devam etmektedir.

4.2.5. Kelime Vektörü ve Görüntü Vektörünün Modele Beslenmesi

Önerilen hecelere dayalı modelin başarımını kıyaslayabilmek için sadece sözcüklerden oluşan metin verisi, alt kelime/köklerden oluşturulmuş metin verisi ve hecelerine ayrılmış metin verisine ait kelime vektörleri "Word Embedding" kullanılarak ayrı ayrı çıkarılmıştır. Kelime vektörüne dönüştürülmeden önce ***load_aciklama()*** metodu ile metin verisi görüntü adı ve karşılığında ona ait açıklamalar olacak şekilde dictionary

nesnesine dönüştürülmüştür. ***aciklama_temizle()*** metodu ile metin içerisinde bulunan rakamlar, noktalama işaretleri elenmiş büyük harfler küçük harfe çevrilmiştir. Metin verisi üzerinde gerçekleşen temizleme işlemi sonrasında ***temizaciklama_load()*** metodu ile metin verisi tekrar yüklenmiştir. Metin verisi içindeki açıklamaların başına <startseq> sonuna ise <endseq> belirteci konularak veri kümesi içindeki açıklamaların başlangıç ve bitişinin nerede olduğu işaretlenmiş ve sonrasında ***create_tokenizer()*** metodu ile metin verisine ait kelime vektörleri çıkarılmıştır. Metin verisinin sayısal değerlere dönüştürülmesi için ***keras*** kütüphanesinin ***tokenizer*** sınıfı kullanılmıştır. Korpusta bulunan her kelime için bir index değeri atanmakta ve sonrasında kelimeler o index değerleriyle değiştirilmektedir. Metin verilerini modele doğru sırada verebilmek için ***create_sequence()*** metodu ile açıklamalar diziye dönüştürülmektedir. Metin verilerindeki açıklamalar farklı sayıda kelime içerdiğinden ***"0 padding"*** işlemi ile her açıklama sabit boyutlu vektör haline getirilmiştir. Benzer şekilde kodlayıcı mimarisine beslenecek görüntüler VGG16 modelinin girdi formatı olan 224x224 boyutuna dönüştürülmüştür. Başarım kıyası yapılacak her üç model için de "aciklama_raw.txt", "aciklama_altsoz.txt" ve "aciklama_hece.txt" dosyaları ve görüntü özniteliklerinin çıkarıldığı ".pkl" uzantılı özellik vektörü oluşturulmuştur. Görüntü öznitelikleri çıkarılırken öğrenme aktarımı yöntemi ile VGG16 modeline ait ağırlıklar kullanılmış, VGG modelinin son katmanı olan sınıflandırma katmanı çıkarılarak sadece öznitelik vektörleri alınmıştır. Alt yazılara ait sayısal vektörler ve kodlayıcı ile elde edilen görüntü vektörleri birleştirme(merge) modeli kullanılarak görüntü alt yazılama için önerilen, kodlayıcı-kod çözücü tabanlı LSTM modeline beslenmiştir [45]. Böylece LSTM modeli ardışık olarak kelimeleri tahmin etmeye başlamıştır. Model ardışık olarak bir sonraki kelimeyi tahmin etmek için bir önceki kelimeye dayalı bir dil modeli kullanmıştır ve açıklamaya ait kelimeler sırayla tahmin edilmiştir. ***Categorical_crossentropy*** kayıp fonksiyonu kullanılarak gerçek değer ile tahmin değeri arasındaki farkın azaltılması hedeflenmiştir. Her üç model için de ilk etapta epoch değeri 20 olarak belirlenmiş ve kayıp değerleri gözlemlenmiştir. Flickr30k veri kümesi üzerinde yirminci epoch sonunda kayıp değerinin lineer bir şekilde azalmaya devam ettiği gözlemlenmiştir. Modelin veriyi ezberleme noktasına gelmediği tespit edildiğinden epoch değeri artırılarak 100 epoch ile eğitime devam edilmiştir. 88. epoch itibariyle kayıp değerinde duraklama başlamıştır. Eğitim süresinin uzun sürmesi sebebiyle yüzüncü epoch'da eğitim sonlandırılmış olup 88. epoch sonucu elde edilen modelin başarım değerleri 4. bölüm sonunda tablo halinde paylaşılmıştır.

