T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İÇERİK TABANLI WEB SAYFASI KATEGORİZASYONU

EBUBEKİR BÜBER

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
PROF. DR. BANU DİRİ

İSTANBUL, 2018

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

İÇERİK TABANLI WEB SAYFASI KATEGORİZASYONU

Ebubekir BÜBER tarafından hazırlanan tez çalışması 05.12.2018 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Banu DİRİ

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Prof. Dr. Banu DİRİ

Yıldız Teknik Üniversitesi _____

Doç. Dr. Fatih AMASYALI

Yıldız Teknik Üniversitesi _____

Doç. Dr. Özgür Koray ŞAHİNGÖZ

İstanbul Kültür Üniversitesi _____

ÖNSÖZ

Tez çalışmam boyunca benden bilgisini, desteğini ve sabrını esirgemeyen hocam Prof. Dr. Banu Diri'ye, manevi desteğini hiç eksik etmeyen Şeyma Cengiz'e, tez çalışmasına olan desteklerinden dolayı Roksit Arge Ltd. Şti.'e Teşekkürü bir borç bilirim.

Aralık, 2018

Ebubekir BÜBER

İÇİNDEKİLER

Sayfa

KISALTMA LİSTESİ

BRNN	Bidirectional Recurrent Neural Network
CSS	Cascading Style Sheets
CNN	Convolutional Neural Network
ÇYYSA	Çift Yönlü Yinelemeli Sinir Ağı
DDİ	Doğal Dil İşleme
ESA	Evrişimli Sinir Ağları
EYiSA	Evrişimli Yinelemeli Sinir Ağı
FTKG	Frekans Tabanlı Kelime Gösterim
FQDN	Fully Qualified Domain Name
GİA	Gradyan İzdüşüm Algoritması
HTML	Hypertext Markup Language
LSTM	Long Short Term Memory
NN	Neural Network
RCNN	Recurrent Convolutional Neural Networks
RNN	Recurrent Neural Network
RSA	Rekürsif Sinir Ağları
TTKG	Tahminleme Tabanlı Kelime Gösterim
URL	Uniform Resource Locator
YiSA	Yinelemeli Sinir Ağı

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖZET

İÇERİK TABANLI WEB SAYFASI KATEGORİZASYONU

Ebubekir BÜBER

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Prof.Dr. Banu DİRİ

Web Sayfalarının Sınıflandırılması, her geçen gün daha da önem kazanan bir makine öğrenmesi problemidir. Web sayfalarının kategorize edilmesi, verimli İnternet kullanımı, spam filtreleme ve daha birçok uygulama alanı için faydalı bilgiler sağlamaktadır. Milyonlarca web sitesi arasından kullanıcının aradığı konuyla ilgili sonuçların hızlı bir şekilde bulunması, arama motorları için çözülmesi gereken bir problemdir. Web sayfası sınıflandırma, zararlı içeriğe sahip web sayfalarının kullanıcı tarafından görüntülenmeden önce engellenmesi ile siber güvenlik uygulamaları tarafından da kullanılabilmektedir.

Web sayfası sınıflandırması, birçok farklı uygulama alanı için temel oluşturabilecek faydalı bilgiler sağlayan bir Bilgi Çıkarımı (Information Extraction) uygulamasıdır. Bir diğer uygulama alanına ise ağda anomali tespiti için kullanıcının internet kullanım profilinin oluşturulması örnek olarak verilebilir.

Bu çalışmada, web sayfalarının sınıflandırılmasına yönelik bir sistem geliştirilmiştir. Geliştirilen sistemde derin öğrenme tabanlı yaklaşımlar test edilmiş ve kullanılmıştır. Web sayfalarının sınıflandırılabilmesi için bir web sayfasının içeriğinde yer alan meta etiketler adı verilen başlık (title), açıklama (description) ve anahtar kelimeler (keywords) gibi metinsel bilgiler kullanılmıştır.

Yapılan çalışmanın testleri sırasında Yinelemeli Sinir Ağı (YiSA, Recurrent Neural Networks) tabanlı derin öğrenme mimarisi kullanılmıştır. Bu derin öğrenme mimarisi üzerinde bazı hiperparametre ayarlamaları gerçekleştirilerek performans analizi de yapılmıştır. Ayrıca, geliştirilen sistemde Öğrenme Transferi denenmiştir. Öğrenme Transferi, bir problemi çözmek için önceden eğitilmiş parametreler kullanılarak bir makine öğrenmesi modeli oluşturma yaklaşımına verilen isimdir.

Elde edilen sonuçlara göre, Web sayfası sınıflandırma sisteminin başarı oranı yaklaşık %85 olarak elde edilmiştir. Gerçekleştirilen testler, CPU ve GPU üzerinde çalıştırılmış olup, bu iki farklı donanım üzerinde elde edilen çalışma sürelerine ilişkin performans karşılaştırması da ayrıca yapılmıştır.

Anahtar Kelimeler: web sayfası sınıflandırma, sınıflandırma, derin öğrenme, Yinelemeli Sinir Ağları, öğrenme transferi.

ABSTRACT

CONTENT BASED WEB PAGE CATEGORIZATION

Ebubekir BÜBER

Department ComputerEngineering

MSc. Thesis

Adviser: Prof. Dr. BanuDİRİ

Classification of Web Pages is a machine learning problem which gets more and more important every day. Categorizing web pages provides useful information for efficient internet use, spam filtering and many other application areas. Finding results quickly from the millions of websites users are looking for is a problem that must be solved for search engines. Web page classification can also be used by cyber security applications by blocking web pages with malicious content before they are displayed by the user.

Web page classification is an Information Retrieval application that provides useful information that can be a basis for many different application domains. Another example of application is the creation of an internet usage profile of a user for network anomaly detection.

In this study, a system for classifying web pages was developed. Deep learning-based approaches have been tested and used in the developed system. Textual information in the content of the web page is used to classify web pages. For the classification mechanism, the meta tags contained in the web page are used. The meta tags used for classification are title, description, keywords.

During the tests, a deep learning architecture based on Recursive Neural Networks (RNN) was used. Performance analysis has been performed by performing some hyperparameter tuning on this deep learning architecture. In addition, Transfer

Learning has been tested in the developed system. Transfer Learning is the name given to the approach of building a machine learning model using pre-trained parameters to solve a problem.

According to the results obtained, the success rate of the web page classification system is about 85%. The tests were run on the CPU and GPU, and the performance comparison of the run times obtained on this two different hardware was made.

Keywords: web page classification, classification, categorization, deep learning, RNN, transfer learning.

BÖLÜM 1

GİRİŞ

Web Sayfaların Sınıflandırılması, her geçen gün daha da önem kazanan bir makine öğrenmesi problemidir. İnternetin kullanılmaya başladığı 90'lı yıllardan bu yana İnternet kullanıcı sayısı ve kullanıcılara hizmet veren web sayfası sayısı günümüze kadar büyük bir hızla artmış ve artmaya devam etmektedir. Web sayfalarının kategorize edilmesi, verimli İnternet kullanımı, spam filtreleme ve daha birçok uygulama alanı için faydalı bilgiler sağlamaktadır. Milyonlarca web sitesi arasından kullanıcının aradığı konuyla ilgili sonuçların hızlı bir şekilde bulunması, arama motorları için çözülmesi gereken ciddi bir problemdir. Bu nedenle kullanıcı sorgularına daha sağlıklı sonuçlar döndürülebilmesi için bazı arama motorları web sayfaları üzerinde konu bazlı sınıflandırma yapmaya ihtiyaç duymuştur. Bunların yanı sıra kurumlar ya da bireysel kullanımlar için internet kullanım politikaları belirlenebilmesi için web sayfaların sınıflandırılması gerekmektedir. Web sayfası sınıflandırma, zararlı içeriğe sahip web sayfalarının kullanıcı tarafından görüntülenmeden önce engellenmesi ile siber güvenlik uygulamaları tarafından da kullanılabilmektedir. Web sayfalarının otomatize bir şekilde sınıflandırılması, çok büyük miktardaki verinin anlamlandırılması ve yönetilebilmesini gerekli kılmaktadır. Web sayfalarınınınsan eli ile kategorize edilmesi oldukça maliyetli bir işlemdir. Bu nedenle elle sınıflandırılmış olan web sayfaları günümüzdeki web sayfalarının çok küçük bir kısmını oluşturmaktadır.

Web sayfası sınıflandırılması, birçok farklı uygulama alanı için temel oluşturabilecek faydalı bilgiler sağlayan bir Bilgi Çıkarımı (Information Retrieval) uygulamasıdır. Örnek bir diğer uygulama alanı ise, ağda anomali tespiti için bir kullanıcının internet kullanım profilinin oluşturulması örnek olarak verilebilir.

1.1 Literatür Özeti

Bilgisayar ve ağ teknolojilerindeki hızlı gelişim, Web'in popülerliğinin çok kısa sürelerde artmasını sağlamıştır. Web'in çok büyük hızlarla büyümesi ve her geçen gün çok sayıda Web sitesinin yayına alınmasıyla arama motorlarının işini zorlaştırmaktadır. Bu nedenle bazı arama motorları arama yaparken sadece konu ile ilişkili web sayfaları arasında arama yaparak sorgu sonuçlarının başarısını optimize etmeye çalışmıştır. Literatürde bu amaca yönelik geliştirilen çalışmalar [1-3] bulunmaktadır. Otomatik Web Sayfası Sınıflandırma, bir Eğitimli Öğrenme (Supervised Learning) ile çözülebilen bir problemdir. Literatürde web sayfası sınıflandırmaya yönelik makine öğrenmesi algoritmaları kullanılarak geliştirilen çalışmalar bulunmaktadır [4-7]. Literatürde yer alan çalışmalarda sıklıkla kullanılan algoritmalardan bazıları şunlardır; Karar Ağaçları, Yapay Sinir Ağı [6], Destek Vektör Makineleri [5], [8], K En Yakın Komşu [9], Bayesian Algoritması [10].

Bu algoritmaların yanı sıra Genetik Algoritma Tabanlı [11-15], Karınca Koloni algoritması tabanlı [16] web sayfası sınıflandırmaya yönelik yapılan çalışmalar da litaretürde mevcuttur.

Web sayfalarınının Eğitimsiz Yöntemler ile gruplandırarak kategorize edildiği çalışmalar da mevcuttur [17]. Bu yaklaşımlarda web sayfaları benzerliklerine göre gruplara ayrılır. Benzerliklerin belirlenebilmesi için mesafe hesabı yapılır. [17], arama motoru sonuçlarının kümelenmesi ile arama sorgularına ilişkin bir optimizasyon yapmıştır.

Web sayfası sınıflandırma probleminin geleneksel metin sınıflandırma problemlerinden yapısal bazı farklılıkları bulunsa da sınıflandırma işlemi kullanılan algoritmalar ve yaklaşımlar birbirlerine benzemektedir. Bu çalışma kapsamında web sayfalarının sınıflandırılmasına yönelik bir sistem geliştirilmeden önce literatürde bulunan metin sınıflandırmaya yönelik çalışmalar da incelenmiştir.

Metin Sınıflandırma problemlerinden en önemli sorunlardan bir tanesi özellik gösterimidir (feature representation). Geleneksel yöntemlerin kullanıldığı metin sınıflandırma yaklaşımlarında genel özellik gösterimi yaklaşım Kelime Çantası (Bag of Word)'dır. Kelime çantası yaklaşımlarında kelimelere ilişkin özellikler; unigram, bigram, n-gram gibi gösterimlerkullanılmaktadır.

Geleneksel yaklaşımlarda özellik gösterimlerde ortaya çıkan en önemli sorunlardan birisi ortaya çıkan özellik sayısının çok fazla olmasıdır. Özellik sayısının çok fazla olması, sistemlerin eğitilmesi ve test edilmesi aşamalarında çok fazla işlem yüküne neden olmaktadır. Bu nedenle özelliklerin tamamının kullanılmasının yerine belirleyici bazı özelliklerin ayıklanması ve bu özelliklerin kullanılması oldukça sık kullanılan bir yaklaşımdır. Ayırt edici özelliklerin seçilmesi için geliştirilmiş birçok yöntem bulunmaktadır. Bu yöntemlerden bazıları MI (Mutual Information) [18], pLSA (Probabilistic Latent Semantic Analysis) [19], LDA (Latent Dirichlet Allocation) [20]'dir. Özellik Azaltımı ile ilgili detaylı bilgiler Bölüm 1.5'te verilmiştir.

Metin sınıflandırma problemlerinin çözülmesi için geleneksel yaklaşımlar kullanlanılarak geliştirilen yaklaşımların bir diğer önemli dezavantajı kelimelere ilişkin içeriksel/anlamsal bilgilerin genellikle kullanılamamasıdır. Geleneksel özellik gösterimi yaklaşımlarında (örn. n-gram gösterimi) kelimelere ilişkin anlamsal bilgiler tutulamamaktadır. Bunların yanısıra deyimler gibi kelimelerin bir araya gelerek oluşturduğu anlamsal kelime gruplarının da temsil edilemesi geleneksel özellik gösterimi yaklaşımlarda oldukça zordur. Bazı kelimeler biraraya gelerek gerçek anlamlarından çok daha farklı anlama gelen kelime grupları oluşturabilmektedirler. Bu kelime gruplarının işlenmesi esnasında kelimelerin ayrı ayrı değerlendirilmesi sistemi yanıltıcı etki oluşturabilmektedir. Bu sorun n-gram yaklaşımlarında n sayısının artırılması ile kısmen çözülebilse de ortaya çıkan yeni anlamın temsil edilmesi geleneksel yaklaşımlarda büyük bir sorun teşkil etmektedir. Bunların yanı sıra n sayısının artırılması veri seyrekliği (data sparsity) sorununa da neden olmaktadır.

Birçok makine öğrenmesi probleminde olduğu gibi derin öğrenme yaklaşımları metin sınıflandırma alanında da son yıllarda oldukça yaygın bir şekilde kullanılmaya başlanmıştır. Derin öğrenme yaklaşımları kullanılarak geliştirilen metin sınıflandırma yaklaşımları literatürde oldukça önemli yer tutmaktadır.

1.1.1 Literatürde Yer Alan Veri Setleri

Web sayfası sınıflandırması sisteminin başarısının test edilebilmesi için doğruluğundan emin olunan etkiketli bir veri setine ihtiyaç duyulmaktadır. Literatürde bu amaç için Web Dizinleri (Web Directory) kullanılmaktadır. Web dizinleri, web sitelerinin belirli

kategorilere göre bölündüğü ve genellikle elle sınıflandırma yapılan sistemlerdir. Web dizinleri sayesinde kullanıcı, sadece ilgilendiği konuya ait web sayfaları içerisinde web sorgulamaları yapabilir. Örnek bir web dizini Şekil 1.1'de verilmiştir.

Şekil 1.1. Örnek Web Dizini

Konu ile ilgili yapılan çalışmalarda en çok kullanılan web dizinlerinden bazıları şunlardır;

- Yahoo Directory
- DMON (from directory.mozilla.org)
- Best of the web
- World Wide Web Virtual Library
- Martindale's Reference Desk
- Jasmine Directory
- Hotfrog
- Incrawler
- Family Friendly Sides
- Enquira

Bunların yanı sıra piyasada çeşitli amaçlar için geliştirilmiş olan bazı ürünler de güçlü bir web sayfası sınıflandırma veri tabanına sahiptir.

HTML'de yazılan bir web sayfası içerisinde bulunan elementler HTML etiketleri ile organize edilir. HTML etiketleri ile web sayfasına metin, resim, ses dosyası, video konulabilir. Ayrıca, bu etiketler aracılığı ile başka web sayfalarına köprü bağlantıları (hyperlinkler) tanımlanabilir. Köprü bağlantı kurulan komşu web sayfaları, analiz edilen web sayfası hakkında oldukça faydalı bilgiler vermektedir. Bu web sayfalarına ilişkin bilgi, analiz edilen web sayfası içerisinde yer almamasına rağmen, komşu web sayfası analizi web sayfa sınıflandırma işlemi için önemli bir işlemdir. Bu nedenle web sayfası sınıflandırması için kullanılan özellikler ikiye ayrılmaktadır. Bunlar;

- Analiz edilen sayfadan elde edilen bilgiler
- Komşu web sayfalarından elde edilen bilgiler.

1.1.2 Analiz Edilen Web Sayfasından Elde Edilen Bilgiler

Analiz edilen web sayfasından elde edilen bilgiler iki çeşittir. Bunlar; metinsel içerik ve görsel içeriktir.

Metinsel İçerik, bir web sayfasının sınıflandırılmak üzere analiz edilmesi sırasında kullanılan ilk bilgilerdir. Web sayfalarının yapısı gereği barındırdığı kontrol edilemez gürültü ve yapısal değişkenlik gibi etkenler Kelime Torbası modeliyle yüksek performanslar elde edilmesini engellemektedir. Bu nedenle araştırmacılar metinsel içeriklerin kullanılmasında kelime tabanlı n-gram yöntemini tercih etmektedir. Bu yaklaşımda, her doküman bir özellik vektörü ile ifade edilir. Kullanılan özellikler tek bir kelime değil, birbiri ardına gelmiş n sayıda kelime grubundan oluşur. Bu sayede deyimler ve tamlamalar gibi kelimelerin bir arada kullanılarak anlam kazandığı ifadeler temsil edilebilmekte ve sınıflandırma mekanizmasında kullanılabilmektedir. Örneğin, bir dokümanın "New York" kelime grubunu içerdiğini düşünülürse, Kelime Torbası yönteminde olduğu gibi bu kelimeler birbirinden bağımsız özellikler olarak değerlendirilirse "New" ve "York" olmak üzere iki adet farklı özelliğimiz bulunurdu. Ancak, bu kelime bir arada bulunduğunda daha farklı bir anlam taşımaktadır. N-gram yaklaşımında n=2 olduğu durumda "New York" şeklinde iki kelimenin bir arada yer

aldığı bir özellik kullanılır. Bu yöntemin bir dezavantajı, bir doküman grubu için Kelime Torbası yaklaşımına göre toplamda daha fazla sayıda özellik çıkarılmasıdır. Bu nedenle n-gram yaklaşımını kullanan çalışmalar özellik seçimi kullanarak dokümanları iyi şekilde ifade edecek daha az sayıda özelliği seçerek bu özellikler ile çalışmaktadır.

Metinsel içeriğin yanı sıra bir web sayfasının sınıflandırılması için kullanılabilecek en belirgin ikinci veri kaynağı HTML etiketleridir. Yapılan akademik çalışmalarda, metinsel içerikle birlikte HTML etiketlerininkullanılmasının başarıyı artırdığı gösterilmiştir. HTML etiketleri, web sayfasının görsel olarak tarayıcı ekranında görünmesini sağlayan yapılardır. Farklı HTML etiketlerikullanılarak aynı görsel çıktı elde edilebilmektedir. Bu nedenle bazı çalışmalarda HTML etiketlerigruplara ayrılmış ve bu şekilde özellik çıkarılmış ve sınıflandırmada kullanılmıştır.

Literatürde yapılan bazı çalışmalar, web sayfasından tüm metini işleyerek sınıflandırmak yerine web sayfasında bulunan metinsel bilgilerin özetleri üzerinde sınıflandırma yapmışlardır [21], [22]. Bu yaklaşımla saf metin tabanlı sınıflandırıcılara kıyasla %8,8 performans artışı sağlanabilmiştir.

Sınıflandırılmak istenilen bir web sayfası içeriği tasarlanan sınıflandırıcılar tarafından işlenebilir formatta olmayabilir, web sayfasındaki içerik çok az olabilir ya da web sayfası analiz edilen anda ulaşılamaz durumda olabilir. Bu durumlarda web sayfasının başarılı bir şekilde içerik tabanlı sınıflandırılması mümkün değildir. Bu gibi durumlara karşılık, Kan ve Thi [23] web sayfasını sadece URL'e göre analiz ederek sınıflandırmaya çalışmıştır. Sadece URL analizi ile yapılan sınıflandırma yüksek başarılara sahip olmayabilir ancak, web sayfasında birtakım bilgiler elde edilmesi açısından faydalı görülmektedir.

Analiz edilen web sayfasından elde edilen bilgilerden bir diğeri görsel içeriktir. Internette yer alan web sayfaları genelde iki şekilde temsil edilir. Birincisi HTML ile yazılmış olan metinsel temsil, ikincisi ise HTML içeriğin tarayıcı tarafından kullanıcılar için görselleştirildiği temsildir. Web sayfalarının sınıflandırılmasına yönelik literatürde yer alan çalışmaların önemli bir kısmı sadece metinsel içerik ve HTML etiketlerininişlenmesi sonucu çıkarılan özelliklerle çalışmaktadır [24]. Ancak, görsel

içeriğin analiz edilmesi ile de web sayfalarının sınıflandırılasında faydalı bilgiler elde edilebilmektedir [24].

Kovacevic [25]'in çalışmasında web sayfaları, yapılarında bulunan HTML etiketlere gore "Visual Adjacency MultiGraph" olarak temsil edilmiştir. Graf içerisindeki her düğüm bir HTML nesnesini ifade etmektedir. Graf içerisinde yer alan kenarlar ise HTML nesneleri arasındaki uzamsal ilişkileri temsil etmektedir. Oluşturulan grafa uygulanan sezgisel kurallar ile graf içerisinde bazı mantıksal bölgeler ayırt edilebilmiştir. Yapılan bu çalışmada Kelime Torbası yaklaşımına göre ciddi bir performans artışı elde edilmiştir.

Yapılan bir diğer çalışmada [26] ise web sayfasında bulunan resimler, histogramlarına göre analiz edilerek yapay resimler ile doğal fotoğraflar birbirinden ayırt edilmiştir. Web sayfasında bulunan resimlerin analiz edilmesi histogramlardan faydalanılarak çıkarılan özellikler, web sayfası sınıflandırmada performansı artırıcı etki oluşturduğu görüşmüştür.

Web sayfasında yer alan özellikler web sayfalarının sınıflandırılması için faydalı bilgiler sunmaktadır. Ancak, web sayfası içeriği sayfa yöneticisinin bakış açısı ile ilgili bir konudur. Bazı durumlarda web sayfalarının sınıflandırılması için web sayfası içerisinde doğrudan yer almayan özelliklerin kullanılması da sınıflandırma mekanizmalarına katkı sağlamaktadır.

1.1.3 Komşu Web Sayfalarından Elde Edilen Bilgiler

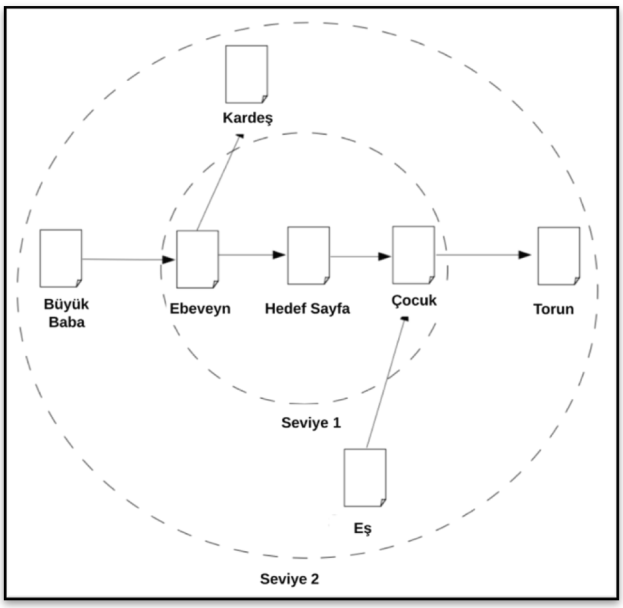
Bazı durumlarda web sayfalarında bulunan içerikler eksik, iyi yapılandırılmamış ya da büyük boyutlu flash nesneleri kullanılmış olabilir. Bu gibi durumlarda sadece sayfa içerisinde bulunan verilerin analiz edilmesi ile doğru sınıflandırma yapılması zordur. Bu nedenle komşu sayfalardan elde edilen özelliklerin kullanılması doğru sınıflandırmada yardımcı olmaktadır. Yapılan bazı çalışmalar da [10], [27], komşu sayfaların analiz edilmesi ile web sayfalarının sınıflandırılması üzerine odaklanılmıştır.

Sayfalar arasında çok çeşitli şekillerde bağlantılar kurulabilir. En sık kullanılan yöntemlerin başında köprü bağlantıları gelmektedir. Komşu web sayfalarından elde edilen özelliklerin kullanıldığı çalışmaların çoğu köprü bağlantılarına odaklanmıştır.

Komşu web sayfalarından elde edilen özellikler bazı varsayımlardan yola çıkılarak oluşturulmaktadır. Örneğin, Pa ve Pb aynı kategoriye sahip iki web sayfası ise bu web sayfalarının komşularından oluşturulan web graflar birbirlerine benzer karakteristik özellikler göstermektedirler. Bu varsayım, komşu web sayfalarının aynı kategoride olmasını gerektirmez. Bu varsayım *zayıf varsayımlar* sınıfına girmektedir. Bu tarz varsayımlar Ana Konu Sınıflandırması ve Fonksiyonel Sınıflandırmada iyi sonuçlar vermektedir [24].