4.3. Modelin Eğitildiği Test Ortamı ve Kullanılan Teknolojiler

Görüntü alt yazılama modeli geliştirilirken kullanılan teknolojiler aşağıda açıklanmıştır.

4.3.1. Programlama dili

Bu çalışmada makine öğrenmesi ve derin öğrenme uygulamalarında sıklıkla tercih edilen Python dili kullanılmıştır. Kolay kodlanabilir ve kolay anlaşılır bir dil olmasının yanı sıra yaygın kullanım sebebiyle bünyesinde birçok hazır kütüphaneyi de barındırmaktadır. Hazır kütüphanelerin çokluğu ve her geçen gün artıyor oluşu derin öğrenme modellerinin kolaylıkla tasarlanmasında büyük katkılar sağlamaktadır. Öğrenme modeli tasarlanırken derin öğrenme uygulamalarında sıklıkla kullanılan ve içerisinde birçok fonksiyon barındıran Tensorflow ve Keras kütüphaneleri tercih edilmiştir.

4.3.2. Geliştirme ortamı

Bu çalışmada Python temelli projelerin geliştirilmesinde sıklıkla kullanılan açık kaynak erişim olan Anaconda platformu tercih edilmiştir. Anaconda ortamı NumPy, Matplotlib, Pandas gibi sık kullanılan bazı kütüphaneleri içerisinde doğrudan bulundurmaktadır. Kodlama JetBrains tarafından geliştirilen ve Python dili ile yazılım geliştirmek için gelişmiş araçlar sunan PyCharm geliştirme ortamında gerçekleştirilmiştir.

4.3.3. Test ortamı

Hazırlanan modellerin eğitim ve testinin gerçekleştirildiği makine 64GB RAM ve NVIDIA Quadro M2000 ekran kartına sahiptir.

4.4. Modelin Değerlendirilmesi

Görüntü alt yazılama problemlerinin bilgisayarlı görü ve doğal dil işleme yöntemlerinin birlikte kullanılmasıyla çözüldüğü önceki bölümlerde aktarılmıştır. Literatürde görüntü alt yazılama alanında yapılan birçok çalışma bulunmaktadır ve bu çalışmalarda görüntü özniteliklerinin çıkarılması kullanılan dilden bağımsız olarak ilerleyen bir işlemdir. Ancak veri kümesinde bulunan diğer bir parametre olan metin verisi işlenirken, başka bir deyişle doğal dil işleme kısmı çözülürken kullanılan dilin özellikleri göz önünde bulundurulmalıdır. Literatürde yaygın olarak kullanılan alt yazılama modellerinin çoğu