Ana Konu Sınıflandırmasında daha güçlü bazı varsayımlar da kullanılmaktadır. "Bir web sayfası komşu web sayfalarının ağırlıklı olduğu kategoridendir" varsayımı buna örnektir. Örneğin, komşu web sayfalarının "Spor" ağırlıklı olarak kategorisinde yer almaları analiz edilen web sayfanın kategorisinin "Spor" olma olasılığını artırır. Güçlü varsayımlar üst düzey Ana Konu Sınıflandırmada başarılı sonuçlar vermektedir [28-30].

Komşulardan elde edilecek özelliklerin kullanılmasında önemli bir husus hangi komşuların analiz edileceğine karar vermektir. Literatüdeki çalışmalar analiz edilen sayfaya mesafesi en çok 2 olan komşu sayfaların analiz edilmesine odaklanmıştır. Mesafesi ikiden az olan komşular altı farklı sınıfta değerlendirilebilirler. Bunlar; ebeveyn, çocuk, kardeş, eş, dede ve torun olarak adlandırılabilir. Komşu web sayfalarının ayrımına ilişkin bilgiler Şekil 1.2'de verilmiştir.



Şekil 1.2. Komşu Web Sayfları

Genellikle, ebeveyn ve çocuk sayfalarda bulunan metinsel içerikler, hedef sayfadan farklı kategoriye sahip olabilecekleri için komşu sayfalardan oluşturulan alt örneklemler üzerinde işlem yapılan çalışmalar mevcuttur [31-33]. Ayrıca, analiz edilen komşu sayfaların tüm içeriklerinin analiz edilmesi yerine belirli kısımlarının analiz edilmesine üzerine yoğunlaşılmıştır.

Bazı çalışmalar, komşu web sayfalarını analiz ederken bir uzman tarafından elle etiketlenmiş web sayfalarını kullanmışlardır [31], [34], [35]. İnsan tarafından etiketlenen verilerin kullanılması sınıflandırma mekanizmalarından daha başarılı sonuçlar üretir. Ancak, web sayfası sınıflandırılmasına yönelik geliştirilecek bir çalışmada kullanılabilecek etiketli web sayfası sayısı internette yayın yapmakta olan web sayfalarına kıyasla oldukça azdır.

Komşu web sayfalarının tüm içeriklerinin analiz edilmesi oldukça maliyetli bir işlemdir. Bu nedenle komşu web sayfaları analiz edilirken sayfada yer alan bilgilerin bir kısmı kullanılarak özellikler çıkarılmaya çalışılır. Bağlantı metinleri (anchor text) genelde kısadır ve sınıflandırma yapılabilmesi için yeterince bilgi içermezler. Yapılan bazı çalışmalar Bağlantı Metinlerini içeren bir grup metinin işlenmesine yönelik olmuştur [36]. İçeriğin özenle oluşturulduğu web sayfalarında Bağlantı Metinlerinin yanı sıra Başlık Bilgilerinin kullanılması da web sayfalarının sınıflandırılması için faydalı bilgiler sunmaktadır.

1.2 Tezin Amacı

Web Sitesi, birbirlerine çeşitli yollarla bağlantılı olan bir grup web sayfasının oluşturduğu yapıya verilen isimdir. Bir web sitesi altında çok sayıda web sayfası bulunabilir. Web sayfaları da birbirlerine linkler aracılığı ile bağlanırlar.

Web sayfası sınıflandırma (kategorizasyonu), bir web sayfasını daha önceden belirlenmiş olan kategorilerden birisine atama işlemine verilen isimdir. Sınıfllandırma problemi, eğitimli öğrenme (supervised learning) ile çözülebilen bir makine öğrenmesi problemidir. Eğitimli öğrenme problemleri 2 temel aşamadan oluşur. Bu aşamalar; eğitim ve testtir. Belirli sayıda etiketli örnekle eğitilen sistem ile problemi çözecek bir

model oluşturulabilir. Bu model, kategorisi hakkında bilgi sahibi olmadığımız web sayfalarını kategorize edebilmemizi sağlayan yapıdır.

Bir web sayfanın ilgili olduğu kategorinin tespit edilebilmesi için sadece ilgili web sayfası içerisinde yer alan bilgilerin analiz edilmesi yeterli olurken, bir web sitesinin kategorize edilebilmesi için, o web sitesi içerisinde bulunan tüm web sayfalarının da analiz edilmesine ihtiyaç duyulabilir. Literatürde yer alan her iki amaca yönelik geliştirilmiş çalışmalar mevcuttur. Bir web sitesinin ana sayfasının, o web sitesi hakkında özet bir bilgi sunduğu varsayımından yola çıkarak, bazı çalışmalar sadece web sitenin ana sayfasını analiz ederek web site kategorizasyonu yapmışlardır. Bu sayede çok sayıda web sayfanın analiz edilmesine gerek duyulmadan web siteleri kategorize edilebilmektedir.

Web sayfası kategorizasyonu kendi içerisinde birçok alt başlığa ayrılmaktadır. Bu başlıklar web sayfaları üzerinde yaptıkları analiz çıktılarına göre birbirlerinden ayrılırlar. Bu başlıklardan bazıları şunlardır;

- **Ana Konu Sınıflandırması:** Web sayfasını, ilgili olduğu konuya göre sınıflandırma (sanat, iş, spor)
- **Fonksiyonel sınıflandırma:** Web sayfasını, fonksiyonel mekanizmasına göre sınıflandırma (ana sayfa, giriş sayfası, yönetici sayfası)
- **Duygu sınıflandırma:** Yazarın belirli bir konudaki görüşünü anlamaya yönelik yapılan sınıflandırma.

Bunların yanı sıra literatürde web sayfalarının; üsluba göre ya da spam olup olmamalarına göre de sınıflandırma yapan çalışmalarda mevcuttur. Bu çalışmada sadece Ana Konu Sınıflandırması ve Fonksiyonel Sınıflandırma üzerine odaklanılmıştır.

Sınıflandırma mekanizması, sınıf sayısına göre ikili sınıflandırma (binary classification) ve çoklu sınıflandırma (multi-class classification) olarak ikiye ayrılır. İkili sınıflandırmada, veri örneğinin iki sınıftan hangisine ait olduğu tespit edilmeye çalışılır. Çok sınıflandırmada ise bir veri örneği için olası sınıf sayısı ikiden fazladır.

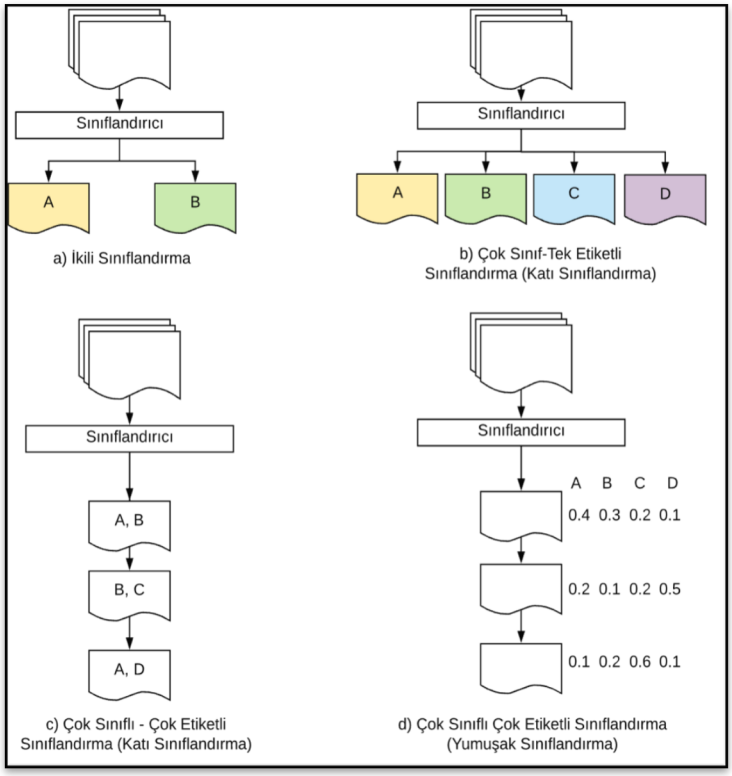
Bir veri örneğine atanacak sınıf bilgisinin sayısına göre sınıflandırma mekanizmaları, tek etiketli veyaçok etiketli olmak üzere ikiye ayrılır. Tek etiketli sınıflandırmada her bir veri

örneğine sadece bir sınıf bilgisi atanır. Çok etiketli sınıflandırmada ise bir veri örneğine birden fazla sınıf etiketi atanabilmektedir.

Bir veri örneğinin sınıfının belirlenmesi işlemi, sınıflandırma yöntemlerini ikiye ayırmaktadır. Bunlar Katı Sınıflandırma (Hard Classification) ve Yumuşak Sınıflandırma (Soft Classification). Katı Sınıflandırmada veri örneğinin belirli bir sınıfa ait olup olmadığı kesin çizgilerle belirlenir. Yumuşak Sınıflandırmada ise sınıflandırmada olasıklıksal çıktılar kullanılır.

Kategorilerin organizasyonuna göre bir sınıflandırma problemi Düz Sınıflandırma ve Hiyerarşik Sınıflandırma olarak ikiye ayrılır. Düz sınıflandırmada her sınıf birbirinden ayrı değerlendirilir ve bir sınıfın diğer sınıfın seçilmesinde herhangi bir etkisi yoktur. Hiyerarşik sınıflandırmada ise önce üst seviye bir sınıflandırma yapılır, ardından ikinci düzey sınıflandırma yapılır. İkinci düzey sınıflandırmada kullanılan kategoriler üst düzey sınıflandırmanın sonucuna bağlıdır.

Sınıflandırma türleri Şekil 1.3 ’te görsel olarak sunulmuştur.



Şekil 1.3. Sınıflandırma Türleri

Bu çalışmada web sayfalarının kategorize edilmesine yönelik bir proje geliştirilmiştir. Geliştirilen sistem çok sınıf tek etiketli sınıflandırma yaklaşımına göre geliştirilmiştir.

1.3 Hipotez

Web sayfası, World Wide Web için hazırlanan ve web tarayıcısı kullanılarak görüntülenebilen dokümanlardır. Web sayfaları çoğunlukla HTML formatında kodlanır, CSS, betik, görsel ve diğer yardımcı kaynaklardan yararlanılarak son görünümüne sahip olur ve işlevsellik kazanır.

1.3.1 Web Sayfası Nedir?

HTML belgelerinin birbirlerine nasıl bağlanacaklarını ve belge içindeki metin ve resimlerin nasıl yerleşeceklerini belirleyen ve etiket (tag) denilen kod parçalarından oluşan bir sistemdir. Internet üzerinde yaşayan World Wide Web (WWW ya da Web), HTML sisteminin arkasında ki etkileşimli, çok platformlu, multimedya ve istemci/sunucu uygulamaları yaratmak için kullanılır.

Bir Web Sayfası, formatlı metinler, ses ve resimlerden oluşan belgelerden oluşur. HTML veriler içerisinde yer alan "tag" parçaları; metnin, resmin ya da başka herhangi bir veri kaynağının tarayıcı tarafından nasıl işleneceğini belirler. Tarayıcı tarafından işlenen HTML veri, kullanıcı için görsel bir hale dönüştürülür.

Bir html etiketi“<” ile açılır ve “/>” ile kapatılır. HTML kullanacağımız bir sayfaya “<html>” etiketibaşlanır ve “</html>” etiketi ile bitirilir.

<html> tagları içerisinde sayfamız iki bölümden oluşur;

- Head – Üst bilgiler kısmı
- Body – Gövde, içerik kısmı

Head (Üst bilgiler kısmı): Sitemize ait anahtar kelimeler, açıklama, site başlığı gibi bilgileri kapsayan kısımdır. Bu kısımda sayfamızın özellikleri kodlanır, kullanıcıya hitap eden üye giriş forumları, yazılar vb. bu kısımda kodlanmazlar.

Body (İçerik kısmı): Adından da anlaşılacağı üzere kullanıcıya hitap eden, formlar, yazılar, görseller, linkler vb. bu kısımda kodlanır ve sitemizin kullanıcıya yönelik tüm işlemlerin yapılacağı kısımdır (Şekil 1.4).

```
<html>
<head>
<title>Site Başlığı</title>
</head>
<body>
Site içerik kısmı.
</body>
</html>
```

Şekil 1.4. Site HTML İçeriği

Bu tarz bir sayfanın çıktısı Şekil 1.5'teki gibi olacaktır.

Şekil 1.5. Sitenin Tarayıcıdaki Görüntüsü

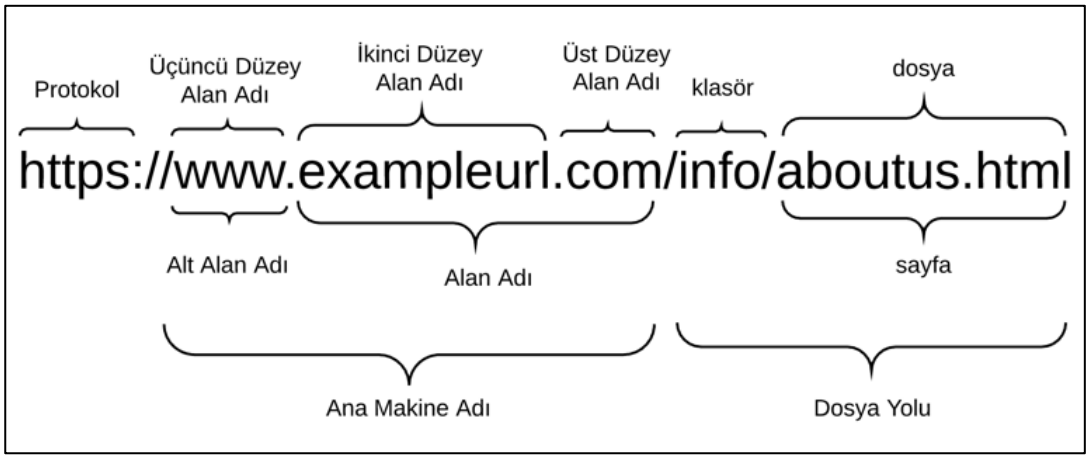
Bir web sayfası programlanırken ilk başta bu HTML yapı kurulur ve içerikle ilgili kodlamalar body etiketleri arasında yapılır.

1.3.2 URL Nedir?

URL (Uniform Resource Locator), en basit şekilde internet sitelerine ulaşmamızı sağlayan adreslerdir. Bir URL, sayfaya erişmek için kullanılan bir Protokol ile başlar. Alan Adı, web sayfasını barındıran sunucuyu tanımlar. Alan adı, kayıt altına alınan ve kullanılması için ücret ödenen alandır. Bir alan adı, İkinci Düzey Alan Adı (İDAA) ve Üst Düzey Alan Adı (ÜDAA) bileşenlerinden oluşur. İDAA, genellikle marka ismini belirtir,

ÜDAA ise alan adı uzantısını belirtir. Ana Makine Adı, alt alan adı ve alan adını kapsayan kısmı belirtir.

FQDN (Fully Qualified Domain Name), alan adı sisteminde bir alan adının tamamını nitelemek için kullanılır. Örneğin; *"secure.serverxfirma.com"* tanımlarken *"secure"* alt alan adı, *"serverxfirma.com*" alan adı ve "secure.serverxfirma.com" FQDN olarak belirtilir. Şekil 1.6, bir URL de bulunan bileşenleri göstermektedir.



Şekil 1.6. URL Bileşenleri

1.3.3 Metin Sınıflandırma ile Web Sayfası Sınıflandırma Arasındaki Farklar

Metin Sınıflandırma ile Web sayfası sınıflandırma problemleri birbirlerine benzerlikler gösterseler de yapısal olarak birbirlerinden çok farklı oldukları bazı noktalar vardır.

İlk olarak klasik metin sınıflandırma problemlerinde sınıflandırılan dokümanlar, çok iyi organize edilmiş, cümleler ve paragraflar arasında anlamsal ve yapısal ilişkilerin olduğu oldukça iyi yapılandırılmış veri kaynaklarıdır. Bu tip dokümanlarda yazının dizaynı tamamıyle yazarın kontrolündedir. Bu nedenle yazı içerisinde yazara özel bazı karakteristik özellikler de bulunabilir. Bu sayede metinin yazarının tespit edilmesi gibi uygulamalar geliştirilebilir. Web sayfalarında ise yapı daha farklıdır. Web sayfaları, HTML içerikleri, etiketler, resimler, ses dosyaları ve videolardan oluşabilir. Metinsel içerikler birbirleri ile anlamsal ya da yapısal ilişki içerisinde bulunmayabilir, tamamen farklı konulara ilişkin yazılar aynı web sayfası içerisinde bulunabilir. Web Sayfasında yer alan bölüm başlıkları gibi metinsel bilgiler tam bir cümle olamayabilirler. İçeriğinde ağırlıklı olarak metinsel içerik barındıran web sayfaları bulunabileceği gibi tamamıyla

görsel elementlerden oluşan web sayfaları da olabilir. Klasik metinsel dokümanların sınıflandırılmasına göre web sayfası sınıflandırma probleminde sınırlı metinsel bilgiye ve daha çeşitli veri tiplerinin analiz edilmesi gerekebilir.

İkinci olarak, web sayfaları tag bileşenlerinden oluşan HTML içeriklere sahiplerdir. Bu HTML içerikler görsel olarak kullanıcıya sunulur. Klasik metinsel dokümanlardan farklı olarak bir web sayfasının ham verinin (source code) yanı sıra bir de görsel çıktısı vardır.

Son olarak Web sayfaları ile diğer web sayfaları ya da dokümanlar arasında hypertext'ler ile kurulmuş olan bağlantılar vardır. Bir web sayfasının sınıflandırılmasında analiz edilen web sayfası içerisinde bulunan köprülerin (hypertext) analiz edilmesi oldukça önemlidir. Web sayfasından başka kaynaklara kurulmuş olan bağlantılar analiz edilen web sayfası hakkında bazı bilgiler vermektedir. Bu bilgiler web sayfasının sınıflandırılmasında faydalı olabilmektedir.

Web sayfası sınıflandırma işlemi yakın zamanlara kadar uzman insanlar tarafından bilgisayardan yardım almadan yapılmaktaydı. Ancak, son yıllarda yapılan çalışmalar ile bu işlem yarı-otomatik ya da tam otomatik mekanizmalar ile yapılabilir hale gelmiştir.

Web sayfası sınıflandırma problemi, klasik metin sınıflandırmadan daha zor problemdir. Geleneksel makine öğrenmesi algoritmaları ile yapılmış birçok çalışma literatürde mevcuttur. Web sayfaları kategorize edilmek üzere işlenmeden bazı ön işlemlerden geçirilmelidir. Bu işlemler, analiz edilecek veriyi makine öğrenmesi algoritmalar ile işlenebilmek üzere hazır hale getirmeye yarar. Ayrıca, yapılacak bazı veri ön işleme yöntemleri ile analiz edilecek web sayfa içeriği uygun formatta indekslenir ve bu sayede veri bellekte daha az yer tutacak hale getirilir.

Web sayfalarının diğer metinsel dokümanlardan en belirgin farklarından birisi sayfaların HTML tagları ile oluşturulmuş yapılara sahip olmasıdır. Literatürde yer alan bazı web sayfası sınıflandırma çalışmaları HTML taglarını temizleyerek geliştirilen sınıflandırma mekanizmalarında bu bilgileri kullanmamışlardır. Ancak, yapılan çalışmalar göstermektedir ki HTML etiketlerin kullanılması sınıflandırma başarısını artırıcı etki oluşturmaktadır.

1.3.4 DDİ İçin Derin Öğrenme Tabanlı Yaklaşımlar

Özelliklerin tanımlanması ve gösterimi makine öğrenmesi problemleri için oldukça önemli bir adımdır. Son zamanlarda, önceden eğitilmiş Kelime Gösterimlerinin kullanılması ve derin sinir ağlarının hızlı gelişimi, çeşitli DDİ problemlerinin çözülmesine yeni bir ilham kaynağı olmuştur. Kelime Gösterimleri (ya da Kelime Vektörleri), veri kısıtlaması sorununu büyük ölçüde azaltan bir özellik gösterimi çeşitidir [37]. Mikolov ve arkadaşlarının yaptığı çalışma [38], Kelime Vektörlerinin anlamsal düzenleri yakalayabildiğini göstermektedir. Kelime Gösterimleri hakkında detaylı bilgiler Bölüm 2.4.3'te verilmiştir.

Metin sınıflandırma için kullanılan birçok derin öğrenme mimarisi bulunmaktadır. Socher ve arkadaşları [39], [40] bir cümlenin temsil edilmesi için Rekürsif Sinir Ağları (RSA - Recursive Neural Network) mimarilerini önermiştir. RSA yapısında bir cümle anlamsal içeriğine göre bir ağaç yapısı oluşturularak analiz edilebilmektedir. Bu yapıda modelin performansı, oluşturulacak ağaç yapısına bağlıdır. Bu tarz bir ağacın oluşturulması $\mathcal{O}(n^2)$ zaman karmaşıklığına sahip bir problemdir (n metin uzunluğunu temsil eder). Bu nedenle bu yapı uzun cümle ve dokümanlarda kullanılması verimli olmamaktadır.

Metinlerin anlamsal içeriklerine göre analiz edilmesine olanak tanıyan bir diğer derin öğrenme mimarisi ise Yinelemeli Sinir Ağı (YiSA - Recurrent Neural Network) mimarileridir [40], [42]. Bu yapıda bir metin kelime kelime işlenmektedir. Bir kelime analiz edilirken o kelimeden önce kullanılmış olan kelimelere ilişkin bilgiler de gizli katmanlarda yer alır. Bu yapının zaman karmaşıklığı ise O(n)'dir. Bu nedenle YiSA, uzun metin ve dokümanlar için RSA yapısından daha efektif çalışabilmektedir. Bu yapının bir dezavantajı ise YiSA mimarisinin bias'a elverişli olmasıdır. Bu yapıda son gelen kelime diğer kelimelere oranla sistemde daha baskın rol oynamaktadır. Halbuki ağırlığı fazla olan kelimeler cümle ya da doküman içerisinde herhangi bir yerde bulunabilir.

Evrişimsel Sinir Ağları (ESA - Convolutional Neural Networks) yapıları DDİ problemleri için kullanılan unbiased yapıya sahip bir derin öğrenme yaklaşımıdır [40]. Bu yapıda ayırt edici kelime grupları max-pooling katmanları ile çıkarılabilmektedir. Bu sayede anlamsal ilişkileri yakalamada YiSA yapılarından daha başarılı olduğu söylenebilir.

Ayrıca ESA yapısının zaman karmaşıklığı da O(n)'dir. Bu yapı, veri üzerinden belirli bir sayıda uzunluğa sahip bir çerçeve gezdirilmesi ile çalışır. Çerçeve uzunluğunun belirlenmesi önemli bir problemdir. Küçük çerçeveler bazı kritik anlamsal bilgilerin kaybolmasına neden olabilirken, büyük çerçeveler kullanıldığında ise çok fazla sayıda parametrenin öğrenilmesini gerekeceği için eğitim işlemi zorlaşmaktadır.

YiSA ve ESA mimarilerinin eksiklerinin giderilmesi ve daha gürbüz bir sistemlerin geliştirilmesi için Evrişimli Yinelemeli Sinir Ağları (EYiSA) mimarileri ve Çift Yönlü Yinelemeli Sinir Ağı (ÇYYiSA) mimarileri önerilmiştir. Lai ve arkadaşları EYiSA mimarilerini metin sınıflandırma probleminin çözülmesi için önermişlerdir [41]. Bu çalışmada, max-pooling katmanı ile metin sınıflandırmada önemli rol oynayan özellikler belirlenmekte ve ardından BRNN mimarisi ile işlenmeye devam edilmektedir. Lai ve arkadaşlarının metin sınıflandırması için geliştirmiş olduğu EYiSA yaklaşımının zaman karmaşıklığı da O(n)'dir.

Bu amaca yönelik geliştirilen birçok çalışma [8], [10], [11] mevcuttur. Derin Öğrenme yaklaşımları sayesinde başarılı metin sınıflandırma çalışmaları yapılabilmektedir. Bu çalışmada metin sınıflandırmaya yönelik derin öğrenme yaklaşımları kullanılarak web sayfalarının sınıflandırılması amaçlanmıştır.

1.4 Veri Ön İşleme

Bir web sayfasının sınıflandırılmasında sayfanın metinsel içeriğinin önemi büyüktür. Web sayfası içerisinde yer alan metinsel bilgilerin Doğal Dil İşleme teknikleri ile işlenmeden önce bazı ön işlemlere tabi tutulması gerekmektedir. Bu ön işlemler şunlardır;

- Noktalama işaretlerinin temizlenmesi
- Durak Kelimelerinin (Stop Words) temizlenmesi
- Kelime köklerinin bulunması

Noktalama işaretleri, durak kelimeleri ve kelimelere gelen ekler sınıflandırma mekanizması için değeri olmayan bilgilerdir. Bu bilgilerin temizlenmesi ile yapılan

işlemler ile daha sağlıklı sonuçlar elde edilebilmektedir. Kelime köklerinin bulunmasına yönelik İngilizce dili için kullanılan en yaygın yöntem Porter Algoritmasıdır [43], [44].

Literatürde yer alan birçok çalışma Kelime Torbası (Bag of Words) mantığına göre çalışmaktadır. Bu çalışma mantığına göre her kelime kökü bir özellik olarak alınarak her bir web sayfası bir vektör olarak ifade edilir. Tüm web sayfaları için bu şekilde vektörizasyon yapılarak oluşturan veri setinde çok fazla sayıda özellik ortaya çıkmaktadır. Çok sayıda veri örneği için çok fazla sayıda özelliğin işlenmesi uzun sürmekte, bazı algoritmalar tarafından bu tip bir veri setinin işlenmesi oldukça zordur. Bu nedenle metinsel içerik işlenirken Kelime Torbası mantığının yanı sıra n-gram veri setleri üzerinde çalışan akademik çalışmalar da yapılmıştır. Kelime tabanlı n-gram veri setleri kelimeleri tamamıyla birbirlerinden ayrık bileşenler olarak değerlendirmek yerine kelimeleri belirli gruplar olarak işlemeye de olanak tanıyan veri setleridir. Bu veri setlerinde, her kelime kendisinden sonra n adet kelime ile birlikte değerlendirilerek bir özellik olarak kullanılmaktadır. Bu sayede deyimler, tamlamalar gibi birden fazla kelimenin bir araya gelerek anlamlandığı yapılar bu veri setinde rahatlıkla işlenebilmektedir.

N sayısı istenilen bir rakam seçilebilir. (Bag of words yaklaşımı n=1 olan n-gram verisetine denk gelmektedir.) N sayısı arttıkça daha fazla sayıda anlamsal bağ içeren kelime grupları bir arada analiz edilebilmektedir. Ancak, n sayısı çok fazla olduğu takdirde bir web sayfasını birbirine gerçekten çok benzer dokümanlar olsa bile tamamen farklı dokümanlar olarak algılanabilmektedir. Literatürde yer alan n-gram veri setlerinin birçoğu kelime tabanlı 2, 3, 4 gram veri setlerinden oluşur.

Bazı web sayfaları metinsel bilgiden çok görsel elementlerden oluşabilir. Görsel elementler aynı zamanda metinsel bilgi de içerebilir. Ancak, görsel elementlerdeki metinsel bilginin makine öğrenmesi algoritmaları tarafından işlenebilmesi için öncelikle metin olarak elde edilmesi gerekir. Resimlerdeki yazıların tanınması problemine Optik Karakter Tanıma (Optical Character Recognition) problemi denir. Bu işlemi yapan hazır kütüphaneler kullanılarak resimlerdeki metinler çıkarılabilir ve sınıflandırma mekanizmasında kullanılabilir.

1.5 Özellik Azaltımı

Hangi tip özelliklerin kullanılmasının önemli olmasının yanı sıra kullanılacak özelliklerin ağırlıklandırılması da web sayfalarınınsınıflandırılmasında önemli bir adımdır. Doğal Dil İşleme tabanlı yöntemlerde kelime veya kelime grupları vektör şeklinde ifade edilir, ardından makine öğrenmesi algoritmaları ile işlenir. Kelime / kelime gruplarının vektörizasyonu sonucunda çok fazla sayıda özellik elde edilir. Çok sayıda özellik bilgisinin makine öğrenmesi algoritmaları tarafından işlenmesi maliyetli olmaktadır. Bu nedenle sınıflandırma için pozitif yönde katkı sağlamayan özelliklerin özellik listesinden çıkarılması performans iyileştirilmesi için oldukça önemlidir. Bunların yanı sıra sınıflandırmaya katkısının en fazla olduğu bazı alt özellik setleri ile çalışmak performans için olumlu etki oluşturmaktadır. Bu nedenle, özellikle metin tabanlı veriler üzerinde sınıflandırma problemleri için özellik azaltılması adımı oldukça önemlidir. Web sayfalarının sınıflandırılması için yapılan bazı akademik çalışmalar, özelliklerin azaltılması için bazı yöntemler deneyerek karşılaştırılmıştır.

Özellik azaltılması tüm verinin işlenmesi sonucunda elde edilen özellikler üzerinde yapılan analizlerle azaltılabileceği gibi, işlenen verinin azaltılması yoluyla daha az özellik çıkarılması şeklinde de yapılabilmektedir [45]. Özellik seçimi için Bilgi Kazancı (Information Gain) yöntemini kullanan çalışmalar mevcuttur [35]. Doğal Dil İşleme alanında bilinen bir diğer yaklaşım olan Gizli Anlam İndeksleme (Latent Semantic Indexing-LSI) da metin tabanlı veri setleri için özellik azaltılması için kullanılabilir. Ancak, bu yöntemin hesaplamasal karmaşıklığı fazla olduğu için yüksek boyutta veriler için uygulanması efektif olmamaktadır. LSI kullanan ve bu yöntemin geliştirilmesine yönelik geliştirilmiş çalışmalar literatürde bulunmaktadır [46], [47]. Büyük veri setleri üzerinde bu yöntemin kullanılması için verimlilik optimizasyonu için daha fazla çalışma yapılmasına ihtiyaç duyulmaktadır [24].

Mangai ve Kumar [48], web sayfası sınıflandırılması için geliştirilen sistemde özellik azaltımı için CfsSubsetEval algoritmasını kullanmış ve ardından elde edilen özellikleri C4.5 karar ağacı sınıflandırcısı ile test etmiştir. Gerçekleştirilen testler ile özellik azaltımı işleminin sınıflandırma başarısını artırdığı gözlenmiştir.

[41] tarafından yapılan başka bir çalışmada, özellik azaltımı için Genetik Algoritma kullanılmıştır. Yapılan çalışmada özellik seçimi için genetik algoritmanın kullanılması ile sınıflandırma başarısının artırılmasının yanında çalışma süresinin de azaltılması sağlanmıştır. Özellik azaltılması adımı sonrasında elde edilen özellikler ile yapılan testler ile kNN algoritması ile %96 başarı elde edilmiştir.

BÖLÜM 2

MATERYAL VE YÖNTEM

Geliştirilen içerik tabanlı sınıflandırma sistemi sadece İngilizce dilinde hazırlanmış web sayfalarının sınıflandırılması üzerine tasarlanmış olup, ilerleyen aşamalarda sınıflandırılabilen dil sayısının artırılması hedeflenmektedir.

İçerik tabanlı web sayfalarının sınıflandırılması için geliştirilen projelerde kullanılmak üzere yayınlanan birçok veri seti bulunmaktadır. Bu çalışma kapsamında kullanılan verisetine ilişkin detaylar Bölüm 2.1'de verilmiştir. Geliştirilen projede web sayfalarının sınıflandırılması için derin öğrenme tabanlı bir YiSA yaklaşımı kullanılmıştır. Kullanılan bu yaklaşıma ilişkin teknik detaylar Bölüm 2.2'de verilmiştir. Son yıllarda derin öğrenme çalışmalarından sıklıkla kullanılan Öğrenme Transferine ilişkin bilgiler Bölüm 2.3'te, derin öğrenme yaklaşımlarında kullanılan Kelime Gösterimleri Bölüm 2.4'te ve web sayfalarının sınıflandırılması için tasarlanan derin öğrenme mimarisine ilişkin detaylar da Bölüm 2.5'te verilmiştir.

Veri saklama platformu olarak ElasticSearch, raporlama ara birimi olarak Kibana kullanılmıştır.

2.1 Veri Seti

İçerik tabanlı web sayfalarının sınıflandırılması için literatürde birçok veri seti bulunmaktadır. Bu verisetleri Bölüm 1.5.3'te açıklanmıştır. Web sayfaların sınıflandırılması için kullanılan veri setleri iki aşamalı olarak düşünülmelidir. Bunlar;

- **Birinci Aşama:** Sadece web sayfası ve kategoriler yer alır.

- **İkinci Aşama:** Web sayfalarının sınıflandırılabilmeleri için kullanılan verilerdir.

Sınıflandırma için kullanılan veriler web sayfasının metinsel içeriği olabileceği gibi komşu sayfalardan elde edilecek bilgiler de olabilir. Web sayfalarının sınıflandırılması için kullanılabilecek özellikler Bölüm 1.5.3'te açıklanmıştır.

Web sayfalarının sınıflandırılmasında kullanabilecek veri setlerinden birisi Roksit firmasının web sınıflandırma veri tabanıdır [49]. Bu veri tabanında milyonlarlarca web sitesi sınıflandırılmıştır. Sınıflandırma işlemi esnasında web sayfası içeriği kullanıldığı gibi bazı ağ (network) bazındaki özelliklerden de faydalanılmıştır. Kullanılan özellik setinin geniş olması başarılı bir web sınıflandırıcı oluşturulmasına olanak tanımıştır.

Roksit firmasının web sayfası sınıflandırma mekanizması hiyerarşik bir sınıflandırma yapmaktadır. Bu sistemde, 37 ana kategori, 121 alt kategori bulunmaktadır. Ayrıca, kategoriler 5 farklı güvenlik seviyesine göre gruplandırılmıştır [49]. Sınıflandırma mekanizması bir miktar Yanlış Pozitif (False Positive) değerine sahiptir. Birinci aşama verisi için Roksit firmasından sağlanan web sayfası sınıflandırma verileri kullanılmıştır. Ardından ikinci aşama veriler için bir web crawler modülü geliştirilmiş ve veriler bu şekilde toplanmıştır. Birinci ve ikinci aşamada kullanılan veriler Bölüm 2.1.1 ve 2.1.2'de açıklanmıştır.

2.1.1 Birinci Aşama Veri Seti

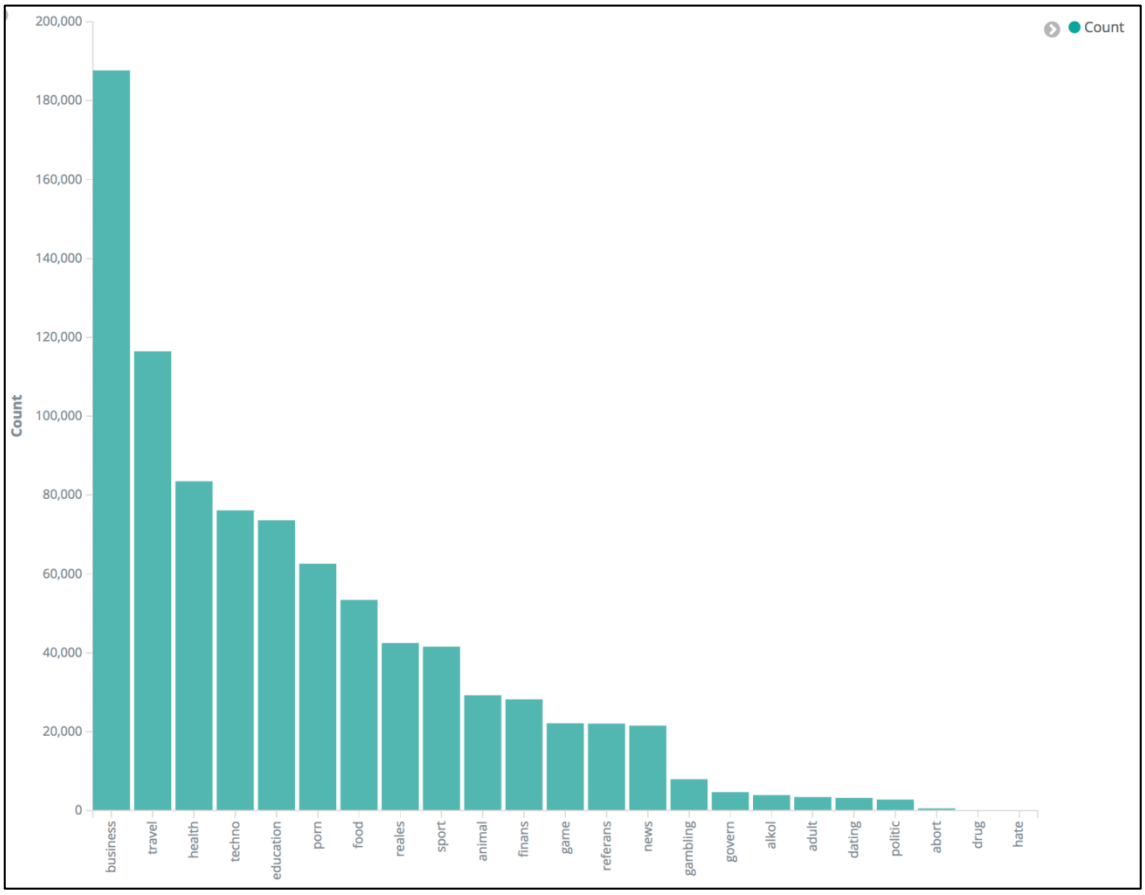
Roksit veri tabanında 37 ana kategori ve 121 alt kategori bulunmaktadır. Mevcut sınıflar içerisinde birbirlerine oldukça yakın sınıflar bulunmaktadır. Sınıflandırma mekanizmasının performansının yüksek olması amacıyla birbirine yakın bazı sınıflar birleştirilmiş ve bazı kategoriler de içerik tabanlı sınıflandırma mekanizması ile işlenmemek üzere çıkarılmıştır. Bu işlemlerin uygulanması ile sınıflandırma mekanizması için işlenecek sınıf sayısı azalmıştır.

Web sayfalarının sınıflandırılması amacıyla kullanılacak verisetinin ilk aşamasında Roksit firmasından 23 farklı kategoriye ait 887,184 adet FQDN (Fully Qualified Domain Name) (bkz. Bölüm 1.3) alınmıştır. Alınan veriler, dünya genelinde ziyaret edilen İngilizce içeriğe sahip FQDN'lerden oluşmaktadır. Kategorizasyon sisteminde kullanılacak 18 kategori Çizelge 2.1'de verilmiştir.

Çizelge 2.1. Kategoriler

adult	alkol	animal	dating	education	business
finans	food	gambling	game	govern	reference
health	news	politic	porn	reales	abort
sport	techno	travel	hate	drug	

Verilerin kategorilere göre dağılımı Şekil 2.1'de verilmiştir.



Şekil 2.1. Birinci Aşama Veri Setinin Kategorilere Göre Dağılımı

Her kategori içerişinde bulunan veri miktarı Çizelge 2.2'de verilmiştir.

Çizelge 2.2. Birinci Aşama Verilerinin Kategorilere Göre Dağılım Verileri

Kategori	Veri Örneği Sayısı	Kategori	Veri Örneği Sayısı
business	187,664	referans	22,057
travel	116,434	news	21,543
health	83,502	gambling	7,953
techno	76,115	govern	4,686
education	73,608	alkol	3,931
porn	62,586	adult	3,429
food	53,410	dating	3,223
reales	42,481	politic	2,800
sport	41,565	abort	555
animal	29,242	drug	33
finans	28,207	hate	24
game	22,136		

2.1.2 İkinci Aşama Veri Seti

İçerik tabanlı web sayfası sınıflandırması yapabilmek için öncelikle hangi türde verilerin kullanılacağının belirlenmesi gerekmektedir. Kullanılacak veriler, analiz edilen sayfa içeriğinden elde edilebileceği gibi komşu sayfalardan da elde edilebilmektedir (bkz. Bölüm 1.5.3). Bu çalışmada, web sayfalarının sınıflandırılması için analiz edilen sayfalardan elde edilen bilgiler kullanılmıştır.

Bir web sayfasının hedef kullanıcı kitlesine başarı ile ulaşabilmesi web sayfası içeriğinin iyi yapılandırılmış olması gerekmektedir. Arama motorları tarafından indexlenip arama sorgularından hedef kullanıcılara üst sıralarda görünebilmek için web sayfaları için oldukça önemlidir. Web sayfası içeriğinin güzel bir şekilde indekslenmesi ve web sayfasının arama motorlarında yüksek sıralarda görünmesi için kullanılan yöntemlerden birisi Meta Etiketlerdir. İyi yapılandırılmış bir web sayfası içerisinde meta etiketler bulunur. Bu etiketler, web sayfası yöneticisi tarafından girilen ve web sayfasının amacını/işlevini gösteren özet bilgiler içerir. Meta Etiketlerin en önemlileri başında şu etiketler gelmektedir;

- Başlık (Title)
- Açıklama (Description)
- Anahtar Kelimeler (Keywords)

Meta etiketler web sayfasının kaynak kodunda yer alır ve kullanıcı tarafından görüntülenmek üzere tarayıcı tarafından ekrana basılmayabilir. Örneğin, Hürriyet Emlak sitesi [e50] için meta etiketlerin bulunduğu kaynak kodlar Şekil 2.2'de verilmiştir. (Kaynak kodlar chrome tarayıcıdan [51]'deki adresten görüntülenebilir.)

```
    <title>Hürriyet Emlak – Türkiye'nin En Büyük Emlak Sitesi</title>
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <script src="//cdn.optimizely.com/js/9418201852.js"></script>
            <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta name="description" content="Hürriyet Emlak yüzbinlerce satılık ve
kiralık daire, konut, sahibinden arsa, işyeri, dükkan, mağaza ve konut projelerini
bulabileceğiniz tek emlak sitesi!" />
        <meta name="keywords" content="sahibinden, emlak, satılık daire, kiralık
daire, sahibinden satılık daire, konut projeleri, emlak ilanları" />
```

Şekil 2.2. Meta Etiket Örneği

Şekil 2.2'de görüleceği Hürriyet Emlak sitesinde yer alan meta etiketler şu şekildedir;

- **Başlık:** Hürriyet Emlak – Türkiye'nin En Büyük Emlak Sitesi
- **Açıklama:** Hürriyet Emlak yüzbinlerce satılık ve kiralık daire, konut, sahibinden arsa, işyeri, dükkan, mağaza ve konut projelerini bulabileceğiniz tek emlak sitesi!
- **Anahtar Kelimeler:** sahibinden, emlak, satılık daire, kiralık daire, sahibinden satılık daire, konut projeleri, emlak ilanları

İyi yapılandırılmış bir web sayfasında bu bilgilerin girilmiş olması beklenir. Ancak, bu bilgiler girilmeden oluşturulan web sayfaları da bulunmaktadır.

İkinci aşama verilerinin elde edilmesi için birinci aşamada elde edilen İngilizce içeriğe sahip 887,184 adet FQDN için bir crawler modülü yazılmıştır. Bu modül, ilgili sayfanın -varsa- meta etiketlerini toplamaktadır. Birinci aşamada yer alan web sayfalarının içerisinde birçok dilde hazırlanmış web sayfaları bulunmaktadır. Bu proje kapsamında İngilizce dilinde hazırlanmış web sayfaları sınıflandırılmaya çalışılmıştır. Bu nedenle ikinci aşama veriler elde edilirken, web sayfalarının dillerinin de tespit edilmesine

ihtiyaç duyulmuştur. Web sayfaları crawl edilirken web sayfalarının metinsel içeriklerine göre dil tespiti yapılmasına ihtiyaç duyulmuştur. Bu işlem için Google'un dil tespit modülü kullanılmıştır [52].

Crawl işlemi esnasında bir web sayfasına ilişkin aşağıdaki bilgiler toplanmıştır.

- Web sayfası içeriğinin dili
- Başlık
- Açıklama
- Anahtar Kelimeler
- Metinsel İçerik

Bu proje kapsamında sadece en yaygın kullanılan İngilizce dili için sınıflandırma mekanizması tasarlanmıştır. Roksit firmasından elde edilen birinci aşama veriler İngilizce içeriğe sahip web sayfalarıdır.

İyi yapılandırılmış bir web sayfasında meta etiketlerin girilmiş olması beklenir. Ancak, bazı web sayfası yöneticileri bu adımı önemsememektedir. Dolayısıyla bazı web sayfalarında meta etiketler bulunamayabilmektedir. Bu durumdaki sayfaların da sınıflandırılabilmesi için web sayfasında kullanıcı tarafından görüntülenen metinsel içerik bilgisi kullanılabilir. Metinsel içeriğin sistem eğitim aşamasında kullanılmasının başarı oranına etkisi incelenirken meta etiketlerin toplanması için crawler modülü çalıştırılmış ve her FQDN için veriler toplanmıştır. Crawl edilen web sayfası için bir veri örneği Şekil 2.3'te verilmiştir.

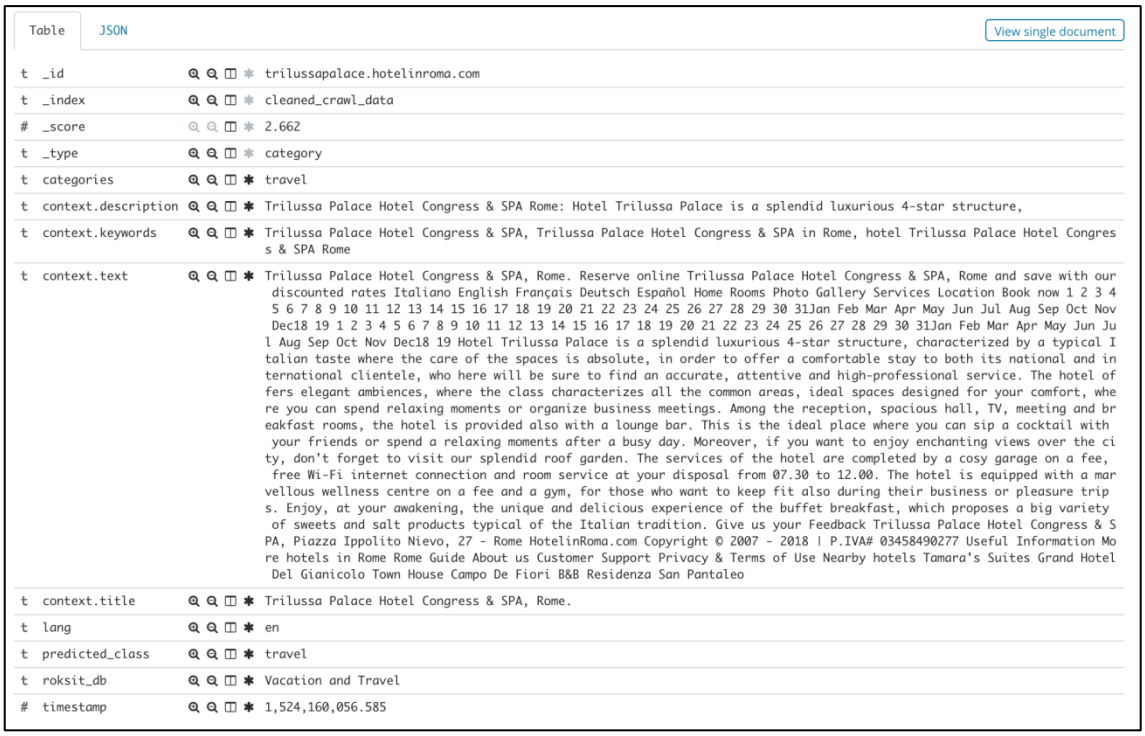


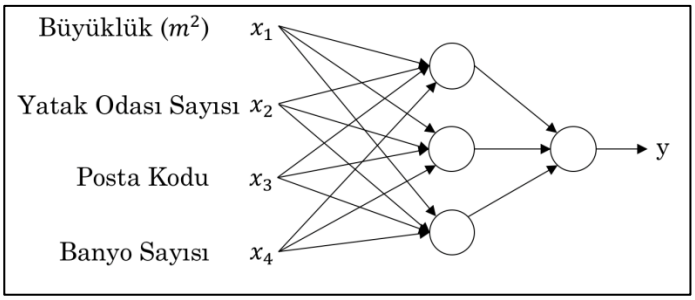
Table JSON View single document

t	_id	trilussapalace.hotelinroma.com
t	_index	cleaned_crawl_data
#	_score	2.662
t	_type	category
t	categories	travel
t	context.description	Trilussa Palace Hotel Congress & SPA Rome: Hotel Trilussa Palace is a splendid luxurious 4-star structure,
t	context.keywords	Trilussa Palace Hotel Congress & SPA, Trilussa Palace Hotel Congress & SPA in Rome, hotel Trilussa Palace Hotel Congress & SPA Rome
t	context.text	Trilussa Palace Hotel Congress & SPA, Rome. Reserve online Trilussa Palace Hotel Congress & SPA, Rome and save with our discounted rates Italiano English Français Deutsch Español Home Rooms Photo Gallery Services Location Book now 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec18 19 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec18 19 Hotel Trilussa Palace is a splendid luxurious 4-star structure, characterized by a typical Italian taste where the care of the spaces is absolute, in order to offer a comfortable stay to both its national and international clientele, who here will be sure to find an accurate, attentive and high-professional service. The hotel offers elegant ambiences, where the class characterizes all the common areas, ideal spaces designed for your comfort, where you can spend relaxing moments or organize business meetings. Among the reception, spacious hall, TV, meeting and breakfast rooms, the hotel is provided also with a lounge bar. This is the ideal place where you can sip a cocktail with your friends or spend a relaxing moments after a busy day. Moreover, if you want to enjoy enchanting views over the city, don't forget to visit our splendid roof garden. The services of the hotel are completed by a cosy garage on a fee, free Wi-Fi internet connection and room service at your disposal from 07.30 to 12.00. The hotel is equipped with a marvellous wellness centre on a fee and a gym, for those who want to keep fit also during their business or pleasure trips. Enjoy, at your awakening, the unique and delicious experience of the buffet breakfast, which proposes a big variety of sweets and salt products typical of the Italian tradition. Give us your Feedback Trilussa Palace Hotel Congress & SPA, Piazza Ippolito Nievo, 27 - Rome HotelinRoma.com Copyright © 2007 - 2018 I P.IVA# 03458490277 Useful Information More hotels in Rome Rome Guide About us Customer Support Privacy & Terms of Use Nearby hotels Tamara's Suites Grand Hotel Del Gianicolo Town House Campo De Fiori B&B Residenza San Pantaleo
t	context.title	Trilussa Palace Hotel Congress & SPA, Rome.
t	lang	en
t	predicted_class	travel
t	roksit_db	Vacation and Travel
#	timestamp	1,524,160,056.585

Şekil 2.3. Veri Örneği

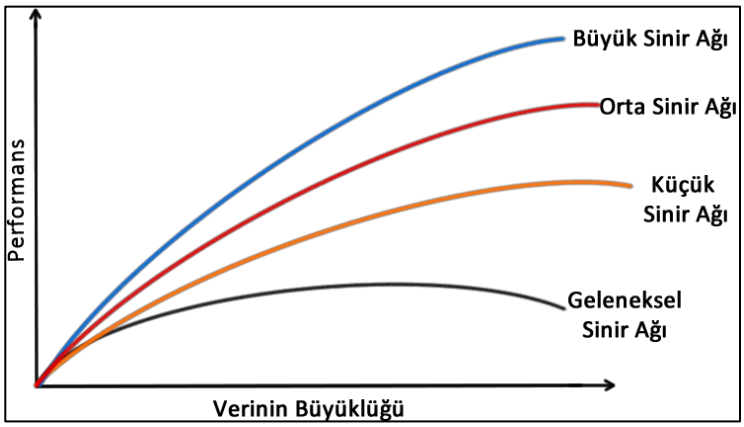
2.2 Geleneksel Derin Öğrenme Yaklaşımları

Temeli Yapay Sinir Ağlarına dayanan Derin Öğrenme Yaklaşımları, çok sayıda katman ve nörondan oluşur. Geliştirilen ilk mimarilere, İleri Beslemeli Yapay Sinir Ağı (İBYSA) isminde literatürde sıkça rastlanmaktadır. Bu ağ yapısında nöronlar birbirlerine bağlıdır. Bir nörona birçok girdi olabilir ancak her nöronun bir adet çıktısı bulunabilir. Örnek bir ağ yapısı Şekil 2.4'te verilmiştir. Şekil 2.4'te tasarlanan ağ yapısı ev fiyatı tahmini probleminin çözülmesi için modellenmiştir. Bu nedenle girdi bilgileri bu problem ile ilişkilidir.