İngilizce diline özel tasarlanmıştır ve doğrudan alınıp Türkçe için kullanılması mümkün değildir. Bu tez çalışmasında, Türkçe dilinde görüntü alt yazılama yapılması hedeflenmiş ve Türkçe diline özgü yapılar incelenmiştir. Alt yazılama yapılırken hecelerin kullanımının etkisini gözlemlemek üzere hecelere dayalı bir görüntü alt yazılama modeli önerilmiştir. Kıyas yapabilmek amacıyla kullanılan veri kümelerine ait metinler ön işlemden geçirilerek kelime, alt kelime veya köklerine ayrılmıştır. Metin verisi "word embedding" yöntemiyle kelime vektörlerine dönüştürüldükten sonra görüntü vektörü ile birlikte öğrenme modeline girdi olarak verilmiştir. Tablo 4.7.'deki deneysel sonuçlar incelendiğinde hecelere dayalı modelin tüm veri kümelerinde diğer iki modele göre başarılı olduğu görülmektedir. Tüm veri kümelerinde en başarısız sonuçların metnin boşluk karakteri yardımıyla sözcüklerine ayrıldığı *"sözcük modeli"*nde elde edildiği görülmüştür. Flickr30k veri kümesi özelinde başarımlara bakıldığında, test verisindeki metinlerin sözcüklere ayrıldığı ilk yöntemde BLEU-1 değeri 0.526482, kök veya alt kelime şeklinde temsil edildiği ikinci yöntemde 0.545882, hecelere dayalı yöntemde 0.584491 bulunmuştur. Bu durumda hecelere dayalı yöntemin ilk modelden %11, ikinci modelden %7 daha başarılı olduğu görülmektedir. Tablo 4.7. incelendiğinde veri kümesi boyutu arttıkça başarımların da paralel şekilde arttığı görülmüştür. Manuel veri toplama yöntemiyle oluşturulan veri kümesi gürültülü veriler içerdiğinden ve veriler arasında yapısal birliktelik bulunmadığından her üç modelde de diğer veri kümelerine göre en düşük başarımı göstermiştir. Oluşturulan metin verisi incelendiğinde Tablo 4.1.'den de görüleceği gibi bir görüntüye karşılık yazılan alt yazı sayısı bu çalışmada kullanılan diğer iki veri kümesinden düşük kalmıştır. Gönüllü kişiler tarafından manuel oluşturulan cümleler incelendiğinde bir kısmının doğal dil yapısından uzak olduğu, görüntü betimlenirken sadece nesne ve nesneye ait sıfatlar kullanıldığı görülmüştür. Görüntü verilerinin video materyallerinden kod yardımıyla çekilen karelerden (frame) oluştuğu önceki bölümlerde aktarılmıştır. Elde edilen görüntü verileri incelendiğinde akan videodan çekilen görüntülerin hazır veri kümelerindeki görüntüler kadar net olmadığı görülmektedir. Bu sebeple görüntüler içerisindeki nesneler, sahneler net bir şekilde ayırt edilememiştir ve elde edilen görüntü verileri beklenen kalitenin çok altında bir nesne tanıma ve öznitelik çıkarma olanağı sunmaktadır. Sayılan tüm bu sebepler diğer veri kümelerine kıyasla daha düşük başarım elde edilmesini açıklamaktadır. Ancak bu veri kümesi üzerinde her üç modelin başarımı karşılaştırıldığında hecelere dayalı yöntemin kelime modeline göre %5, alt kelime/kök modeline göre %3 daha başarılı olduğu görülmektedir. Bu sebeple manuel oluşturulan veri kümesinden elde edilen sonuçlar kıymetlidir ve hazır veri kümeleriyle elde edilen sonuçlarla paralel şekilde heceleme modelinin daha başarılı olduğunu göstermektedir.

 arazide beyaz bir köpek bir engelin üzerinden atlıyor	 bir adam bir su kütlesinin içinde yüzüyor
 şapkalı bir adam bir topun önünde duruyor	 bir köpek bir köpek tarafından kovalanıyor
 bir adam bir boğanın yanında duruyor	 arazide bir köpek bir köpek tarafından kovalanıyor

Şekil 4.4. Flickr8k veri kümesi üzerinde Heceleme Modeli ile oluşturulan alt yazılar

Şekil 4.4'te Flickr8k veri kümesinden örnek alınmış görüntüler ve önerilen model sonucu üretilen alt yazılar gösterilmektedir. İlk iki satırda başarılı sonuçlar, son satırda ise başarısız sonuçlar görülmektedir. Üst satırdaki görsellerde nesneler ve görüntü içerisindeki eylemlerin gerçek verilere yakın bir şekilde tahmin edildiği görülmektedir. Ortadaki satırda nesneler doğru tahmin edilirken aktivitelerin yanlış yorumlandığı, son satırda ise nesnelerin

karmaşık arka plandan doğru bir şekilde ayırt edilemediği görülmektedir. Genel olarak üretilen alt yazılar incelendiğinde önerilen hecelere dayalı modelde cümle yapısının doğru bir şekilde kurulabildiği gözlemlenmektedir.

Tablo 4.7. Deneysel Sonuçlar

Veri Kümesi	**Flickr8k (tamamı)**			**Flickr8k (Rastgele 2000)**			**Oluşturulan X Veri kümesi**			**Flickr30k (tamamı)**		
Model	***Sözcük***	***Hece***	***Alt kelime/kök***	***Sözcük***	***Hece***	***Alt kelime/kök***	***Sözcük***	***Hece***	***Alt kelime/kök***	***Sözcük***	***Hece***	***Alt kelime/kök***
Train	6475	6475	6475	1615	1615	1615	1615	1615	1615	25607	25607	25607
Test	1616	1616	1616	385	385	385	385	385	385	6401	6401	6401
BLEU-1	0.489017	**0.548882**	0.502412	0.452502	**0.531653**	0.482604	0.351822	**0.373656**	0.367809	**0.526482**	**0.584491**	**0.545882**
BLEU-2	0.283722	**0.364546**	0.290584	0.250868	**0.347743**	0.268843	0.187165	**0.205200**	0.175975	**0.341753**	**0.361488**	0.323413
BLEU-3	0.185128	**0.275884**	0.188704	0.170248	**0.257388**	0.181093	0.139792	**0.156844**	0.134659	0.214394	**0.235131**	0.193421
BLEU-4	0.045069	**0.158320**	0.084352	0.079783	**0.144433**	0.082684	0.060434	**0.082597**	0.068626	0.095217	**0.123773**	0.075244