Şekil 2.4. Örnek bir Yapay Sinir Ağı

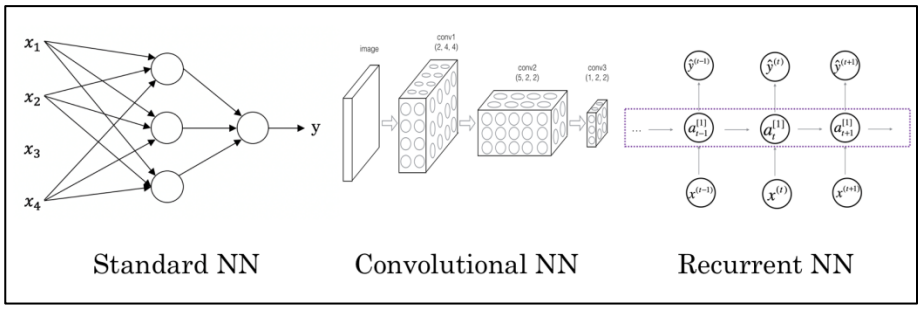
İBYSA'da öğrenme işlemi düğümler arasındaki ağırlıkların öğrenilmesi ile sağlanır. Her veri örneği düğümler arası ağırlıklarla bir miktar güncelleştirmeye neden olur. Çok sayıda verinin yeterli sayıda kullanılması ile problem tanımına ilişkin doğru parametreler öğrenilebilinir. Derin Öğrenme yaklaşımları ile yüksek başarı elde edilebilmesi için çok miktarda veriye ihtiyaç duyulur. Katman sayısının artması derin öğrenme yaklaşımlarına daha karmaşık problemleri modelleyebilme yeteneği kazandırır. Karmaşık problemler için artan katman sayısı ile doğru orantılı artan veri performansının da artmasında etkilidir. Karmaşık problemlerde kullanılabilecek sinir ağı boyutlarının artan veri ile ilişkisi Şekil 2.5'te verilmiştir.



Şekil 2.5. Veri Büyüklüğü ile Sinir Ağı boyutu Arasındaki İlişiki

Karmaşık problemler için sinir ağını derinleştirmek başarılı sonuçların alınmasını sağlayabilmektedir. Ancak, çözülen problemin yapısına göre az katmandan oluşan sığ mimariler de başarılı sonuçlar verebilmektedir.

Bazı problem türleninin daha başarılı çözülebilmeleri için geleneksel derin öğrenme yaklaşımlarının yanısıra farklı derin öğrenme mimarileri de geliştirilmiştir. Örneğin, bazı görüntü işleme problemleri için Evrişimli Sinir Ağları (Convolutional Neural Network) yapıları oldukça başarılı sonuçlar üretmektedir. Bazı DDİ problemleri için ise YiSA (Yinelemeli Sinir Ağları, RNN), ÇYYSA (Çift Yönlü Yinelemeli Sinir Ağı, BRNN), EYSA (Evrişimli Yinelemeli Sinir Ağı, RCNN) gibi yaklaşımlar önerilmiştir. Geliştirilen farklı derin öğrenme yaklaşımlarına ilişkin örnek Şekil 2.6'da verilmiştir.



Şekil 2.6. Özelleştirilmiş Bazı Derin Öğrenme Yaklaşımları

Derin Öğrenme yaklaşımının teknik anlatımında kullanılan notasyon şu şekildedir;

- Üstindis (superscript) de yer alan [l]üzerinde bulunduğu objenin hangi katmanda yer aldığını gösterir.
- Üstindis [i] objenin sırasını belirtir.

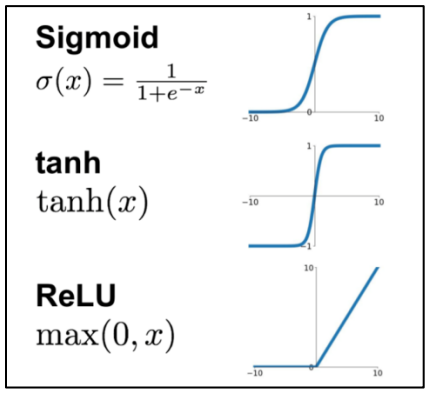
Derin öğrenme yaklaşımlarında, öğrenme işleminin tamamlanması üç temel adımın birbiri ardınca uygulanması ile sağlanır. Bu adımlar;

- İleri Yayılım (Forward Propagation)
- Geri Yayılım (Backward Propagation)
- Parametre Güncelleme (Update Parameter)

İleri yayılım adımında her bir düğümde, verilen girdilere ilişkin aktivasyon değeri hesaplanır. Girdiler ile girdilere ilişkin ağırlık değerlerinin çarpılmasının ardından bias değeri ile toplanır. Ardından elde edilen değer, aktivasyon fonksiyonuna verilir. Aktivasyon fonksiyonu, elde edilen değeri [0, 1] arasına indirgeyen matematiksel bir fonksiyondur. Aktivasyon değerinin hesaplanması Denklem 2.1 ile yapılır.

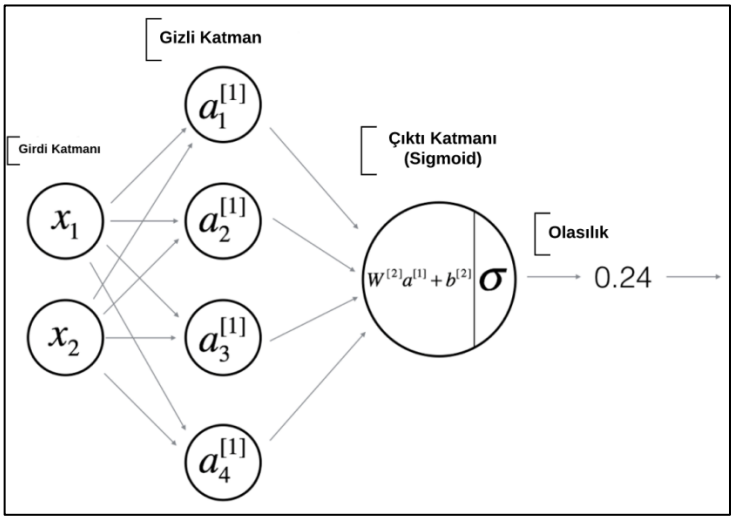
$$\hat{y} = \sigma\,(W^T x + b) \tag{2.1}$$

Literatürde aktivasyon fonksiyonu olarak kullanılan birçok fonksiyon bulunmaktadır. Bunlardan bazıları; sigmoid, RELU, tanh ‘dır. Aktivasyon fonksiyonları Şekil 2.7’de verilmiştir.



Şekil 2.7. Aktivasyon Fonksiyonları

Ağda bulunan bir düğüm üzerinde hesaplanan işlemler Şekil 2.8'de gösterilmiştir.



Şekil 2.8. Bir düğümde yapılan Hesaplamalar

İleri yayılım adımına başlanmadan önce tüm ağırlık w ve b değerlerinin ilklendirilmesi gerekmektedir. Öğrenme işlemi bu değerlerin adım adım güncellenmesi şeklinde gerçekleşecektir. İlklendirme işleminde tüm değerlere 0 değeri atanabileceği gibi rastgele değerler ile de ilklendirme yapılabilmektedir.

W, b değerlerinin ne yönde ve miktarda güncelleneceğinin belirlenebilmesi için bir hata fonksiyonuna (loss function, L) ihtiyaç duyulur. Hata fonksiyonu olması gereken değer y ile hesaplanan değer $\hat{y}$ arasındaki hatayı hesaplar. Literatürde kullanılan birçok farklı hata fonksiyonu bulunmaktadır. Kullanılan hata fonksiyonlarından birisi olan Cross Entropy Cost Function Denklem 2.2 hesaplanır.

$$\mathcal{L}(\hat{y}, y) = -\big(y \log(\hat{y}) + (1-y)(\log(1-\hat{y}))\big) \tag{2.2}$$

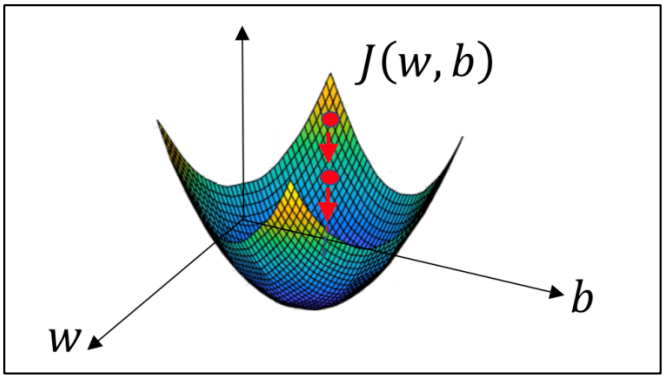
Hata fonksiyonu, algoritmamızın veri örneği bazında ne kadar iyi çalıştığını ölçtüğümüz metriktir.

Hata fonksiyonu her bir veri örneği için ayrı ayrı hesaplanır. Bu nedenle sistemimizin genel hata durumunu gösteren bir fonksiyona ihtiyaç duyulur. Bu fonksiyona Maliyet Fonksiyonu denir. Maliyet fonksiyonu her veri örneği için hesaplanan hata fonksiyonunun ortalamasıdır. Maliyet fonksiyonu (J) Denklem 2.3 ile hesaplanır;

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}\,(y,\hat{y}) \tag{2.3}$$

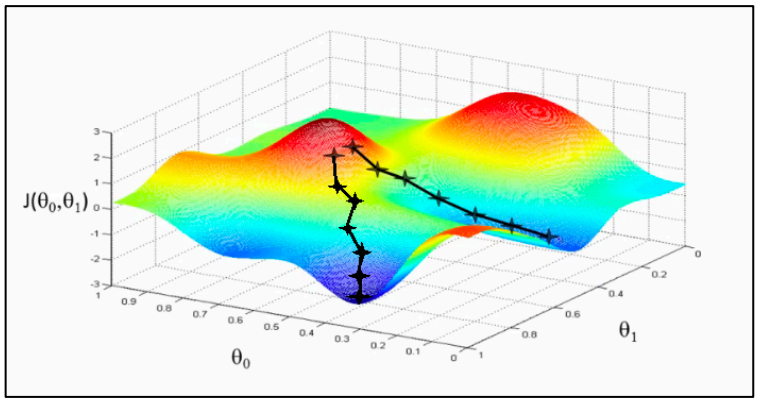
Geri Yayılım adımında, hesaplanan maliyet fonksiyonuna göre bu maliyeti minimize edecek (w,b) güncellenme miktarları belirlenir. Her parametre için güncelleme miktarının bulunması için gradyan değerleri hesaplanır. Bir parametre için gradyan değeri maliyet fonksiyonundan başlayarak geriye zincir kuralı kullanarak türev alınması ile hesaplanır. Kullanılan bu yönteme Gradyan İzdüşüm Algoritması (Gradyan Decent Algorithm - GİA) denir. GİA, her adımda maliyeti minimize edecek şekilde hareket edilmesini sağlar.

Maliyet fonksiyonunun alabileceği değerlere ilişkin örnek Şekil 2.9'da verilmiştir.



Şekil 2.9. Maliyet Fonksiyonu Örneği

Başlangıç durumunda Şekil 2.9'daki grafikte kırmızı noktada başlayan bir öğrenme işlemi için hareket yönü aşağı doğru yani maliyet fonksiyonunu minimize edecek yöne doğru gerçekleşmelidir. GİA, her adımda maliyet fonksiyonunun minimizasyonu hareket yönünü belirler. Bir maliyet fonksiyonu için birden fazla yerelminimum noktası bulunabilir. GİA hata değerini bu yerel minimumlardan birisine götürebilir. Birden fazla minimumu bulunan maliyet fonksiyonu örneği Şekil 2.10'da verilmiştir.



Şekil 2.10. Maliyet Fonksiyonu Örneği

GİA öğrenme işlemi sırasında Şekil 2.10'da belirtilen 2 farklı yolu takip ederek yerel minimumlardan birisine erişebilir. Parametrelerin başlangıç değerlerinin rastgele olarak seçildiği durumda öğrenme işlemi sonucunda ulaşılan minimum hata değeri de değişebilir.

Türev değerlerinin hesaplanması için maliyet fonksiyonunun ilgili parametreye göre türevi alınır. Türev işlemi için zincir kuralından faydalanılır. Bir derin öğrenme ağı üzerinden türevlerin hesaplanması karmaşık olduğu için bu işlem daha basit bir örnek üzerinden açıklanmıştır.

Maliyet fonksiyonumuz (J), (a, b, c) değişkenlerine bağlıdır. Bu değişkenlerin ağdan güncellenebilmesi için maliyet fonksiyonuna göre türevleri $(\frac{\partial J}{\partial a}, \frac{\partial J}{\partial b}, \frac{\partial J}{\partial c})$ alınmalıdır.

GİA ile adım yönü belirlendikten sonra ilgili paremetrelerin güncellenmesi adıma geçilir.

α değeri öğrenme katsayıdır. Bu parametre belirlenen yönde ilerlenecek adım boyutunun belirlenmesini sağlar. Adım boyutu yüksek iken minimum noktaya daha hızlı ulaşılabilir, ancak bazı durumlarda adım miktarı çok büyük seçilirse minimum nokta atlanarak geçilebileceği için minimum noktaya ulaşılamayabilir. Adım miktarının çok küçük seçildiği durumlarda ise öğrenme işlemi yavaş gerçekleşir. Öğrenme katsayısı test aşamasında farklı değerleri kullanılarak birçok kez denenerek belirlenmesi gereken bir hiperparametredir.

GİA güncelleme fonksiyonun bazı dezavantajlarının giderilmesi için birçok farklı güncelleme fonksiyonu geliştirilmiştir. Bu fonksiyonların en çok kullanılanlarından bazıları şunlardır; ADAM, Momentum, AdaDelta, RmsProp, NADAM.

Öğrenme işlemi için herbir parametre için ayrı ayrı türev değerleri hesaplanması gerekmektedir. Bu durum derin ağ yapısına sahip mimariler için çok fazla işlem gücü harcanması anlamına gelmektedir. Derin öğrenme kullanılarak geliştirilen sistemlerde büyük miktarlarda veriler işlenmektedir. Veriler bütün olarak işlenirken bütün parametrelerin hesaplanması sistemde çok fazla işlemci gücü tüketmektedir. GİA üzerine geliştirilen Stokastik GİA bu soruna çözüm amacıyla geliştirilmiş bir yaklaşımdır. Bu yaklaşım ile derin öğrenme yaklaşımlarında veriler bütün halinde işlenmek yerine parçalara ayrılarak işlenmesi tercih edilmektedir. Veri setinin parçalara ayrıldıktan sonra işlenmesi daha az işlemci yükü oluşturmaktadır. Stokastik GİA yaklaşımında minimum değere ulaşmak, standart GİA'ya kıyasla daha fazla adımda gerçekleşebilmektedir. Ayrıca, Stokastik GİA da elde edilen başarı değeri standart GİA'dan daha az da olabilmektedir. Ancak, İşlemci gücünden tasarruf edilmiş olur.

Veri setinin parçalara ayrılması sonrasında oluşan parçalara *veri parçası* denir. Kaç adet veri parçası ile çalışacağının belirlenmesi için Veri Parçası Boyutu belirlenmesi gerekir. Veri Parçası Boyutu (Batch Boyutu ya da Mini-Batch) verinin kaç parçaya ayrılıp işleneceğini gösteren bir hiperparametredir. N adet parçaya ayrılan veride ki tüm parçalar tek tek işlenir. Bir veri parçası işlenmesi bir iterasyonda tamamlanır

Derin öğrenme yaklaşımlarında verisetindeki veri örneklerinin birden fazla kez kullanılması mümkündür. Bu durum performans artışı sağlayabilmektedir. Tüm veriseti üzerinden bir kez geçilmesi işlemine *Epoch* denir. Epoch parametresi eğitim aşamasına başlanmadan önce belirlenmesi gereken bir hiperparametredir.

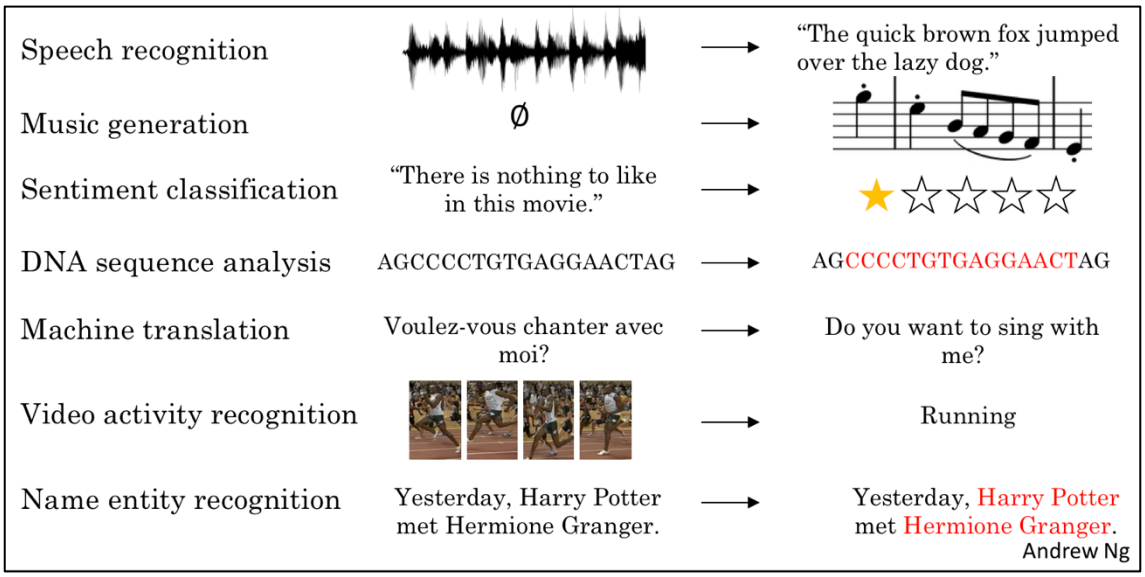
Günümüzde kullanılan Derin Öğrenme araçlarında, kullanıcının sadece ileri yayılım adımını implemente etmesi gerekmektedir. Diğer adımlar geliştirilmiş araçlar sayesinde otomatik bir şekilde ilerletilir. Bu sayede derin öğrenme araştırmacıları ağır matematiksel işlemler içeren geri yayılım adımı ile uğraşmak durumunda kalmazlar.

2.3 Yinelemeli Sinir Ağları (Recurrent Neural Network - YiSA)

İnsanlar bir konu hakkındaki düşünürken düşüncelerine her an sıfırdan başlamazlar. Bir yazıyı okurken ya da filmi seyrederken şimdiki durum geçmişteki gözlemler ile ilişkilidir. Bir metin okurken kelimeleri ayrı ayrı değerlendirmek yerine her kelimeyi o kelimeden önce okuduğumuz kelimelere göre anlamlandırırız. Ancak, bu sayede kelimelere doğru anlamları yükleyip okuduğumuz metni anlayabiliriz. Bu tarz problemlere sekans problemleri denir. Literatürde yer alan bazı sekans problemleri şunlardır;

- Konuşma Tanıma (Speech Recognition)
- Konuşmadan Metne Çevirme (Speech to Text)
- Müzik Üretimi (Music Generation)
- Metinler üzerinde duygu sınıflandırması (Sentiment Classification)
- DNA analizi (DNA sequence analysis)
- Makine Çevirileri (Machine Translation)
- Video AktiviteTanıma (Video Activity Recognition)
- İsimlendirilmiş Varlıkların Tanınması (Named Entitiy Recognition)

Bu problemlere ilişkin örnekler Şekil 2.11'de verilmiştir.

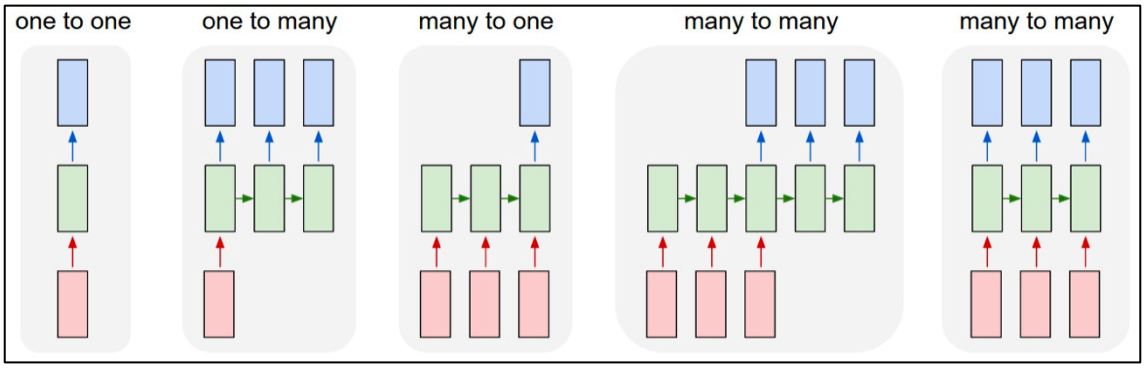


Şekil 2.11. Sekans Problemi Örnekleri [53]

Sekans problemleri kendi içerisinde dörde ayrılır;

- Birden bire (one to one)
- Birden çoğa (one to many)
- Çoktan Bire (many to one)
- Çoktan çoğa (many to many)

Farklı sekans problemlerinin temsili Şekil 2.12'de verilmiştir.



Şekil 2.12. Sekans Problemi Türleri [54]

Çoktan çoğa problemleri girdi saysının çıktı sayısına eşit ya da farklı olması şeklinde iki farklı şekilde tanımlanabilir.

Geleneksel yapay sinir ağı modelleri şimdiki durumu geçmiş gözlemlerle birlikte analiz etmekte yetersiz kalmaktadır.

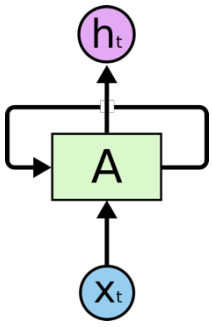
Sekans problemlerinde geleneksel derin öğrenme yaklaşımları yerine önerilen yaklaşımlardan birisi Yinelemeli Sinir Ağı mimarileridir.

Geleneksel derin öğrenme yaklaşımlarının sekans problemlerinde yetersiz kalmasının başlıca nedenleri şunlardır;

- Sekans modellerde girdi ve çıktı boyutları farklı veri örnekleri için farklı olabilmektedir.
- Metnin farklı konumlarında öğrenilen özellikler kelimeler arasında paylaşılamamaktadır.

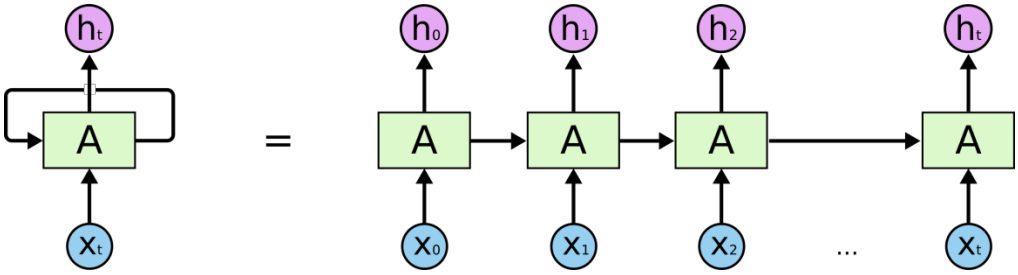
YiSA'lar, bize geçmiş durumlarda elde edilen bilgilerin *t* anında kullanılması yeteneğine sahip modeller üretmemize olanak tanımaktadır. Bu derin öğrenme yaklaşımında,

$t$durumunun değerlendirilmesi için t anından önceki durumlar t anındaki durum için girdi olarak verilir. YiSA mimarilerine ilişkin yapı Şekil 2.13'te verilmiştir.



Şekil 2.13. Yinelemeli Sinir Ağı Yapısı [55]

Şekil 2.13'te verilen diyagramda, sinir ağının bir hücresi (A), x_t ve x_{t-1}girdilerinin sonucunda bir h_t değeri üretir. Bu döngü, bilginin ağın bir adımından diğerine geçmesine olanak tanır. Şekil 2.13'te verilen döngü gösterimi ağ yapısının anlaşılmasını zorlaştırabilmektedir. Bu nedenle literatürde bu döngü yapısının açık haline de rastlanmaktadır. Şekil 2.13'teki döngü yapısının açılmış hali Şekil 2.14'te verilmiştir.



Şekil 2.14. Yinelemeli Sinir Ağları için Döngü Yapısının Açılmış Gösterimi [55]

Bu mimari sekans ve liste şeklindeki veri yapılarının işlenmesi için sıklıkla tercih edilmektedir.

Bu mimarinin açıklanmasında kullanılan notasyon şu şekildedir;

- a: Aktivasyon değeri
- w: Ağırlık
- b: Bias
- x: Veri Örneği
- y: Gerçek Etiket

- $\tilde{y}$: Analiz edilen x girdisine karşılık tahmin edilen y değeri
- Üstindis (superscript) de yer alan $[\ l\ ]$üzerinde bulunduğu objenin hangi katmanda yer aldığını gösterir.

  $a^{[4]}$: Dördüncü Katmandaki aktivasyon değerlerini,

  $w^{[5]}$: Beşinci Katmandaki ağırlık değerlerini

  $b^{[3]}$: Üçüncü Katmandaki bias değerlerini göstermektedir.
- Üstindis (i)objenin sırasını belirtir.

  $x^{(3)}$: Üçüncü eğitim örneğini temsil etmektedir.
- Üstindis $\langle t \rangle$ objenin hangi zaman adımında olduğunu gösterir.

  $x^{(i)\langle t \rangle}$: i'inci eğitim örneğinin t zaman adımındaki durumunu temsil etmektedir.
- Altindis i bir vektörün belirtilen sıradaki girdisini temsil eder.

  $a_i^{[l]}$: l katmandaki aktivasyon değerlerinin i sıradaki girdisini belirtir.

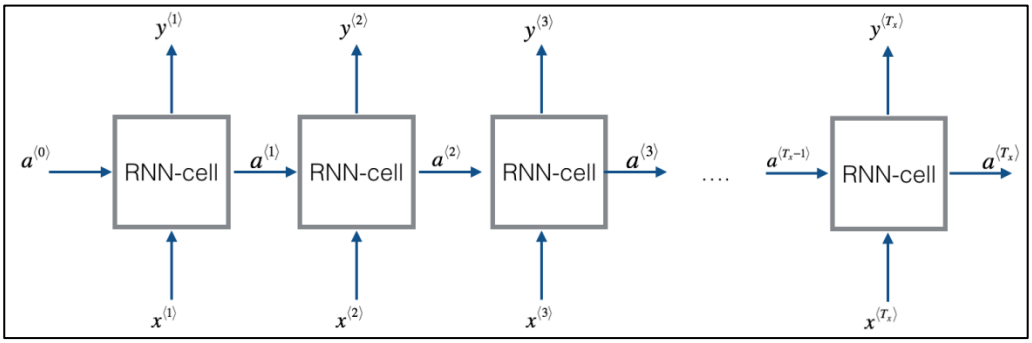
  T_x: veri örneği boyutu (veri örneği sayısı)

  T_y: çıktı boyutu

  a_0:Başlangıç aktivasyon değeri (rastgele değerler ile ilklendirilebilir)

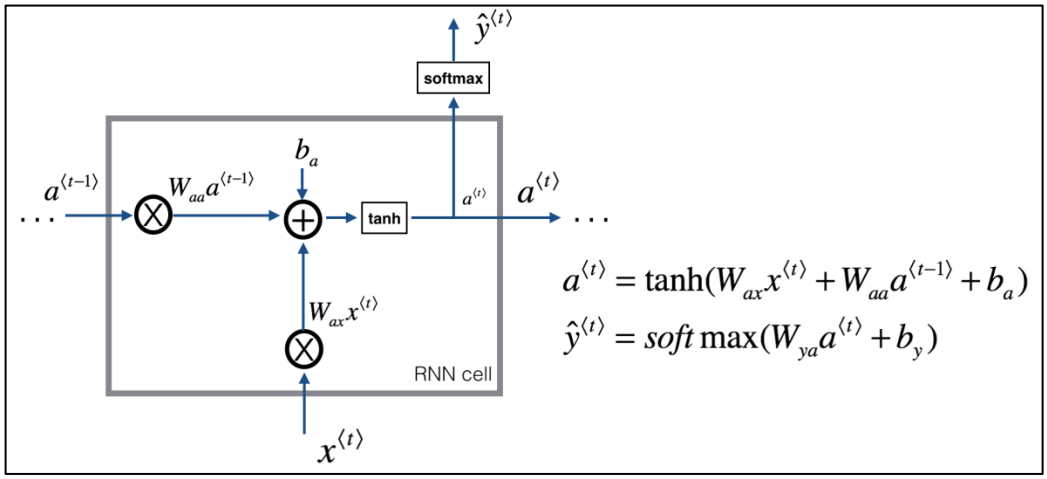
2.3.1 Temel YiSA Hücresi

T_x= T_y olan bir problem için notasyon gösterimlerinin YiSAmimarisinde kullanımı Şekil 2.15'de verilmiştir.



Şekil 2.15. YiSA Mimarisi Notasyon ile Gösterim [53]

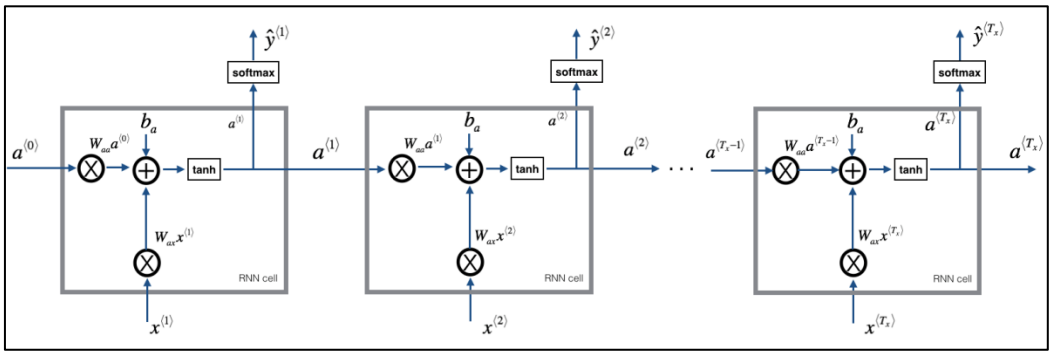
YiSA'lar, tek bir hücrenin tekrarı olarak görülebilir. İlk adımda tek bir zaman adımı için YiSA hücresinde bulunan hesaplamalar uygulanır. Ardından artan t adımları ile bu işlemler tekrarlarnır. Şekil 2.16, bir YiSA hücresinin tek bir zaman adımı için uygulanan işlemleri göstermektedir.



Şekil 2.16. Temel YiSA Hücresi [53]

2.3.1.1 **Temel YiSA Hücresi ile İleri Yayılım**

Temel YiSA hücresi, $x^{\langle t \rangle}$: geçerli giriş ve $a^{\langle t-1 \rangle}$: geçmişten gelen bilgileri içeren önceki gizli durumu girdi olarak alır. Hesaplanan$a^{\langle t \rangle}$ değeri bir sonraki YiSA hücresine verilir. Ayrıca $a^{\langle t \rangle}$, $\tilde{y}^{\langle t \rangle}$: tahmin değeri hesaplanması için de kullanılmaktadır. Birbirini tekrarlayan YiSA hücreleri ile oluşturulan YiSA mimarisine ilişkin örnek Şekil 2.17'de verilmiştir.



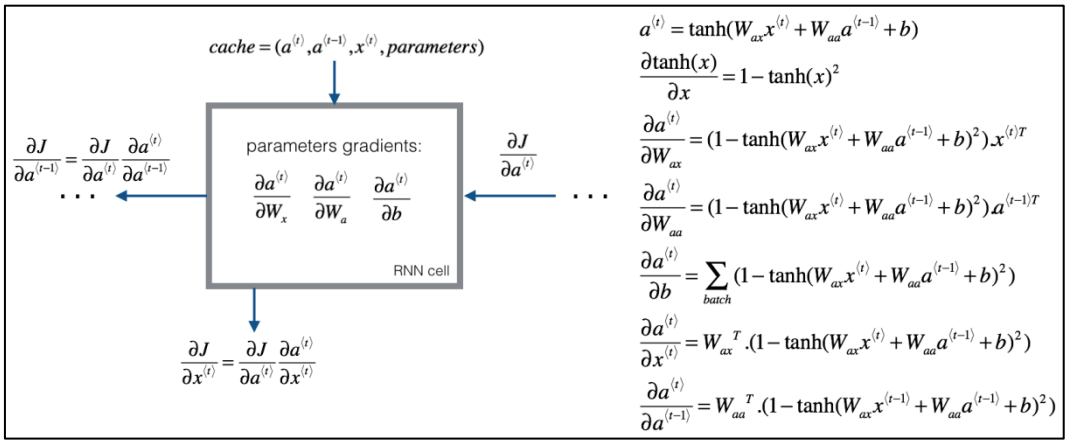
Şekil 2.17. YiSA Hücre Yapısı Detaylı Gösterimi [56]

Girdi sekansı boyutu 10 ise 10 adet YiSA hücresinin kopyalanarak yan yana yerleştirilmesi ile YiSA mimarisi tasarlanır. Her hücre, bir önceki hücrede hesaplanan

$a^{\langle t-1 \rangle}$ ve geçerli durum için sekans objesini $x^{\langle t \rangle}$ girdi olarak alır. Bu bilgiler kullanılarak Şekil 2.18'de verilen formüller kullanılarak $a^{\langle t \rangle}$ ve $\tilde{y}^{\langle t \rangle}$ değerleri hesaplanır. T_x= T_y olan bir problem için, girdi $x = (x^{\langle 1 \rangle}, x^{\langle 2 \rangle} \dots x^{\langle T_x \rangle})$ ve $y = (y^{\langle 1 \rangle}, y^{\langle 2 \rangle} \dots y^{\langle T_y \rangle})$ sayıları birbirine eşittir. Bu hesaplamalar öğrenme işleminin ileri yayılım kısmını oluştururlar.

2.3.1.2 **Temel YiSA Hücresi ile Geri Yayılım**

Derin öğrenme yaklaşımlarında öğrenme işleminin tamamlanması için parametrelerin güncellenmesinin tamamlanması gerekmektedir. Parametrelerin güncellenebilmesi için İleri yayılım adımı ardından da Geri Yayılım adımının uygulanması gerekir. Geri yayılım adımının uygulanmasının ardından hesaplanan gradyanlar kullanılarak ilgili parametreler güncellenir. Geri yayılım adımına ilişkin detaylı bilgiler Bölüm 2.2.2'de verilmiştir. Temel YiSA hücresi ile tasarlanan mimaride geri yayılıma ilişkin görsel ve formüller Şekil 2.18'de verilmiştir.



Şekil 2.18. Temel YiSA Hücresi ile Geri Yayılım [53]

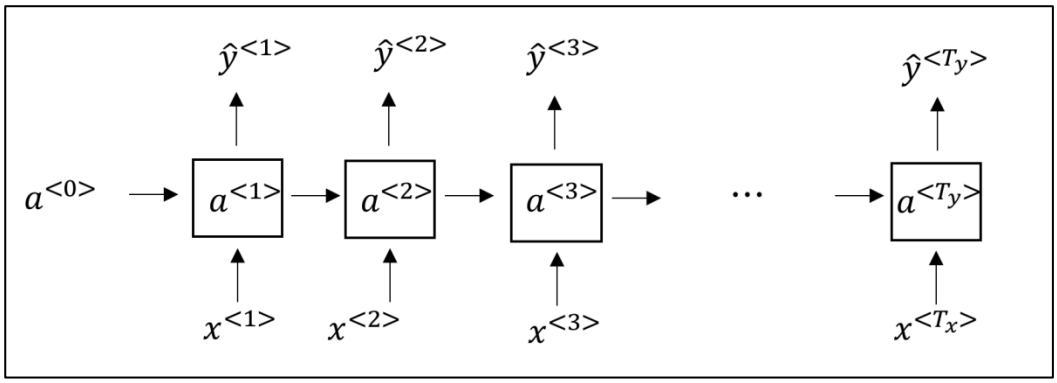
YiSA hücresinde geriye yayılım adımı geleneksel derin öğrenme yaklaşımlarında olduğu gibi maliyet fonksiyonun (J) ilgili parametreye göre zincir kuralı uygulanarak türevi alınması şeklinde gerçekleşir. Zincir kuralı, $\left(\frac{\partial J}{\partial W_{ax}}, \frac{\partial J}{\partial W_{aa}}, \frac{\partial J}{\partial W_b}\right)$'nin hesaplanması ve (W_{ax}, W_{aa}, b_a)'nın güncellenmesi için kullanılır. YiSA mimarisinde ileri yayılımda bir hücre çıktısı olarak $a^{\langle t \rangle}$ ve $\tilde{y}^{\langle t \rangle}$ olmak üzere iki farklı değer hesaplanır. Bu nedenle iki farklı ağırlık matrisine$(W_{ax}, W_{aa},)$ ihtiyaç vardır.

- W_{aa}: Aktivatsyon değerleri (a) ile çarpılan ağırlık matrisi

- W_{ax}: Veri örnekleri (x) ile çarpılan ağırlık matrisi
- b_a: Bias değeri

2.3.2 Yok Olan Gradyan Problemi (Vanishing Gradient Problem)

Şekil 2.19'da verildiği gibi YiSA mimarileri ard arda birçok kez tekrarlanan hücrelerden oluşur. Bir kelime anlamlandırılmaya çalışırken ilgili kelimeden önce geçen kelimelerin bilgilerinden de faydalanılır.



Şekil 2.19. YiSA Mimarisi

Ancak bazı durumlarda aktarılan bilgiler önemli olmasına rağmen ağ mimarisi içerisinden etkisi azalarak ilerlediği için gereken etkiyi oluşturamamaktadır. Her kelimeye ilişkin bilgi bir sonraki hücreye bir miktar azalarak etki eder. Birinci kelimeden elde edilen bilginin ikinci kelimeye etkisi yüksek iken onuncu kelimeye olan etkisi çok daha azdır. Ancak, aralarında uzun mesafe bulunan iki kelime anlamsal olarak birbirleri ile ilişkili olabilmektedir. Örneğin, 2 adet sekansımız olsun;

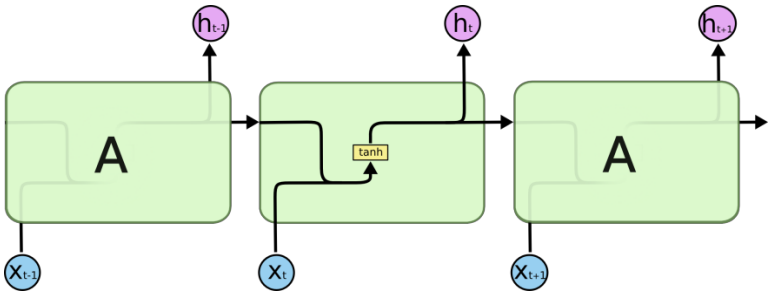
- The cat, which already ate ....., was full.
- The cats, which ....., were full.

`Cat` ve `Cats` kelimeler kendilerinden çok sonra geçen `was` ve `were` kelimeleri ile ilişkilidir. Ancak, bu kelimeler arasında mesafe uzun olduğu için Temel YiSA Hücresi yapısı ile tasarlanmış YiSA mimarisinde bu kelimeler arasında ilişkilerin çıkarılması ve kullanılması oldukça zordur. Bu nedenle uzun sekanslara sahip problemler için temel YiSA hücrelerinin kullanılması çok efektif değildir. Kelimelerin bu şekilde gittikçe azalarak yok olan etkisine Yok Olan Gradyan Problemi denir. Bu durum birçok sekans problemi için çözülmesi gereken ciddi bir problemdir. Bu problemin çözülebilmesi için

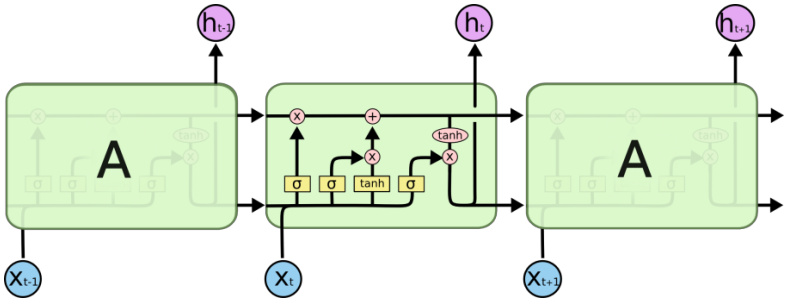
bazı yaklaşımlar önerilmiştir. Bunlardan en çok bilinenlerinden birisi Long Short Term Memory (LSTM) Hücre yapılarıdır.

2.3.3 LSTM Hücreleri ile YiSA

LSTM yapıları temel YiSA hücrelerin bazı iyileştirmeler ile güçlendirilmiş versiyonudur. Bu yaklaşım [57] 1997 yılında Hochreiter ve Schmidhuber tarafından önerilmiştir. LSTM hücreleri ile tasarlanan ağ yapıları temel YiSA yapısı ile geliştirilmiş ağ mimarisi ile aynı tekrarlayan yapıya sahiptir. LSTM yapısı, temel YiSA hücrelerine eklenen bazı kapılar ile YiSA hücrelerine Uzun Dönem Hafıza (Long-term memory) özelliği kazandırmıştır. Temel YiSA yapısına ilişkin örnek Şekil 2.20'de ve LSTM hücresi ile tasarlanan yapıya ilişkin örnek ise Şekil 2.21'de verilmiştir.



Şekil 2.20. Temel YiSA ile Tasarlanan Mimari [55]



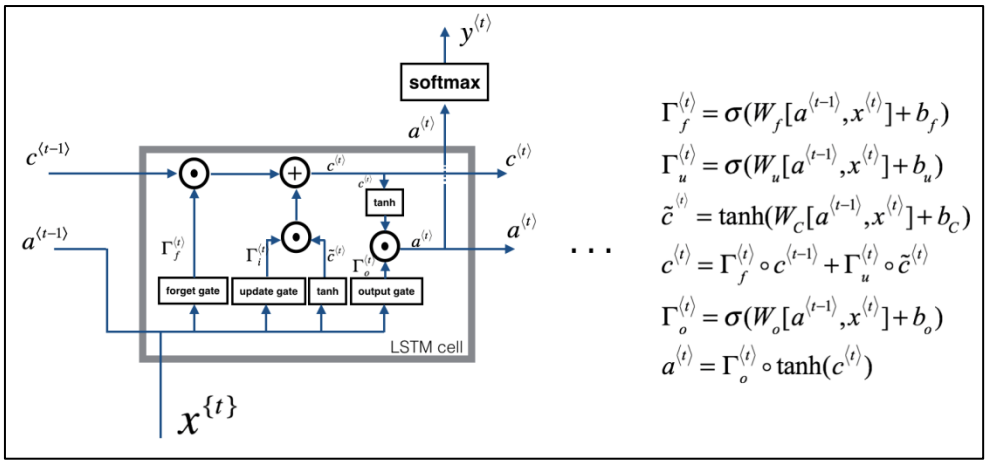
Şekil 2.21. LSTM Hücreleri ile Tasarlanan Mimari [55]

LSTM hücresi her adımında hücre durumunu / hafıza değişkenini ($c^{\langle t \rangle}$) takip eder ve günceller. Bu işlem temel YiSA hücresinde olduğu gibi T_x adet tekrarlanır. LSTM hücresine eklenen üç kapı şunlardır;

- **Forget Kapısı** -W_f: Forget kapısı, geçmiş bilginin ($c^{\langle t-1 \rangle}$) şimdiki duruma ($c^{\langle t \rangle}$) ne kadar etki edeceğini kontrol eden kapıdır. Bu kapıda $\Gamma_f^{\langle t \rangle}$ değeri hesaplanır.

- **Update Kapısı** -W_u: Yeni bilgiye ne kadar ekleme yapılacağını kontrol eden kapıdır. Bu kapıda $\Gamma_u^{\langle t \rangle}, \tilde{c}^{\langle t \rangle}, c^{\langle t \rangle}$ değerleri hesaplanır.
- **Output Kapısı** -W_o: Şu anki bilginin $(c^{\langle t \rangle})$ ne kadarının çıktıya $(a^{\langle t \rangle})$ aktarılacağını kontrol eder. Bu kapıda $\Gamma_o^{\langle t \rangle}, a^{\langle t \rangle}$ değerleri hesaplanır.

LSTM hücresine ilişkin detaylı gösterim ve kapılarda hesaplanan formüller Şekil 2.22'de verilmiştir.



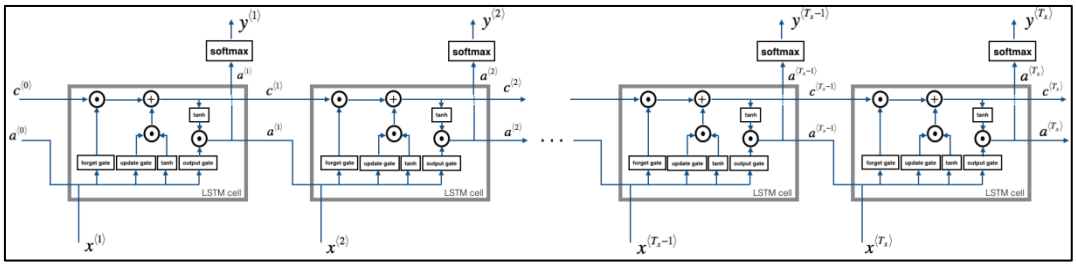
Şekil 2.22. LSTM Hücresi Detaylı İçerik [53]

LSTM hücresinde her kapıdan akan parametler farklıdır. Bu nedenle bu kapılar için farklı ağırlık matrislerine ihtiyaç vardır.

- W_f: Forget kapısında öğrenilen ağırlık matrisi
- W_u: Update kapısında öğrenilen ağırlık matrisidir.
- W_o: Output kapısından öğrenilen ağırlık matrisidir.
- W_c: Sonraki hücreye aktarılan hafıza bilgisinin öğrenildiği ağırlık matrisidir.
- b_f: forget kapısında kullanılan bias değerleri
- b_u: Update kapısında kullanılan bias değerleri
- b_o: Output kapısında kullanılan bias değerleri
- b_c: Sonraki hücreye aktarılan bilgi için kullanılan bias değerleri
- $c^{\langle t \rangle}$: sonraki hücreye aktarılan hafıza bilgisi

2.3.3.1 **LSTM Hücresi ile İleri Yayılım**

LSTM hücreleri zaman adımı sayısında ard arda eklenerek YiSA mimarisini oluşturur. Her zaman adımında hesaplanan hafıza bilgisi $c^{\langle t \rangle}$ ve aktivasyon verisi $a^{\langle t \rangle}$bir sonraki zaman adımına aktarılır. Bu işlem T_x adımı kadar tekrarlanır. İşleyiş mekanizması Şekil 2.23'de verilmiştir.