5. SONUÇ VE ÖNERİLER

Bir önceki bölümde anlatılan hecelere dayalı görüntü alt yazılama modelinin eğitiminde Flickr8k, Flickr30k ve bu çalışma için manuel olarak hazırlanan 2000 adet görüntü ve onlara ait açıklamalardan oluşan veri kümesi kullanılmıştır. Manuel metin girilerek oluşturulan veri kümesinin başarımını Flickr8k ile kıyaslayabilmek için Flickr8k veri kümesinden 2000 adet rastgele görüntü alınarak yeni bir veri kümesi elde edilmiş ve önerilen modelin iki veri kümesi üzerindeki başarımı ölçülmüştür. Her bir veri kümesi %80 train, %20 test verisi olacak şekilde ayrıştırılmıştır. Seçilen bu veri kümeleri üzerinde sözcük, alt kelime/kök ve önerilen heceleme modelleri uygulanmış olup sonuçlar Tablo 4.7.'de gösterilmektedir.

Başarımları ölçmek için BLEU metriği n=1,2,3,4 olacak şekilde [47] kullanılmıştır. BLEU metriği makine çeviri alanında kullanılan en popüler metriklerden biridir ve tüm kelime n-gram eşleşmelerinin hesaplanması sonucu bir puan elde edilmektedir. İnsan çevirisi ile makine çevirisi arasında ne kadar kelimenin örtüştüğünün hesaplanması sonucu 0 ile 1 arasında bir değer döndürmektedir. Bu değer 1'e ne kadar yakınsa çeviri kalitesi o kadar yüksek demektir. Bu metrik görüntü alt yazılama problemlerinde de benzer şekilde ölçüm yapmakta hedef metin ile tahmin edilen metin arasındaki uzaklığı hesaplamaktadır.

Sonuçlar incelendiğinde heceleme tabanlı öğrenme modelinin, ölçüm yapılan her üç veri kümesinde de diğer iki yönteme göre başarılı olduğu görülmektedir. Manuel olarak oluşturulan veri kümesi başarım olarak Flickr8k'ın altında kalmıştır, ancak önerilen yöntemin bu veri kümesi üzerinde ortaya çıkardığı sonuçlar değerlidir. Tablo 4.7. incelendiğinde heceleme yönteminin diğer veri kümeleriyle elde edilen sonuçlarla paralel şekilde diğer kelime ve alt kelime/kök modellerine kıyasla başarılı olduğu görülmektedir. Edinilen bu bulgu heceleme modelinin, bu tez çalışması kapsamında kullanılan tüm veri kümelerinde diğer modellere kıyasla daha başarılı sonuçlar verdiğini kanıtlamaktadır. Hecelerin içerisinde önemli bilgiler kodladığı ve bu bilgilerin kaybedilmeden öğrenme modeline verilmesinin modelin başarımını artıracağı deney sonuçlarıyla net bir şekilde ortaya koyulmuştur.

Elde edilen tüm bu sonuçlar, önerilen heceleme modelinin hem Flickr8k hem Flickr30k veri kümesi üzerinde hem de daha önce üzerinde çalışılmamış bir veri kümesi üzerinde diğer iki yönteme göre daha başarılı olduğunu kanıtlamaktadır. Veri kümesi manuel oluşturulurken görüntülere ait alt yazılar serbest bir şekilde oluşturulduğundan yapısal olarak

bir birliktelik bulunmamakta ve çok fazla gürültü içermektedir. Diğer iki veri kümesine kıyasla elde edilen düşük başarımın bahsedilen bu sebeplerden kaynaklandığı değerlendirilmektedir. Gürültülü ve dolayısıyla değişkenlik gösteren verilerle çalışılırken derin öğrenme modellerinin tek başına yeterli başarım gösteremediği, tek model kullanmak yerine birden fazla modelin beraber kullanılmasıyla başarımın artırılabileceği bilinmektedir [48]. Bu sebeple gelecek çalışmalarda veri kümesi boyutunun artırılması ve veriler toplanırken belirli kurallar çerçevesinde manuel alt yazılama yapılmasının sağlanmasının yanı sıra topluluk öğrenmesi yöntemi uygulanarak birden fazla model ile daha yukarıda bir başarım elde edilmesi hedeflenmektedir.