Şekil 2.23. Birçok Zaman Adımına Uygulanan LSTM Yapısı [53]

2.3.3.2 **LSTM Hücresi ile Geri Yayılım**

LSTM hüceresi ile tasarlanmış mimarinin geri yayılımı için kapı değişkenlerinin türevleri hesaplanır.

db_f, db_u, db_c, db_o değerleri ise sırasıyla $d\Gamma_f^{\langle t \rangle}$, $d\Gamma_u^{\langle t \rangle}$, $d\Gamma_c^{\langle t \rangle}$, $d\Gamma_o^{\langle t \rangle}$ değerlerinin yatay eksende toplanması ile hesaplanır.

Son olarak önceki aktivasyon durumuna (da_{prev}), önceki memory durumu (dc_{prev}) ve girdi $(dx^{\langle t \rangle})$ türevleri hesaplanır.

2.4 Kelime Gösterimleri (Word Representation – Word Embeddings)

Birçok dilde bir kelime, cümlede kullanımına göre farklı anlamlar kazanabilir. Bu nedenle DDİ uygulamalarında anlamsal bilginin çıkarılması metinlerin işlenmesinde oldukça önemlidir.

Kelimelerin DDİ yöntemleri ile işlenebilmesi için algoritmalar tarafından işlenebilir bir formata çevrilmeleri gerekmektedir. Kelimelerin işlenebilir formattaki hallerine Kelime Gösterimi denir.

Kelime Gösterimleri literatürde iki farklı başlık altında değerlendirilmektedir. Bunlar;

- Frekans Tabanlı Kelime Gösterimi (FTKG)

- Tahminleme Tabanlı Kelime Gösterimi (TTKG)

Frekans tabanlı kelime gösterim yöntemlerine (Count Vector), Tf-Idf yöntemleri örnek verilebilir. Bu kelime gösterim yöntemleri geleneksel makine öğrenmesi yöntemlerinde sıklıkla kullanılmaktadır.

Yapay Sinir Ağları ile kelime gösterimi işlemi oldukça sınırlı bir şekilde gerçekleştirilirken, derin öğrenme yaklaşımlarında kullanılmak üzere sinir ağı tabanlı yöntemler geliştirilmiştir. Bu yöntemlerden en bilinenleri Word2Vec [58] ve GloVe [59] yöntemleridir.

TTKG yaklaşımları ile kelimelerin anlamsal ilişkileri çıkarılabilmektedir. Örneğin; kral ve kraliçe kelimeleri arasındaki anlamsal ilişki erkek ile kadın arasında da vardır. Dolayısıyle, kral, kraliçe kelime vektörleri arasındaki mesafe erkek, kadın kelime vektörleri arasındaki mesafeye benzemektedir. Kelime vektörleri arasındaki mesafeler Kosinüs Benzerliği (Cosine Similarity) ile hesaplanır.

TTKG yaklaşımları ile kelime vektörlerinin öğrenilmesi eğitimsiz bir makine öğrenmesi problemidir (unsupervised problem). Bu sayede çok büyük miktarlarda metinsel veri işlenerek başarılı kelime vektörleri çıkarılabilir.

TTKG yaklaşımları ile her kelime için belirli sayıda özellik öğrenilir. Öğrenilen her özellik bir miktar anlamsal bilgi taşımaktadır. TTKG yaklaşımları ile öğrenilen kelime vektörlerine ilişkin örnek Çizelge 2.3'te verilmiştir.

Çizelge 2.3. Kelime Gösterimi Örnekleri

	Erkek	**Kadın**	**Kral**	**Kraliçe**	**Elma**	**Portakal**
Cinsiyet	-1.0	1.0	-0.95	0.97	0.0	0.01
Kraliyet	0.01	0.02	0.93	0.95	-0.01	0.0
Çağ	0.03	0.02	0.7	0.69	0.03	-0.02
Yemek	0.04	0.01	0.02	0.01	0.95	0.97

Çizelge 2.3'te verilen örnekte 4 farklı özelliğe ilişkin öğrenilmiş kelime vektörleri gösterilmiştir. Büyük metinler üzerinden yapılan öğrenme işlemi sonrasında verilen kelimelere ilişkin özellik değerleri hesaplanır. Verilen örnekteki özellikler cinsiyet, kraliyet, çağ ve yemektir. Her kelime için bu özellikler öğrenildiğinde "*erkek"* kelimesinin cinsiyeti -1, "*kadın*" kelimesinin cinsiyeti ise 1 olarak hesaplanmıştır. Cinsiyet özelliği bu iki kelimenin anlamlandırılmasında oldukça ayırt edici olduğu için bu değerler birbirlerinden çok farklı çıkmıştır. Ancak, erkek ve kadın kelimesi için "*yemek*" özelliği ayırt edici olmadığı için bu değerleri birbirlerine yakın olarak hesaplanmıştır. Benzer şekilde "*kral"* ve "*kraliçe"* kelimeleri için hesaplanan cinsiyet özelliği değerleri birbirinden çok farklı çıkmıştır. "*Kral*" kelimesi için hesaplanan cinsiyet değeri -0.95 ve "*kraliçe*" kelimesi için hesaplanan cinsiyet özelliği ise 0.97'dir. Bu değerleri *erkek* ve "*kadın"* kelimelerinden öğrenilen değerler ile paralellik göstermektedir. "*Erkek"* kelimesinin cinsiyet özelliği -1 iken, "*kral*" kelimesinin cinsiyet özelliği -0.95 olarak hesaplanmıştır. Bunların yanı sıra "*kral*" ve "*kraliçe"* kelimeleri için "*çağ*" özelliği ve "*kraliyet*" özelliği değerleri birbirlerine yakın olarak hesaplanmıştır. Bu değerler bu iki kelimenin "*çağ*" ve "*kraliyet*" özellikleri düşünüldüğünde birbirlerine benzer olduklarını göstermektedir.

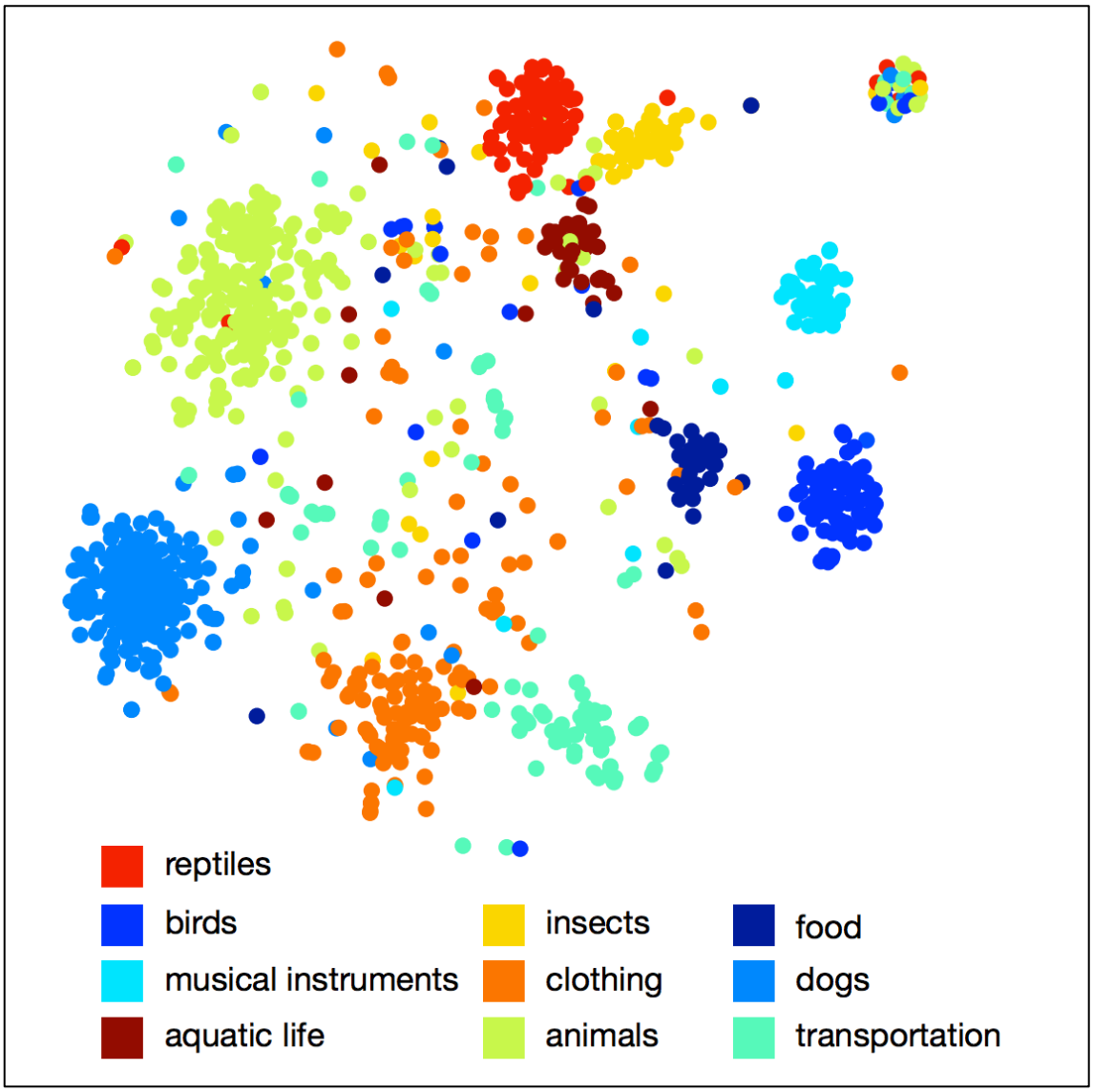
"*Elma*" ve "*portakal*" kelimelerinin ise "*yemek*" özelliği değerleri birbirlerine yakın olarak hesaplanmıştır. Bu iki kelime için "*yemek*" özelliği ayırt edici bir özellik iken, diğer özellikler bu kelimelerin anlamlandırılmasında çok değer taşımadıklarından diğer özelliklerin değerleri 0 a yakın olarak hesaplanmıştır.

Derin öğrenme yaklaşımlarında her kelime, hesaplanan vektör gösterimi kullanılarak işlenir. TTKG yaklaşımları ile hesaplanan vektörlerin boyutu, vektör çıkarımı öncesinde belirlenmesi gereken bir hiperparametredir. TTKG tabanlı kelime gösterimi öğrenimi algoritmaları, hiperparametre olarak belirlenen n sayıda özellik için öğrenme işlemi gerçekleştirir.

Sistemde öğrenilen özellikler, sistem tarafından belirlenen anlamsal ilişkileri temsil eder. Bu özellikler tek başlarına insanlar tarafından anlamlı değildir. Çizelge 2.3'te verilen özellikler örnek olarak anlatılmıştır. TTKG ile hesaplanan kelime vektörlerinde

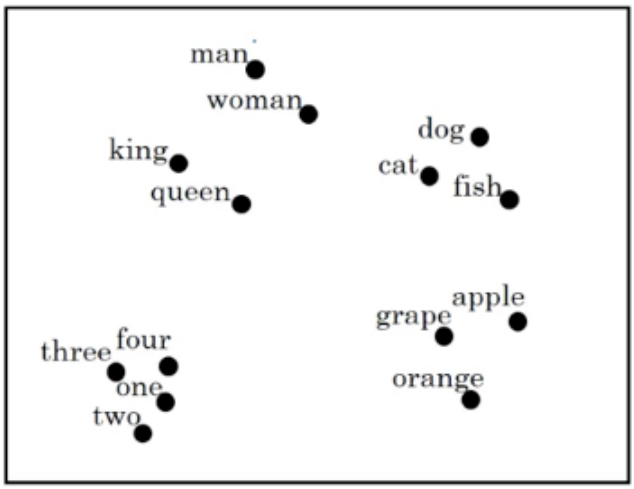
özelliklere Çizelge 2.3'te verildiği gibi insanlar tarafından belirli anlamlar yüklenememektedir.

Öğrenilen kelime vektörleri t-SNE yöntemi [60] kullanılarak görselleştirildiğinde kelimelerin anlamsal ilişkilerine göre gruplar oluşturduğu gözlenmektedir. t-SNE bir boyut azaltma yaklaşımıdır. Bu yaklaşım çok boyutlu verilerin görselleştirilmesinde yaygın olarak kullanılmaktadır. Bir kelime uzayında yer alan kelime vektörlerinin tamamının t-SNE ile görselleştirilmiş hali Şekil 2.24'te verilmiştir. Kelime vektörlerine t-SNE yöntemi ile boyut azaltma uygulandığında kelimelerin anlamsal birlikteliklerine göre gruplandığı gözlenmektedir. Örneğin, yemek ile ilgili kelimeler bir araya toplanmakta, hayvan isimleri bir araya toplanmaktadır.



Şekil 2.24. Örnek Bir Kelime Uzayının t-SNE ile Gösterimi [61]

t-SNE ile görselleştirme işlemi sonrası oratya çıkan kelime gruplarına ilişkin örnek Şekil 2.25'de verilmiştir.



Şekil 2.25. t-SNE ile Kelime Vektörlerinin Görselleştirilmesi [62]

Kelimeler arasında anlamsal ilişkiler kelime vektörleri kullanılarak çıkarılabilmektedir [60]. TTKGile çıkarılan kelime vektörlerine ilişkin örnek Çizelge 2.4'te verilmiştir. Çizelgede her kelime için vektörler yer almaktadır. Kelime vektörleri V_{kelime} notasyonu ile gösterilecektir.

Çizelge 2.4. Kelime Vektörleri

	Erkek V_{erkek}	**Kadın** $V_{kadın}$	**Kral** V_{kral}	**Kraliçe** $V_{kraliçe}$	**Elma** V_{elma}	**Portakal** $V_{portakal}$
Cinsiyet	-1.0	1.0	-0.95	0.97	0.0	0.01
Kraliyet	0.01	0.02	0.93	0.95	-0.01	0.0
Çağ	0.03	0.02	0.7	0.69	0.03	-0.02
Yemek	0.04	0.01	0.02	0.01	0.95	0.97

Örneğin; "Erkek ile Kadın arasındaki ilişkiye karşılık Kral kelimesine karşılık gelen kelime hangisidir?"

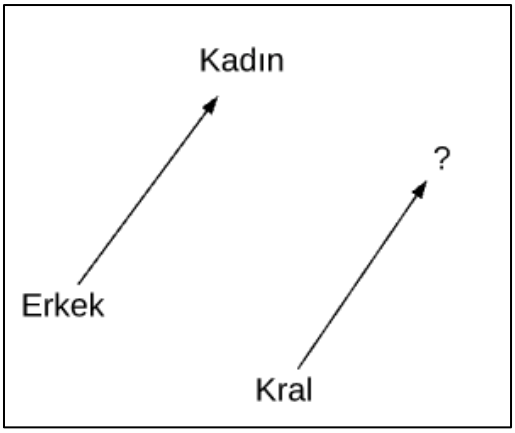
- Erkek →Kadın | Kral → ?

Bu soru kelime vektörleri kullanılarak çözülebilir. Öncelikle erkek kelimesi ile kadın kelimesi arasındaki vektörel uzaklık hesaplanır. (Kelimeleri temsil eden kelime vektörleri Çizelge 2.5'te verilmiştir.)

Erkek ve Kadın kelime vektörlerinin arasında ki fark değeri ile Kral ve aranan kelime arasındaki vektörel farkın benzer olması gerekir. Bu hesaplamaya ilişkin formül Denlem 2.4'de verilmiştir.

$$V_{erkek} - V_{kadın} \cong V_{kral} - V_{?} \tag{2.4}$$

Bu durumun vektörel gösterimi Şekil 2.26'da verilmiştir.



Şekil 2.26. Kelime Benzerliklerinin t-SNE ile Gösterimi

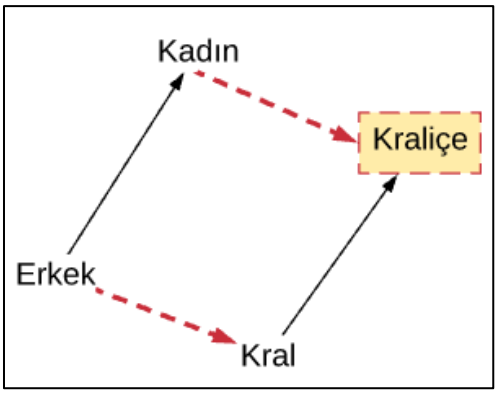
Aranan kelimenin bulunabilmesi için kelime benzerlikleri denkleminde aranan kelime yalnız bırakılır. Denklem 2.5 üzerinde bu işlem uygulandığında Denklem 2.9 elde edilir.

$$V_{?} \cong V_{kral} - V_{erkek} + V_{kadın} \tag{2.5}$$

Bu tarz vektörel hesaplamalarda aranan kelime vektörünün hesaplama sonucunun birebir elde edilmesi zordur. Bu nedenle karşılaştırma işlemi için benzerlik fonksiyonlarından faydalanılır. Vektörel benzerliğin bulunması için yapılacak işlem Denklem 2.6'da tanımlanmıştır.

$$kelime = argmax_{kelime(k)}\,(benzerlik(V_{?}\,,\; V_{kral} - V_{erkek} + V_{kadın})) \tag{2.6}$$

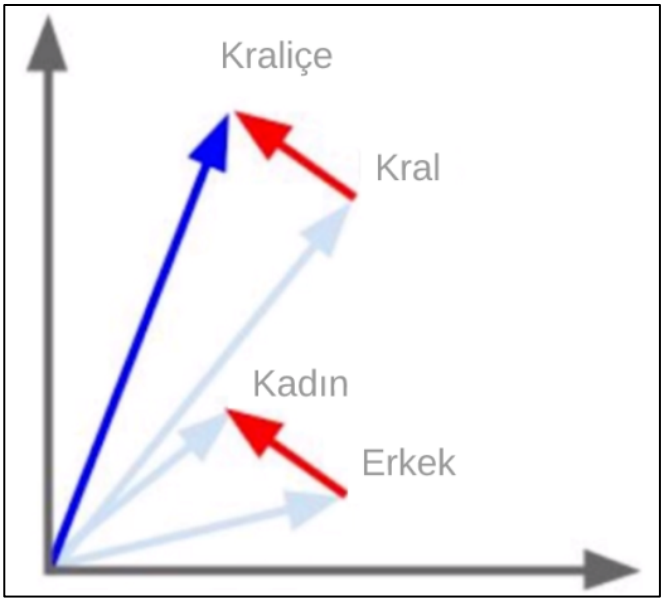
Kelime uzayındaki her kelime vektörü için benzerlik fonksiyonu çalıştırılığında benzerlik değeri $V_{kral} - V_{erkek} + V_{kadın}$ vektörüne en yakın kelime aranır. Aranan kelime (Kraliçe) için yapılan işlemlerin t-SNE ile gösterimi Şekil 2.27'de verilmiştir.



Şekil 2.27. Benzerlik Hesabı Sonucu Bulunan Kelime

İki vektörel değişken arasında benzerliğin hesaplanması için farklı benzerlik fonksiyonları kullanılabilir. Bunlardan birisi Kosinüs Benzerliği fonksiyonudur.

Kosinüs Benzerliği fonksiyonu ile anlamsal bilginin çıkarılmasına ilişkin görsel Şekil 2.28'de verilmiştir.



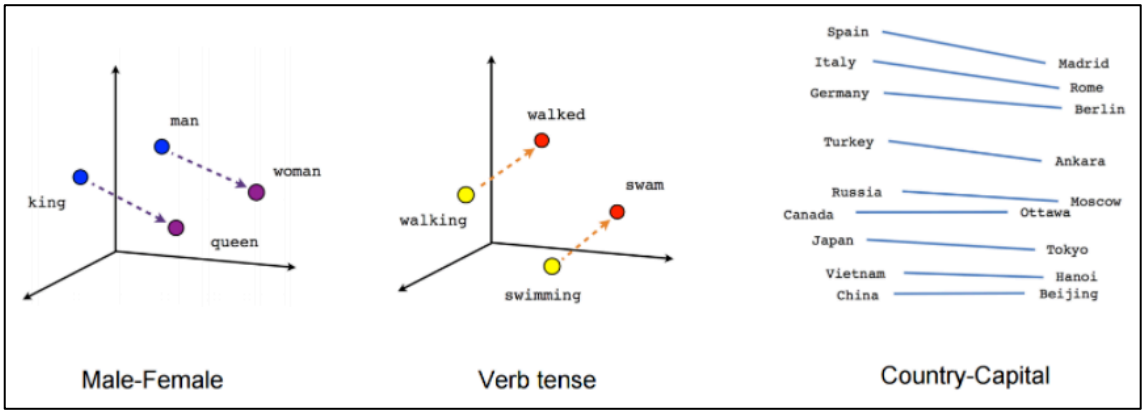
Şekil 2.28. Cosine Similarity ile Anlamsal Bilginin Çıkarılması

Bu yaklaşım kullanılarak elde edilebilecek anlamsal kelime birlikteliklerine ilişkin örnekler Çizelge 2.5'te verilmiştir. Çizelge 2.5'te verilen örneklerde, verilen bir kaynak kelime ikilisine karşılık hedef kelime ikilisinden sadece birisi verildiğinde diğer kelimenin tahmin edilmesi işlemi yapılmaktadır.

Çizelge 2.5. Kelime Vektörleri ile Anlamsal Birlikteliklerin Çıkarılması Örnekleri

Kaynak Kelime İkilisi	**Hedef Kelime İkilisi**
Erkek: Kadın	Abi: Abla
Ankara: Türkiye	Tokyo: Japonya
Türk Lirası: Türkiye	Yen: Japonya

TTKG yaklaşımları ile kelimeler arasında anlamsal ilişkilerin çıkarılmasına ilişkin İngilizce dili için örnekler Şekil 2.29'da verilmiştir.



Şekil 2.29. 3 Boyutlu Uzayda Kelime Vektörleri Örneği [63]

Anlamsal ilişkilerin kelime vektörlerinden başarılı bir şekilde çıkarılabilmeleri için kelime vektörlerinin başarılı bir şekilde elde edilmesi gerekmektedir.

DDİ problemlerinde belirli kelime uzayında işlem yapılır. Kelime uzayında bulunan bütün kelimelerin yer aldığı yapıya sözlük (dictionary) denir. Sözlük içerisinde kelimeler liste şeklinde yer alır. Sözlük içerisinde bulunmayan kelimelerin temsil edilebilmesi için sözlüğe özel bir token (UNK) eklenir. Bir sözlük örneği şu şekildedir. Kelimeler sözlük içerisinde bulunma sıraları ile temsil edilir.

$[a(k_0) \quad adam(k_1) \quad elma(k_2) \quad \ldots \quad portakal(k_{6257}) \quad \ldots \quad zebra(k_{9999}) \quad UNK]$

Problem çözümünde yalnızca bu sözlük içerisinde bulunan kelimeler analiz edilebilir. Sözlük sisteme dışardan elle tanıtılabileceği gibi eğitim aşamasında kullanılan metinler içerisinde bulunan farklı kelimelerin çıkarılması yoluyla da elde edilebilir.

Algoritma içerisinde gerçekleştirilecek işlemlerde her kelime için kelime vektörleri kullanılır. Örnek bir kelime vektörü Çizelge 2.6'da verilmiştir.

Çizelge 2.6. Kelime Vektörü Örneği

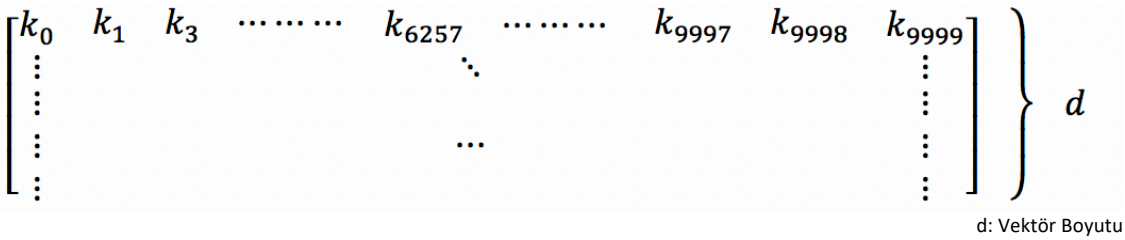
Erkek V_{erkek}
-1.0
0.01
0.03
0.04

Kelimeler, ihtiyaç duyulduğunda sözlükteki ve Vektör matrisindeki karşılığının bulunabilmesi için one-hat gösterimi ile saklanırlar. Örneğin içerisinde 10000 adet kelime bulunan bir sözlüğümüz bulunsun. Her kelime için 10000 boyutunda bir vektör açılır. Bu vektör içerisinde sadece kelimenin sözlükte ki sırasını gösteren değer 1, diğer değerler 0 olarak atanır. Örnek one-hat gösterimleri Şekil 2.30'da verilmiştir.

a	adam		zebra	Zil
1	0		0	0
0	1		0	0
0	0		0	0
. . .	. . .	.....	. . .	. . .
0	0		1	0
0	0		0	1

Şekil 2.30. Örnek One-hat Gösterimleri

Sözlükte bulunan her kelimenin vektörlerinin saklandığı yapıya Vektör Matrisi (Embedding Matrix) denir. Vektör Matrisi yapısı Şekil 2.31'de verilmiştir.



Şekil 2.31. Vektör Matrisi Yapısı

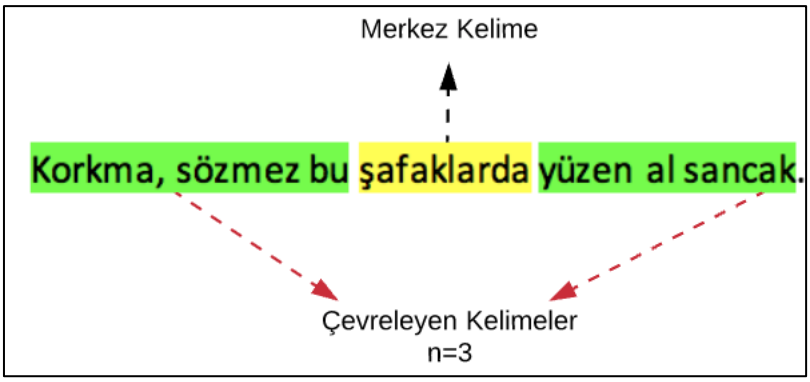
Vektör Matrislerinde bütün kelimelerin vektörleri bulunur. Her kelime vektörü bir sütunda yer alır. Kelime Vektörü boyutu d olan ve sözlükte kulunan kelime sayısı m olan bir örnek için Vektör Matrisi boyutu $(d\ x\ m)$'dir.

Vektör Matrislerinin öğrenilmesi için kullanılan yöntemlerden en bilinenleri Word2Vec [58] ve GloVe [59]'dır. GloVe, Word2Vec yönteminin üzerine bazı performans geliştirmeleri yapılmış halidir. Bu iki yöntem de kelime vektörlerinin öğrenilmesinde kullanılabilir.

2.4.1 Kelime Vektörlerin Öğrenilmesi

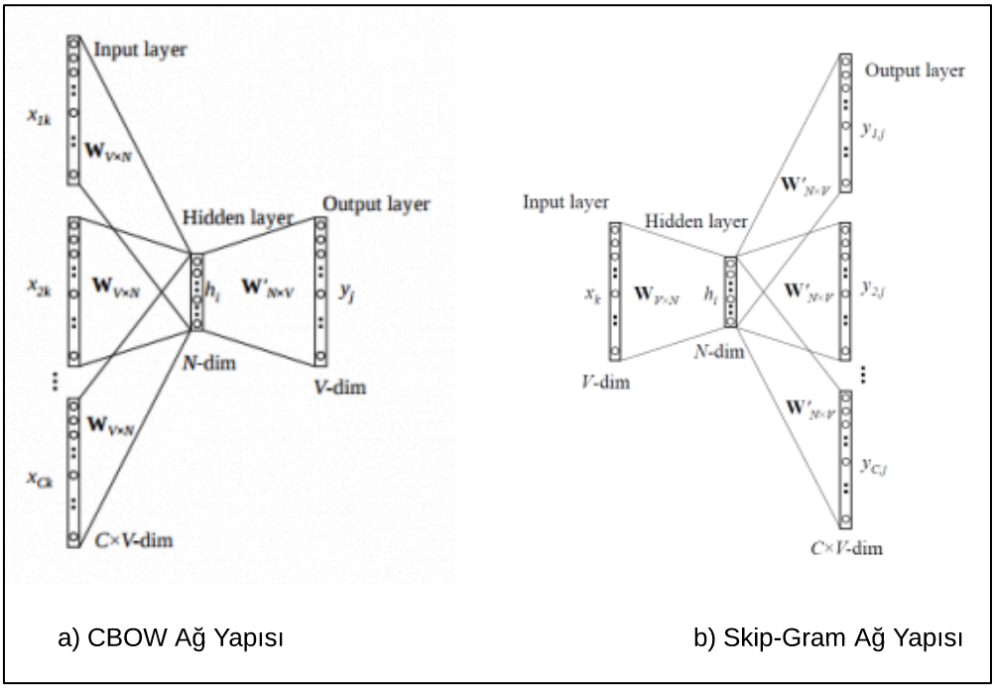
Word2Vec [58], kelimelerin vektörel ifade edilmesi için geliştirilen bir TTKG yaklaşımlardan birisidir. İki adet alt yöntemden oluşur. Bunlar; CBOW (Continious of Words) ve Skip-Gram'dır.

Çerçeve Boyutu (Window Size), Word2Vec için belirlenmesi gereken hiperparametrelerden birisidir. Bu parametre analiz edilen kelimenin sağında ve solunda bulunan n kelimeyi belirlemek için kullanılır. Çerçeve merkezindeki kelimeye Merkez Kelime, bu kelimeye n yakınlıktaki kelimelere Çevreleyen Kelimeler denir. Örnek bir durum Şekil 2.32'de verilmiştir.



Şekil 2.32. Kelime Çerçevesi Örneği

CBOW ve Skip-Gram modelleri girdi ve çıktı yapısı ile birbirlerinden ayrılırlar. CBOW modelinde çevreleyen kelimeler girdi olarak, merkez kelime ise çıktı olarak işlenir. Skip-Gram modelinde ise merkez kelime girdi olarak, çevreleyen kelimeler ise çıktı olarak tahmin edilmeye çalışılır. CBOW ve Skip-Gram modelleri için ağ mimarileri Şekil 2.33'te verilmiştir.



Şekil 2.33. CBOW ve Skip-Gram Ağ Mimarileri [64]

Çerçeve cümle boyunca kelime kelime gezdirilir. Her adımda girdiler ve çıktıları belirten bir veri seti oluşturulur. Örneğin Şekil 2.32'de verilen durum için CBOW ve Skip-Gram'da elde edilen veri Çizelge 2.7'de verilmiştir.

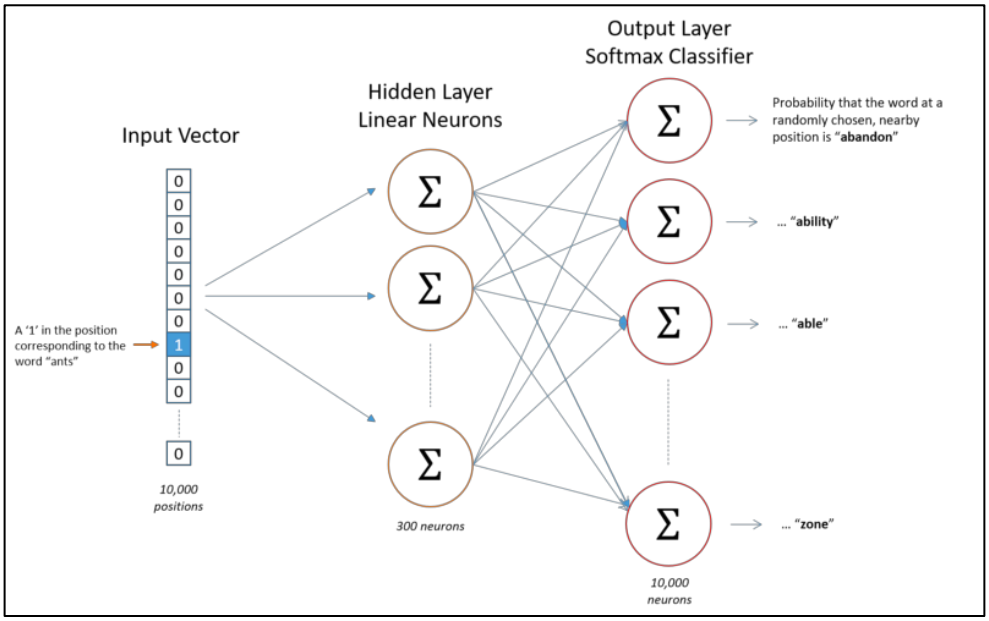
Çizelge 2.7. CBOW ve Skip-Gram İçin Oluşturulan Veri Seti Örneği

CBOW		**Skip-Gram**	
Girdi	**Çıktı**	**Girdi**	**Çıktı**
Korkma, sözmez, bu, yüzen, al, sancak	Şafaklarda	Şafaklarda	Korkma, sözmez, bu, yüzen, al, sancak

Bu işlem metindeki tüm cümleler tüm çerçeve durumları için tekrarlanır. Her cümlede, gezdirilen farklı çerçeve durumları için birer etiketli veri oluşturulur.

CBOW modelleri küçük veri setlerinde daha iyi sonuçlar verirken, büyük verisetlerinde Skip-gram modeli daha iyi çalışmaktadır. CBOW modeli için daha az işlem gücü gerekirken, Skip-gram modeli daha fazla işlem gücü tüketir. CBOW modeli iki veya daha çok anlamlı kelimeleri anlamakta iyi değil iken, Skip-gram iki veya daha çok anlamlı kelimeleri daha iyi bir şekilde öğrenebilmektedir.

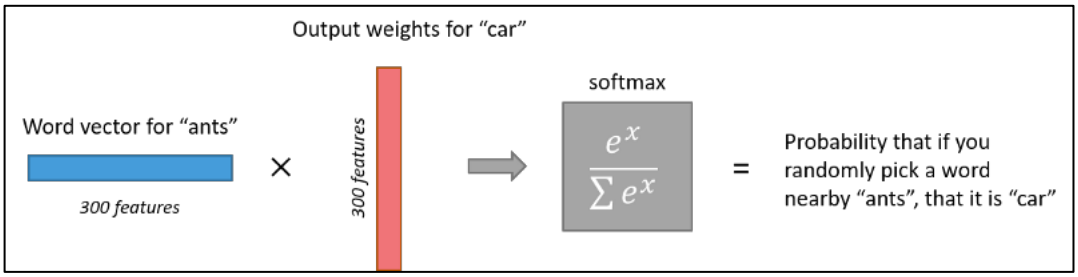
Girdi katmanı ile gizli katman arasında lineer aktivasyon fonksiyonu kullanılır. Yani değerler 0-1 arasına indirgenmeden bir sonraki katmana aktarılır. Gizli katman ile Çıktı Katmanı arasında softmax aktivasyon fonksiyonu kullanılır. Bu sayede bütün kelimelerin olasılık değerlerini içeren ve tekil sözlük boyutunda bir vektör elde edilir. Bu yapının işleyişi Şekil 2.34'te verilmiştir.



Şekil 2.34. Word2Vec İşleyiş Mekanizması [65]

Şekil 2.34'teki örnekte sözlük içerisinde 10.000 farklı kelime bulunmaktadır. Her bir kelime, gizli katmanda 300 boyutunda bir vektör ile temsil edilmektedir. Çıktı olarak her kelime için sözlükte bulunan kelimelerle yakın bulunma olasılıkları hesaplanır.

İki kelimenin birbirlerine n (çerçeve boyutu) yakınlıkta olması olasılığı kelimelerin 300 boyutundaki vektörlerinin çarpılması ile hesaplanır. Örnek bir hesaplama Şekil 2.35'te verilmiştir.



Şekil 2.35. İki Kelimenin Birbirlerine Yakın Bulunma Olasılıkları [65]

Word2Vec'ün çalışma mantığı Yapay Sinir Ağı yaklaşımına dayanır. İlk olarak ağırlıklar ilklendirilir, ardından ileri yayılım, geri yayılım ve parametre güncelleme adımları uygulanır. Bu işlem belirlenen apoch adımı kadar tekrarlanır. Word2Vec modeli için varsayılan epoch sayısı 5 olarak belirlenmiştir. Ancak, bu değer artırılarak performans artışı sağlanabilir.

Word2Vec modelinin birçok matris çarpımı işlemi yapması gerekmektedir. Bu nedenle bu modelin çalışabilmesi için çok fazla işlem gücüne ihtiyaç duyulur. Word2Vec modelinin çalışma performansını artırmak için bazı yöntemler kullanılmaktadır. Bu yöntemler sayesinde model daha hızlı çalışabilmektedir. Performans artışı için kullanılan yöntemler şunlardır;

- Anlamsal kelime gruplarını tek bir kelime olarak işlemek
- Durak kelimelerin işleme tabi tutulmaması (ve, ama, şey, belki)
- Negatif Örnekleme [66]: Çıktı katmanında elde edilen vektörlerde bulunan verilerin sadece bazılarının ağırlık güncelleştirmesinde kullanılması

CBOW ve Skip-Gram yaklaşımları çerçeve tabanlı işleyiş mekanizmalarına sahip oldukları için çerçeve boyutunun doğru belirlenmesi sistemin performansında doğrudan etkilidir. Çevçeve boyutunun sabit olması kelime ya da kelime gruplarının

birlikte kullanımlarının öğrenilmesinde sorun teşkil edebilmektedir. Çerçevenin adım adım ilerletilmesi şeklinde korpus üzerinde gezildiği için korpus verimli bir şekilde kullanılamamaktadır. GloVe, kelime vektörlerinin daha verimli ve başarılı öğrenebilmesi için çevreleyen kelimelerin belirlenmesinde istatistiksel tabanlı bir yaklaşım kullanan kelime vektörü öğrenme yöntemidir. GloVe ile corpus daha verimli bir şekilde işlenerek daha başarılı kelime vektörleri öğrenilebilmektedir.

P, iki kelimenin birlikte kullanımını (co-occuring) ifade eden olasılık değeridir. Birlikte kullanım matrisi (Co-occurance matrix) kelimelerin birlikte bulunmalarının frekansını tutan matrise verilen isimdir. V, çerçeve merkezindeki kelimeyi temsil etmektedir. U ise çerveve içerisinde merkez kelime haricindeki kelimeleri temsil etmektedir. Örneğin aşağıdaki cümlelerin kullanıldığı bir corpus için oluşturulan Birlikte Kullanım Matrisi Çizelge 2.8'de verilmiştir.

- I like deep learning
- I like NLP
- I enjoy flying

Birlikte Kullanım Matrisinde (X);

- $X_{i,j}$: j. Kelimenin i. Kelime ile tüm corpus içerisinde birlikte bulunması sayısı
- X_i: i. Kelimenin tüm corpus içerisinde geçme sayısı

İki kelimenin birlikte bulunma olasılığı $P_{(i,j)} = \frac{X_{i,j}}{X_i}$ ile hesaplanır.

GloVe modeli temel olarak hata fonksiyonunun (J) modellenmesi sırasında kelimelerin olasılık oranlarını da kullanır. Kelimelerin birlikte kullanım oranları ile güncellenen hata fonksiyonu ile CBOW ve Skip-Gram daki gibi çerçeve gezdirerek çevreleyen kelimelerin belirlenmesi işlemi ortadan kalmaktadır. Çünkü $P_{i,j}$ ile birlikte bulunma olasılığı yüksek kelimeler birbirlerine yakın geçtilerse bu kelimeler öğrenme işleminde diğer kelimelerden daha önemli rol oynaması sağlanmıştır.

Çizelge 2.8. Birlikte Kullanım Matrisi

Sayı	I	Like	Enjoy	Deep	Learning	NLP	Flying	.
I	0	2	1	0	0	0	0	0
Like	2	0	0	1	0	1	0	0
Enjoy	1	0	0	0	0	0	1	0
Deep	0	1	0	0	1	0	0	0
Learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
Flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

2.5 Öğrenme Transferi (Transfer Learning)

Bazı problem türlerinde derin öğrenme mimarilerinin eğitilmesi için çok fazla işlem gücü çok fazla sistem kaynağına ihtiyaç duyulur. Problem karmaşıklığı ile orantılı olarak ihtiyaç duyulan işlem gücü ve eğitim aşaması için gereken süreler artmaktadır. Bu nedenle yüksek işlem gücü ve süre gerektiren problemler üzerinde çalışan araştırmacılar Öğrenme Transferinden faydalanırlar. Bu sayede daha az uğraş ile büyük derin öğrenme mimarileri inşa edebilirler.

Öğrenme transferi, bir problemin çözümü için önceden öğrenilmiş parametrelerin kullanılmasına denir. Örneğin, nesne tanımaya yönelik bir görüntü işleme problemi üzerinde çalışıyorsanız, tespit etmek istediğiniz nesne sayısının fazla olması sistem için çok önemli bir kriterdir. Ancak, binlerce nesneyi tanıyacak sistemlerin eğitilmesi için oldukça güçlü bilgisayarlardan oluşan laboratuvarlara ihtiyaç duyulabilir. Böyle bir durumda, oldukça iyi sistem kaynaklarına sahip bilgisayarlar tarafından uzun sürelerce eğitilip kaydedilmiş model ve parametrelerin sadece okunması ile binlerce nesneyi tanıma yeteğine sahip bir sisteme sahip olabiliriz. Öğrenme transferi ile araştırmacılar edindikleri tecrübeleri birbirlerine aktabilirler. Binlerce nesneyi tanıyabilen bir modelin sadece öğrenilmiş parametrelerini okuyarak o sistemi kullanmaya başlayabiliriz. Aynı zamanda bu sistem üzerine geliştirmelere de devam edebiliriz. Tanınan nesneler

arasına kendi belirlediğimiz bir nesneyi dahil etmek için eğitilmiş modele bir miktar hedef nesneyi eğitmek yeterli olacaktır.

Öğrenme transferi, birçok uygulama alanında birçok araştırmacının zaman ve sistem kaynağından tasarruf etmesine olanak tanımaktadır. Bu sayede araştırmalar kümülatif olarak ilerleyecek ve adım adım daha başarılı sistemler geliştirilebilecektir.

Öğrenme transferi, DDİ problemlerinde de oldukça yaygın olarak kullanılmaktadır. Örneğin, kelime vektörlerinin öğrenilmesi oldukça maliyetli bir işlemdir. Kelime vektörlerinin başarılı bir şekilde öğrenilebilmesi için çok büyük corpuslar üzerinde öğrenme işleminin yapılması gerekmektedir. Çok büyük çapta verinin toplanması başlı başına oldukça zor bir problem iken, bu verilerin işlenmesi ve anlamlandırılması da oldukça zor ve zaman alan bir problemdir. Alanında önde gelen bazı teknoloji firmaları, bazı üniversiteler ve araştırmacılar kaliteli kelime vektörlerinin öğrenilebilmesi için çok zaman ve işlem gücü gerektiren bu işlemleri yapararak elde ettikleri parametreleri diğer araştırmacıların da kullanabilmesi için paylaşabilmektedirler. Bu sayede donanımsal kaynakları yeterli olmayan araştırmacılar da iyi öğrenilmiş modeller üzerinde çalışabilmektedirler.

DDİ problemlerinde öğrenme transferinin uygulanabileceği alanların başında kelime vektörlerinin öğrenilmesi gelmektedir.

Stanford Üniversitesinde Pennington ve arkadaşları [59] kelime vektörlerinin öğrenilmesi için ortaya attıkları GloVe yöntemi ile çok büyük corpuslar üzerinde kelime vektörlerini öğrenmişler ve elde ettikleri parametleri diğer araştırmacıların kullanımı için paylaşmışlardır [67]. Yapılan çalışmada dört farklı büyük corpus üzerinde kelime vektörleri öğrenilmiştir. Bu corpuslara ilişkin detaylar şu şekildedir;

- Wikipedia 2014 + Gigaword 5: 6 Milyar token (jeton) ın işlendiği bu corpusdan elde edilen tekil kelime sayısı 400,000'dir. Bu corpus üzerinde öğrenilen kelime vektörleri 50, 100, 200 ve 300 boyutlu olarak öğrenilip paylaşılmıştır.
- Common Crawl 1: Dünya genelinde web sayfaların crawl edilmesi yoluyla elde edilmiştir. 42 milyar tokenın işlendiği bu corpusdan elde edilen tekil kelime

sayısı 1,9 milyondur. Bu corpus üzerinde 300 boyutlu kelime vektörleri öğrenilmiştir.

- Common Crawl 2: Dünya genelinde web sayfalarının crawl edilmesi yoluyla el edilmiştir. 840 milyar tokenın işlendiği bu corpusdan elde edilen tekil kelime sayısı 2,2 milyondur. Bu corpus üzerinde 300 boyutlu kelime vektörleri öğrenilmiştir.
- Twitter: Bu corpus, 2 milyar tweet'ten elde edilen 27 milyar tokendan oluşmaktadır. Bu corpusda işlenen tekil kelime sayısı 1,2 milyondur. Bu corpus üzerinde 25, 50, 100 ve 200 boyutlarında kelime vektörleri öğrenilmiştir.

Stanford Üniversitesinin gerçekleştirdiği çalışmanın yanı sıra Google firması Google News üzerinde yer alan kelimeleri kullanarak oluşturduğu kelime vektörlerini araştırmacıların kullanımına sunmuştur [68]. İşlenen corpus üzerinde 100 milyar token bulunmaktadır. Corpusda yer alan tekil kelime sayısı ise 3 milyondur. Öğrenilen vektör boyutu ise 300'dur.

Facebook firması tarafından geliştirilen çalışmada [69] 294 dil için ayrı ayrı kelime vektörleri öğrenilmiş ve araştırmacılara sunulmuştur. Öğrenilen kelime vektörleri 300 boyutludur. Bu kelime vektörleri [56]'den indirilebilir.

Öğrenilmiş kelime vektörlerinin paylaşıldığı birçok kaynak literatürde yer almaktadır [70-72]. Türkçe diline özel olarak eğitilmiş kelime vektörlerine ilişkin çalışmalar [73] da literatürde yer almaktadır.

2.6 Başarı Değerlendirme Metrikleri

Gerçekleştirilen testler, dört farklı başarı metriği ile değerlendirilmiştir. Bu metrikler şunlardır;

Tutturma (Precision): Getirilen bilgideki doğru sonuçların, getirilen bilginin tamamına oranı olarak hesaplanır. Tutturma metriğinin formülü Denklem 2.7'de verilmiştir.

$$Tutturma = \frac{DP}{DP+YP} \tag{2.7}$$

Hassaslık (Recall, Sensitivity): Getirilen doğru sonuçların, getirilmesi gereken doğru sonuçlara oranı ile hesaplanır. Hassaslık metriğinin formülü Denklem 2.8'de verilmiştir.

$$Hassaslık = \frac{DP}{DP+YN} \quad (2.8)$$

F-Skor (F-Measure): F-skoru, Tutturma ve Hassaslık değerlerinin harmonik ortalamasıdır. F-Skor metriğine ilişkin formül Denklem 2.9'da verilmiştir.

$$FSkor = 2 \times \frac{Tutturma \times Hassaslık}{Tutturma+Hassaslık} \quad (2.9)$$

Başarı Oranı (Accuracy): Veri örneklerinin doğru sınıflandırılma oranıdır. Başarı Oranı formülü Denklem 2.10'da verilmiştir.

$$Başarı\ Oranı = \frac{DP+DN}{DP++DN+YN+YP} \quad (2.10)$$

Duyarlılık (True Negative Rate, Specificity): Negatif örnekler içerisinde doğru sınıflandırma oranı. Duyarlılık metriğinin formülü Denklem 2.11'de verilmiştir.

$$Duyarlılık = \frac{DN}{DN+YP} \quad (2.11)$$

BÖLÜM 3

DENEYSEL SONUÇLAR

Sistemin başarılı çalışmasının önemli olması kadar sistemin hızlı çalışması da önemlidir. Derin öğrenme mimarileri ile hızlı eğitim ve testlerin yapılabilmesi için GPU (Graphical Proceeeing Unit)'ler günümüzde yaygın bir şekilde kullanılmaktadır. Derin öğrenmede hesaplanan matematiksel işlemler paralelleştirilebilir işlemlerdir. GPU lar üzerinde yer alan çok fazla sayıda çekirdek ile hesaplamasal işlemler çok sayıda çekirdeğe dağıtılarak işlemler hızlandırılabilmektedir. Bu proje kapsamında fazla epoch sayısı ile uzun eğitim işlemlerine başlanmadan önce CPU ve GPU arasında performans farkı analiz edilmiştir. Analiz sonucunda elde edilen sonuçlara göre sistemin eğitim ve test aşamasında kullanılmak üzere bir GPU temin edilmesinin gerekli olup olmadığı, GPU temin edilmeden yapılan eğitim süreleri makul seviyelerde ise CPU üzerinde testlere devam edilmesi düşünülmüştür. Bu nedenle doğruluk oranının artırılmasına yönelik testlere başlanmadan önce performans analizi yapılmasına ihtiyaç duyulmuştur. CPU ve GPU arasında performans karşılaştırması verilerine Bölüm 3.1'de, sistem başarısına ilişkin test sonuçları ise Bölüm 3.2'de açıklanmıştır.

3.1 CPU ve GPU Arasında Performans ve Hiperparametre Analizi

Derin öğrenmedeki hesaplamasal işlemler paralelleştirmeye uygun işlemlerdir. Birbirinden bağımsız bir şekilde işlem yürütebilen işlem birimlerinin (çekirdek) sayısının çokluğu, bir işlemin paralelleştirilebilme miktarını belirler. Kişisel kullanım amaçlı bilgisayarlarda bulunan dört veya sekiz çekirdek üzerinde yapılacak paralelleştirme

işlemi ile üzerinde binlerce çekirdek barındıran GPU lar üzerinde yapılan paralelleştirme işlemleri aynı seviyede olmayacaktır.