Türkçe dili özelinde yapılan bu çalışma ile görüntü alt yazılama yapılırken dilin yapısı göz önünde bulundurularak ve dilin en temel bileşenleri belirlenerek dil modeli hazırlanmasının modelin başarımını artırdığı görülmüştür. Hecelerin kelimeler ve alt kelimelere kıyasla içerisinde fazla bilgi barındırdığı bu sebeple öğrenmenin içerisine dahil edilmeleri halinde modellerin daha doğru metinler üreteceği deneysel sonuçlarla gösterilmiştir. Türkçe dışındaki diğer diller için de dilin kendi dinamikleri göz önüne alınarak ve temel bileşenlerine inilerek öğrenme modeli hazırlanmasının başarımı artıran bir faktör olacağı değerlendirilmektedir. Bu anlamda bu tez çalışmasının İngilizce dışındaki diğer diller üzerindeki görüntü alt yazılama çalışmalarına da katkı sağlayacağı düşünülmektedir.

KAYNAKLAR

[1] D. Colarelli and D. Grunwald, "Massive Arrays of Idle Disks For Storage Archives," in *ACM/IEEE SC 2002 Conference (SC'02)*, 2002, vol. 2002-Novem, pp. 47–47. doi: 10.1109/SC.2002.10058.

[2] R. Bernardi *et al.*, "Automatic description generation from images: A survey of models, datasets, and evaluation measures," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 0, pp. 4970–4974, 2017.

[3] A. Farhadi *et al.*, "Every picture tells a story: Generating sentences from images," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6314 LNCS, no. PART 4, pp. 15–29, 2010, doi: 10.1007/978-3-642-15561-1_2.

[4] A. Gupta and L. S. Davis, "Objects in action: An approach for combining action understanding and object perception," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. 2, 2007, doi: 10.1109/CVPR.2007.383331.

[5] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: Using spatial and functional compatibility for recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1775–1789, 2009, doi: 10.1109/TPAMI.2009.83.

[6] X. Liu, Q. Xu, and N. Wang, "A survey on deep neural network-based image captioning," *Vis. Comput.*, vol. 35, no. 3, pp. 445–470, 2019, doi: 10.1007/s00371-018-1566-y.

[7] H. Sharma, M. Agrahari, S. K. Singh, M. Firoj, and R. K. Mishra, "Image Captioning: A Comprehensive Survey," *2020 Int. Conf. Power Electron. IoT Appl. Renew. Energy its Control. PARC 2020*, pp. 325–328, 2020, doi: 10.1109/PARC49193.2020.236619.

[8] M. H. Guo *et al.*, "Attention mechanisms in computer vision: A survey," *Comput. Vis. Media*, vol. 8, no. 3, pp. 331–368, 2022, doi: 10.1007/s41095-022-0271-y.

[9] Z. Zohourianshahzadi and J. K. Kalita, “Neural attention for image captioning: review of outstanding methods,” *Artif. Intell. Rev.*, vol. 55, no. 5, pp. 3833–3862, 2022, doi: 10.1007/s10462-021-10092-2.

[10] M. E. Unal *et al.*, “TasvirEt: A benchmark dataset for automatic Turkish description generation from images,” in *2016 24th Signal Processing and Communication Application Conference (SIU)*, May 2016, pp. 1977–1980. doi: 10.1109/SIU.2016.7496155.

[11] N. Samet, S. Hicsonmez, P. Duygulu, and E. Akbas, “Görüntü Altyazilama için Otomatik Tercümeyle Eğitim Kümesi Oluşturulabilir mi?,” *2017 25th Signal Process. Commun. Appl. Conf. SIU 2017*, pp. 6–9, 2017, doi: 10.1109/SIU.2017.7960638.

[12] M. Kuyu, A. Erdem, and E. Erdem, “Image captioning in Turkish with subword units,” in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, May 2018, pp. 1–4. doi: 10.1109/SIU.2018.8404431.

[13] P. Gage, “A New Algorithm for Data Compression,” *C Users J.*, pp. 1–14, 1994, doi: 10.5555/177910.177914.

[14] M. Stefanini, M. Cornia, L. Baraldi, S. Cascianelli, G. Fiameni, and R. Cucchiara, “From Show to Tell: A Survey on Deep Learning-Based Image Captioning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 539–559, 2023, doi: 10.1109/TPAMI.2022.3148210.

[15] B. Citamak *et al.*, “MSVD-Turkish: a comprehensive multimodal video dataset for integrated vision and language research in Turkish,” *Mach. Transl.*, vol. 35, no. 2, pp. 265–288, 2021, doi: 10.1007/s10590-021-09276-y.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” pp. 1–9.

[17] J. Obdržálek, “The essential Turing, edited by B. Jack Copeland, Oxford University Press, 2004, vii + 613 pp.,” *Bull. Symb. Log.*, vol. 11, no. 4, pp. 541–542, Dec. 2005, doi: 10.1017/S1079898600003012.

[18] Ö. İnika and E. Ülker, “Derin Öğrenme ve Görüntü Analizinde Kullanılan Derin Öğrenme Modelleri” 2017.

[19] D. H. HUBEL and T. N. WIESEL, “Receptive Fields of Single Neurones in the Cat’s Striate Cortex By D . H . HUBEL * and T . N . WIESEL * cortex provides an opportunity to observe and compare single unit re- sponses at several distinct levels . Patterns of light stimuli most effective i,” pp. 574–591, 1959.

[20] S. C. Zhu, L. Angeles, Y. Wu, and L. Angeles, “Learning Vector Representation of Content and Matrix Representation of Change : Towards a Representational Model of V1 Learning Vector Representation of Content and Matrix Representation of Change : Towards a Representational Model of V1,” no. February, 2019, doi: 10.13140/RG.2.2.36719.28325.

[21] E. C. Hildreth and D. MARR, “Theory of Edge Detection Theory of Edge Detection,” no. January, 2016, doi: 10.1098/rspb.1980.0020.

[22] R. A. Kirsch, “Computer Interpretation of English Text and Picture Patterns with tshe qsi-tences . Textu isalsmatte condsta inthere The ~,” pp. 363–376, 1964.

[23] Y. LeCun *et al.*, “Backpropagation applied to handwritten zip code recognition.pdf.”

[24] M. J. Jones and P. Viola, “Robust Real-time Object Detection Paul Viola February 2001,” no. February, 2001.

[25] M. Everingham, “The PASCAL Visual Object Classes Challenge 2006 Development Kit,” pp. 1–11, 2006.

[26] R. M. Oruganti, “Image Description using Deep Neural Networks,” 2016.

[27] F. Rosenblattt, “Perceptron Simulation Experiments,” *Routledge Libr. Ed. Econom.*, pp. 415–451, 1960, doi: 10.4324/9781315717791-15.

[28] D. R. S. Saputro, I. M. Putri, Sutanto, N. H. Noor, and P. Widyaningsih, “Generalized space time autoregressive (gstar)-artificial neural network (ann) model with multilayer feedforward networks architecture,” *IOP Conf. Ser. Earth Environ. Sci.*, vol. 243, no. 1, p. 012039, Apr. 2019, doi: 10.1088/1755-1315/243/1/012039.

[29] L. P. Bordignon and A. Von Wangenheim, “Benchmarking Deep Learning Models

on Jetson TX2," *Tech. Rep. INCoD/LAPIX ...*, 2011, [Online]. Available: www.incod.ufsc.br

[30] V. Aggarwal and G. Kaur, "A review:deep learning technique for image classification," *Accent. Trans. Image Process. Comput. Vis.*, vol. 4, no. 11, pp. 21–25, 2018, doi: 10.19101/tipcv.2018.411003.

[31] F. Zhang, Z. Guo, X. Sun, and J. Xi, "Short-term wind power prediction based on EMD-LSTM combined model," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 514, no. 4, p. 042003, May 2020, doi: 10.1088/1755-1315/514/4/042003.

[32] H. Lee, J. Kang, S. Kim, Y. Im, S. Yoo, and D. Lee, "Long-Term Evaluation and Calibration of Low-Cost Particulate Matter (PM) Sensor," *Sensors*, vol. 20, no. 13, p. 3617, Jun. 2020, doi: 10.3390/s20133617.

[33] Y. Li, "Key Technologies of Financial Digital Industry Innovation and Green Development Driven by Information Technology," *Int. J. Comput. Intell. Syst.*, vol. 16, no. 1, p. 78, May 2023, doi: 10.1007/s44196-023-00262-1.

[34] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout : A Simple Way to Prevent Neural Networks from Overfitting," vol. 15, pp. 1929–1958, 2014.

[35] A. Ahmet KARABAŞ, "Fen Bilimleri Enstitüsü Türkçe Mikrobloglarda İroni Tespiti" 2019.

[36] T. Mikolov, G. Corrado, K. Chen, and J. Dean, "Vector Space," pp. 1–12.

[37] A. Karpathy, "Deep Visual-Semantic Alignments for Generating Image Descriptions".

[38] M. Hodosh, "Framing Image Description as a Ranking Task : Data , Models and Evaluation Metrics," vol. 47, pp. 853–899, 2013.

[39] C. Rashtchian, P. Young, M. Hodosh, and J. Hockenmaier, "Collecting image annotations using Amazon's Mechanical Turk," *Work. Creat. Speech Lang. Data with Amaz. Mech. Turk, Mturk 2010 2010 Annu. Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. NAACL-HLT 2010 - Proc.*, no. June, pp. 139–147, 2010.

[40] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, “From image descriptions to visual denotations : New similarity metrics for semantic inference over event descriptions,” vol. 2, pp. 67–78, 2014.

[41] P. F. Güneş, “Türkçe Ders Kitaplarında Kullanılan Hecelerin İncelenmesi,” 2018.

[42] F. Almeida and G. Xex, “Word Embeddings: A Survey,” no. 1991, 2015.

[43] S. Niu, Y. Liu, J. Wang, H. Song, and S. Member, “A Decade Survey of Transfer Learning (2010 – 2020),” vol. 1, no. 2, pp. 151–166, 2021, doi: 10.1109/TAI.2021.3054609.

[44] F. Zhuang *et al.*, “A Comprehensive Survey on Transfer Learning,” *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.

[45] G. Van Houdt, C. Mosquera, and G. Nápoles, “A review on the long short-term memory model,” *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5929–5955, Dec. 2020, doi: 10.1007/s10462-020-09838-1.

[46] K. Smagulova and A. P. James, “A survey on LSTM memristive neural network architectures and applications,” *Eur. Phys. J. Spec. Top.*, vol. 228, no. 10, pp. 2313–2324, Oct. 2019, doi: 10.1140/epjst/e2019-900046-x.

[47] K. Papineni, S. Roukos, T. Ward, and W. Zhu, “BLEU : a Method for Automatic Evaluation of Machine Translation,” no. July, pp. 311–318, 2002.

[48] I. Lee, D. Kim, S. Kang, and S. Lee, “Ensemble Deep Learning for Skeleton-Based Action Recognition Using Temporal Sliding LSTM Networks,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, vol. 2017-Octob, pp. 1012–1020. doi: 10.1109/ICCV.2017.115.

[49] İnternet: researchgate.net/post/What-is-the-interplay-between-neural-networks-and-meta-algorithms/. Son Erişim Tarihi: 02.07.2023

[50] İnternet: owardsdatascience.com/covolutional-neural-network-cb0883dd6529/. Son Erişim Tarihi: 28.06.2023

[51] İnternet: researchgate.net/figure/MaxPooling-operation-on-a-4-4-image-with-2-2-window-size-where-stride2-and_fig2_344197871/. Son Erişim Tarihi: 11.07.2023

[52] İnternet: freetimelearning.com/software-interview-questions-and-answers.php?Explain-transfer-learning-in-the-context-of-deep-learning.&id=4184/. Son Erişim Tarihi: 14.07.2023

[53] İnternet: opringle.github.io/2017/11/14/CNN_text_classification.html/. Son Erişim Tarihi: 10.07.2023

[54] İnternet: tr.wikipedia.org/wiki/Hece Son Erişim Tarihi: 04.05.2023

[55] İnternet: tdk.gov.tr/icerik/yazim-kurallari/hece-yapisi-ve-satir-sonunda-kelimelerin-bolunmesi/ Son Erişim Tarihi: 04.05.2023