Bu proje kapsamında gerçekleştirilen testlerde büyük miktarda veriler kullanılmaktadır. Bu nedenle testler kişisel bilgisayarlarda yürütülememektedir. CPU testleri Roksit Firmasının [74] sağladığı sunucular üzerinde gerçekleştirilmiştir. GPU testler için ise Floydhub [75]'dan bulut tabanlı GPU sunucusu kiralanmıştır.

CPU testleri esnasında kullanılan işlemci modeli Intel® Xeon® Gold 6126 [76]. Testlerin gerçekleştirildiği veri merkezinde bu işlemcilerden çok sayıda bulunmaktadır. Sunucularda kullanılmak üzere tasarlanmış bu işlemcilerin her birisinde 12 çekirdek bulunmaktadır. Çekirdeklerin her biri 2.6 Ghz (Turbo ile 3.70 Ghz) çalışma frekansına sahiptir. Testlerde kullanılmak üzere tahsis edilen kaynaklar düzenlenebilmektedir. Örneğin, 12 çekirdek üzerinde testler yapılabileceği gibi 50 çekirdek üzerinde de testler yapılması mümkündür. Ayrıca çekirdeklerin çalışma frekansları da ayarlanabilir parametreler arasındadır. Çekirdeklerin 1300 Mhz çalışma frekansına sahip olduğu durumda da testler yapılabileceği gibi çekirdek çalışma frekansının 2600 Mhz olduğu durumlar da testler yapılabilmektedir. Şirket sunucusunda çalıştırılacak testlerin çalışan sistemleri etkilememesi için testlere tahsis edilen maximum çalışma frekansına bir limit de konulabilmektedir.

GPU testlerinin yapılabilmesi için Floydhub [75]'dan bir sunucu kiralanmıştır. Kiralanan sunucuda Nvidia Tesla K80 [77] modelinde GPU bulunmaktadır. Tesla K80, içerisinde 2 farklı işlem birimi barındırmaktadır. Her işlem birimi ayrı GPU lar olarak düşünülebilir. Tesla K80'de 2x2496 adet çekirdek bulunmaktadır. Her bir çekirdeğin çalışma frekansı 562 Mhz, boost modunda ise 875 Mhz'dir. Nvidia Tesla K80'nin diğer Tesla mimarisindeki GPU'lar ile karşılaştırması Şekil 3.1'te verilmiştir.

	Tesla K80	Tesla K40	Tesla K20X	Tesla K20
Stream Processors	2 x 2496	2880	2688	2496
Core Clock	562MHz	745MHz	732MHz	706MHz
Boost Clock(s)	875MHz	810MHz, 875MHz	N/A	N/A
Memory Clock	5GHz GDDR5	6GHz GDDR5	5.2GHz GDDR5	5.2GHz GDDR5
Memory Bus Width	2 x 384-bit	384-bit	384-bit	320-bit
VRAM	2 x 12GB	12GB	6GB	5GB
Single Precision	8.74 TFLOPS	4.29 TFLOPS	3.95 TFLOPS	3.52 TFLOPS
Double Precision	2.91 TFLOPS (1/3)	1.43 TFLOPS (1/3)	1.31 TFLOPS (1/3)	1.17 TFLOPS (1/3)
Transistor Count	2 x 7.1B(?)	7.1B	7.1B	7.1B
TDP	300W	235W	235W	225W
Cooling	Passive	Active/Passive	Passive	Active/Passive
Manufacturing Process	TSMC 28nm	TSMC 28nm	TSMC 28nm	TSMC 28nm
Architecture	Kepler	Kepler	Kepler	Kepler
Launch Price	$5000	$5499	~$3799	~$3299

Şekil 3.1. Tesla Mimarisi GPU'ların Karşılaştırması [78]

Floydhub'da kiralanan sunucularda Tesla K80'in sadece 1 işlem birimi kullanıcılara tahsis edilebilmektedir. Bu nedenle testler esnasında 2496 çekirdek ve 12 GB vMemory kullanılmıştır.

Kelime Vektörlerinin öğrenilmesi, yüksek işlem gücü gerektiren bir işlemdir. Bu nedenle testlerin hızlı sonuçlandırılabilmesi için performans testlerinde Stanford Üniversitesinin GloVe yöntemi kullanarak hesaplamış olduğu kelime vektörleri [67] kullanılarak öğrenme transferi uygulanmıştır. Testler esnasında kullanılan kelime vektörleri 6 milyar kelimenin işlenmesi ile oluşturulan kelime vektörleridir. Kelime vektörlerinin çıkarılmasında 400,000 tekil kelime yer almaktadır. Testler esnasında kullanılan kelime vektörleri 300 boyutludur.

Gerçekleştirilen testler maliyetli oldukları için ancak 2 epoch boyunca çalıştırılmış ve elde edilen sürelerin ortalaması alınmıştır. Tüm testler için kullanılan ortak hiperparametreler şu şekildedir;

- Epoch Sayısı: 2
- Hata Fonksiyonu: Categorical Crossentropy

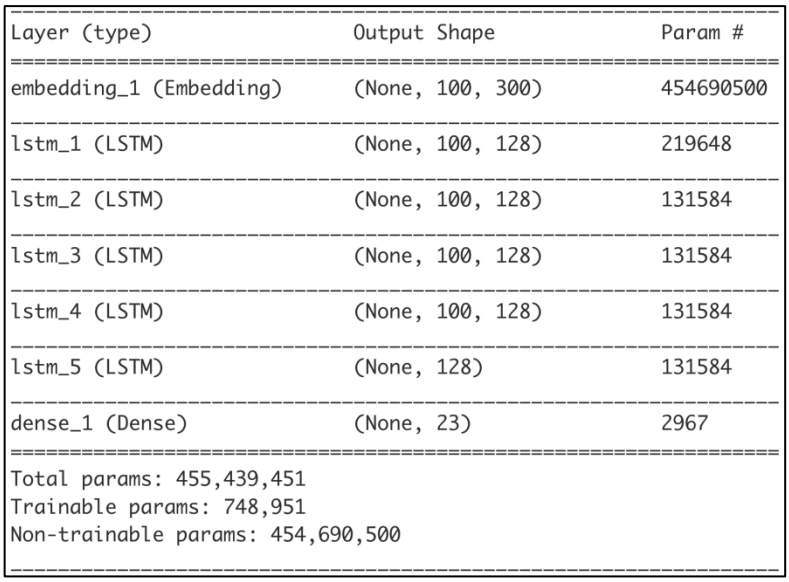
- Maximum Sekans Boyutu: 100
- Parametre Güncelleme Fonksiyonu: ADAM, Öğrenme Katsayısı: 0.01, Beta_1: 0.9, Beta_2: 0.999.
- Test Split Oranı: 0.2
- Sözlükte bulunan tekil kelime sayısı: 1,515,634
- Toplam Veri Örneği Sayısı: 1,210,954
- Eğitim Örneği Sayısı: 968,755
- Test Örneği Sayısı: 242,199
- Öğrenme Transferi kullanılan testlerde kelime vektörü boyutu: 300 [59]

Testler esnasında iki farklı derin öğrenme mimarisi test edilmiştir. Test edilen mimarilerden birisi tek katmanlı bir yapıya sahipken test edilen diğer mimari 5 katmanlı yapıda oluşturulmuştur. Öğrenme transferi kullanılarak gerkleştirilen testlerde kullanılan tek katmanlı derin öğrenme mimarisi Şekil 3.2'de, 5 katmanlı derin öğrenme mimarisi ise Şekil 3.3'de verilmiştir.

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 100, 300)	454690500
lstm_1 (LSTM)	(None, 128)	219648
dense_1 (Dense)	(None, 23)	2967

Total params: 454,913,115
Trainable params: 222,615
Non-trainable params: 454,690,500

Şekil 3.2. Öğrenme Transferi Kullanılan Tek Katmanlı Mimari



Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 100, 300)	454690500
lstm_1 (LSTM)	(None, 100, 128)	219648
lstm_2 (LSTM)	(None, 100, 128)	131584
lstm_3 (LSTM)	(None, 100, 128)	131584
lstm_4 (LSTM)	(None, 100, 128)	131584
lstm_5 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 23)	2967

Total params: 455,439,451
Trainable params: 748,951
Non-trainable params: 454,690,500

Şekil 3.3. Öğrenme Transferi Kullanılan 5 Katmanlı Mimari

Şekil 3.2 ve Şekil 3.3'te öğrenme transferi kullanılarak gerçekleştirilen testlerde kullanılan derin öğrenme mimarileri verilmiştir. Şekillerde görüleceği üzere öğrenme transferi kullanıldığı için hazır kullanılan (öğrenilmeyen) parametre sayısı fazla iken öğrenilen parametre sayısı daha azdır. Gerçekleştirilen performans analizi testlerinde işlem gücü için kullanılan donanımın ve bazı hiperparametrelerin performansa etkisi incelenmiştir.

Çalışma kapsamında 4 farklı donanım senaryosu test edilmiştir. Test edilen senaryolarda CPU'üzerindeki çekirdek sayısı, bir çekirdeğin sahip olduğu çalışma frekansının değişimi gibi durumlar test edilmiştir.

CPU'lar için test edilen 3 farklı donanım senaryosu şu şekildedir.

- 16 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 25000 Mhz. Her bir çekirdek için çalışma frekansı: 1300 Mhz
- 16 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 25000 Mhz. Her bir çekirdek için çalışma frekansı: 2600 Mhz
- 24 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 37500 Mhz. Her bir çekirdek için çalışma frekansı: 2600 Mhz
- 2496 çekirdekli GPU. Her çekirdeğin çalışma frekansı 562 Mhz.

3.1.1 Performans Analizi Deney 1

Gerçekleştirilen ilk deneyde 1300 Mhz CPU, 2600 Mhz CPU ve GPU karşılaştırılmıştır. Test esnasında öğrenme transferi uygulanmıştır. Bu test kapsamında kullanılan hiperparametreler şunlardır;

- Öğrenme Transferi kullanılmıştır.
- CPU testlerinde 16 çekirdek kullanılmıştır.
- 128 Nörona sahip 1 gizli katman kullanılmıştır.
- Batch Boyutu 64 olarak belirlenmiştir.
- Tek katmanlı derin öğrenme mimarisi (Şekil 3.2) test edilmiştir.

Test sonrasında elde edilen çalışma süreleri karşılaştırmaları Çizelge 3.1'de verilmiştir. Çizelgede donanım kısmında verilen frekans değerleri tek bir çekirdeğin sahip olduğu frekans değeridir.

Çizelge 3.1. Deney1 Sonucu Çalışma Süreleri Karşılaştırmaları

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 1300 Mhz	64	71,3 dk	5,5 dk
CPU / 2600 Mhz	64	36 dk	2,7 dk
GPU	64	37,3 dk	2,1 dk

Testler için kaynak tüketimleri Çizelge 3.2'de verilmiştir. Çizelge 3.1'deki verilere göre çekirdek çalışma frekansının iki katına çıkmasının sistemin çalışma hızını yaklaşık olarak **iki katına** çıkardığı, çalışma süresini yarıya düşürdüğü görülmüştür. Çalışma frekansı 2600 Mhz olan çekirdekli CPU'da yapılan test ile GPU üzerinde gerçekleştirilen testlerde çalışma süreleri birbirlerine yakın değerler çıkmıştır.

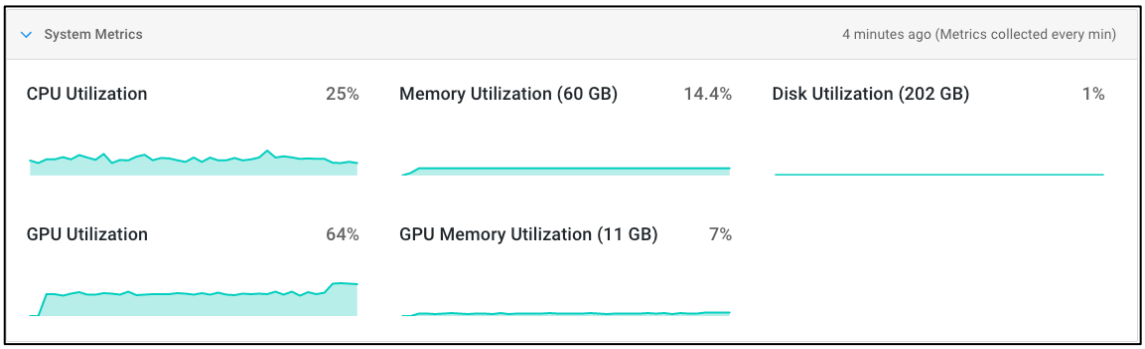
Şekil 3.4'te verilen sistem kaynakları kullanımı incelendiğinde GPU'nun kullanım veriminin düşük olması nedeniyle çalışma süresi kıyaslamasında GPU'nun CPU (2600 Mhz)'ya kıyasla bir hızlandırma sağlamadığı görülmüştür. Bu nedenle GPU kullanım verimini artırmaya yönelik testlerin yapılmaya devam edilmesi gerektiği görülmüştür.

Testler esnasında tüketilen sistem kaynakları anlık olarak değişkenlik göstermektedir. Çizelge 3.2'de verilen kaynak tüketimleri yaklaşık değerlerdir.

Çizelge 3.2. Deney 1 Kaynak Tüketimleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanım Yüzdesi	Bellek Kullanım Miktarı
CPU / 1300 Mhz	%40 - % 50	--	10 – 12 GB Ram
CPU / 2600 Mhz	%40 – % 50	--	10 – 12 GB Ram
GPU	%35 – %40	%40 – % 40	9 GB Ram 0,5 – 1 GB vMemory

GPU üzerinde yapılan testler esnasında sistem kaynaklarının kullanımına ilişkin detaylar Şekil 3.4'te verilmiştir. Şekil 3.4'te verilen sistem kaynakları eğitim işlemi için tüketilen kaynakları ve ardından test işlemi için tüketilen kaynakları göstermektedir.



Şekil 3.4. Deney 1 GPU Üzerinde Sistem Kaynakları Kullanımı

3.1.2 Performans Analizi Deney 2

GPU üzerinde sistem kaynaklarının daha verimli kullanılması çalışma sürelerinin kısalmasına olanak tanımıştır. GPU kullanımının artırılması için Batch Boyutunun artırılması durumu denenmiştir. Batch boyutunun artırılması gerçekleştirilecek hesaplamalarda büyük matrislerin çarpımının kullanılmasına neden olmaktadır. Büyük matris çarpımları yüksek işlem gücü gerektiren işlemlerdir. Bu matris çarpımları paralelleştirilebilir işlemler olduğu için GPU üzerinde daha hızlı gerçekleştirilecektir. Batch boyutunun arttırılmasının GPU kullanımını artmasına ve işlem sürelerinin kısalmasına olanak tanıması beklenmektedir.

Bu test için belirlenen hiperparametreler şunlardır;

- Öğrenme Transferi kullanılmıştır.
- CPU testlerinde 16 çekirdek kullanılmıştır.
- 128 Nörona sahip 1 gizli katman kullanılmıştır (Şekil 3.2).
- Batch Boyutu 500 olarak belirlenmiştir.

Testler sonucunda elde çalışma sürelerine ilişkin bilgiler Çizelge 3.3'te verilmiştir.

Çizelge 3.3. Deney 2 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 1300 Mhz	500	44,4 dk	4,2 dk
CPU / 2600 Mhz	500	21,3 dk	2,0 dk
GPU	500	6,4 dk	0,49 dk

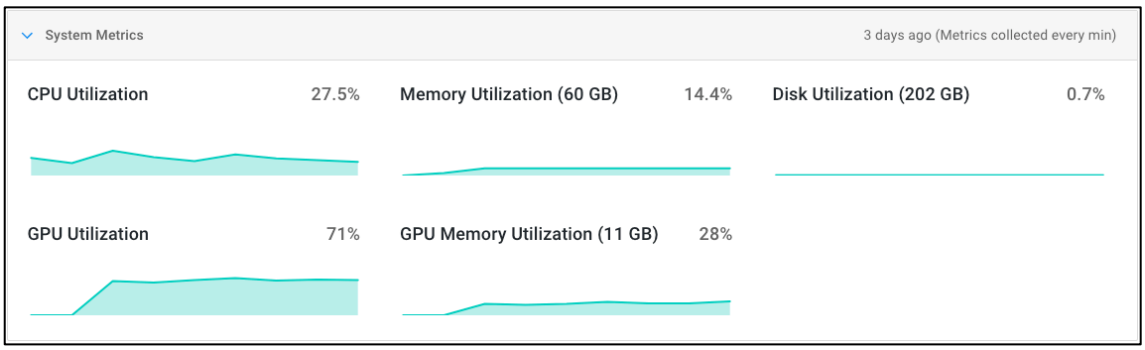
Deney için kaynak tüketimleri Çizelge 3.4'de verilmiştir. Çizelge 3.4'deki verilere göre Test 1'dekigibi çekirdek çalışma frekansının iki katına çıkarılması sistemin çalışma hızını da yaklaşık olarak **iki katına** çıkardığı, çalışma süresini de yarıya düşürdüğü görülmüştür. Çalışma frekansı 2600 Mhz olan çekirdekli CPU'da yapılan test ile GPU üzerinde gerçekleştirilen testlerde çalışma süreleri karşılaştırıldığında GPU üzerinde gerçekleştirilen testlerin 3,37 kat daha hızlı çalıştığı tespit edilmiştir.

Çizelge 3.4. Deney 2 Kaynak Tüketimleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 1300 Mhz	%40 - % 50	--	10 – 12 GB Ram
CPU / 2600 Mhz	%40 – % 50	--	10 – 12 GB Ram
GPU	%35 – %45	%75 – %80	9 GB Ram 3,5 – 4 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ait detaylı bilgiler Şekil 3.5'de verilmiştir. Şekil 3.5'te verilen sistem kaynakları kullanımı

incelendiğinde GPU'nun kullanım veriminin Test 1'e kıyasla artması nedeniyle çalışma sürelerinin azaldığı Çizelge 3.3'te görülmektedir.



Şekil 3.5. Deney 2 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.3 Performans Analizi Deney 3

Bu deney adımında batch boyutu artırılarak GPU kullanımının artırılması durumu denenmiştir. Bu test için kullanılan hiperparametreler şu şekildedir;

- Öğrenme Transferi kullanılmıştır.
- CPU testlerinde 16 çekirdek kullanılmıştır.
- 128 Nörona sahip 1 gizli katman kullanılmıştır (Şekil 3.2).
- Batch Boyutu **1000** olarak belirlenmiştir.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.5'de verilmiştir.

Çizelge 3.5. Deney 3 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz	1000	20,61 dk	1,98 dk
GPU	1000	4,6 dk	0,33 dk

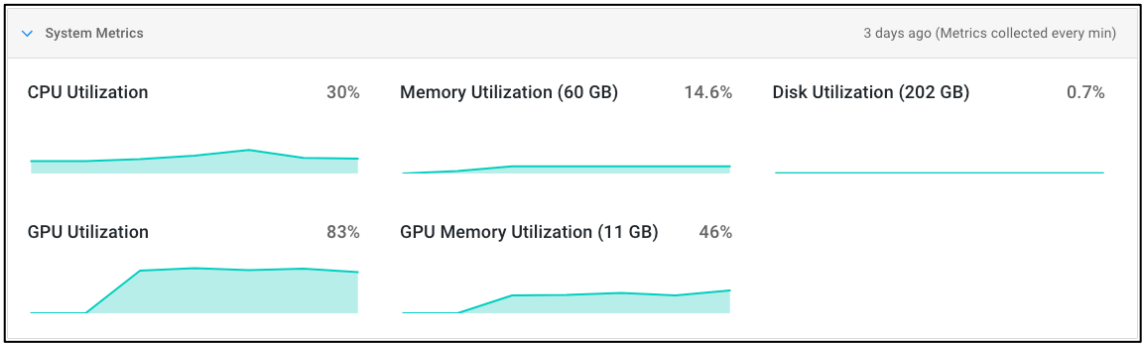
Çizelge 3.5'te verilen bilgilere göre GPU üzerinde gerçekleştirilen testler CPU üzerinde gerçekleştirilen testlere kıyasla **4,48 kat** daha hızlı çalışmıştır. Batch boyutunun artırılmasının GPU üzerinde Test 2 sonuçlarına (6,4 dk – 0,49 dk) kıyasla hızı artırıcı etki oluşturduğu gözlenmiştir. Batch boyutunun artırılmasının CPU üzerinde Test 2 sonuçlarına (21,3 dk – 2,0 dk.) kıyasla bir hız artışı gerçekleştirmediği gözlenmiştir.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.6'da verilmiştir.

Çizelge 3.6. Deney 3 Kaynak Kullanım Değerleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 2600 Mhz	% 50–%55	--	10 – 12 GB Ram
GPU	%35 – %45	%85 – %90	9 GB Ram 4,5 – 5,5 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.6'da verilmiştir.



Şekil 3.6. Deney 3 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.4 Performans Analizi Deney 4

Bu deney adımında da batch boyutu artırılarak GPU kullanımının artırılması durumu denenmiştir. Bu test için kullanılan hiperparametreler şu şekildedir;

- Öğrenme Transferi kullanılmıştır.
- CPU testlerinde 16 çekirdek kullanılmıştır.
- 128 Nörona sahip 1 gizli katman kullanılmıştır (Şekil 3.2).
- Batch Boyutu **5000** olarak belirlenmiştir.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.7'de verilmiştir.

Çizelge 3.7. Deney 4 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz	5000	20,86 dk	2,05 dk
GPU	5000	3,8 dk	0,25 dk

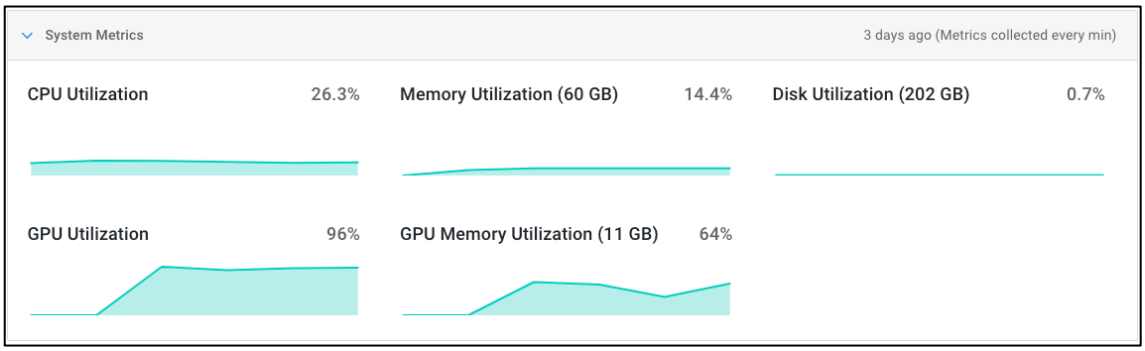
Çizelge 3.7'te verilen bilgilere göre GPU üzerinde gerçekleştirilen testler CPU üzerinde gerçekleştirilen testlere kıyasla **5,48** kat daha hızlı çalışmıştır. Batch boyutunun artırılmasının GPU üzerinde Test 3 sonuçlarına (4,6 dk – 0,33 dk) kıyasla hızı artırıcı etki oluşturduğu gözlenmiştir. Batch boyutunun artırılmasının CPU üzerinde Test 3 sonuçlarına (21,3 dk – 2,0 dk.) kıyasla bir hız artışı gerçekleştirmediği gözlenmiştir.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.8'de verilmiştir.

Çizelge 3.8. Deney 4 Kaynak Kullanım Değerleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 2600 Mhz	%60 – % 70	--	12 – 13 GB Ram
GPU	%30 – %35	%95 – %99	9 GB Ram ~ 7 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.7'de verilmiştir.



Şekil 3.7. Deney 4 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.5 Performans Analizi Deney 5

Bu deney adımında da katman sayısının artırılmasının çalışma sürelerine etkisi ve CPU üzerindeki çekirdek sayısının hıza etkisi incelenmiştir.

Bu test için kullanılan hiperparametreler şu şekildedir;

- Öğrenme Transferi kullanılmıştır.
- CPU testlerinde 16 ve 24 çekirdek kullanılmıştır.
- 128 Nörona sahip 5 gizli katman kullanılmıştır (Şekil 3.3).
- Batch Boyutu **1000** olarak belirlenmiştir.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.9'de verilmiştir.

Çizelge 3.9. Deney 5 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz 16 çekirdek	1000	83,8 dk	6,83 dk
CPU / 2600 Mhz 24 Çekirdek	1000	85,05 dk	4,7 dk
GPU	1000	26,7 dk	1,77 dk

Çizelge 3.9'da verilen bilgilere göre, CPU üzerinde gerçekleştirilen testlerde çekirdek sayısının artmasının eğitim süresinde bir iyileşmeye neden olmadığını göstermektedir. Ancak, çekirdek sayısının artmasının test süresinin azalmasını sağladığı görülmüştür.

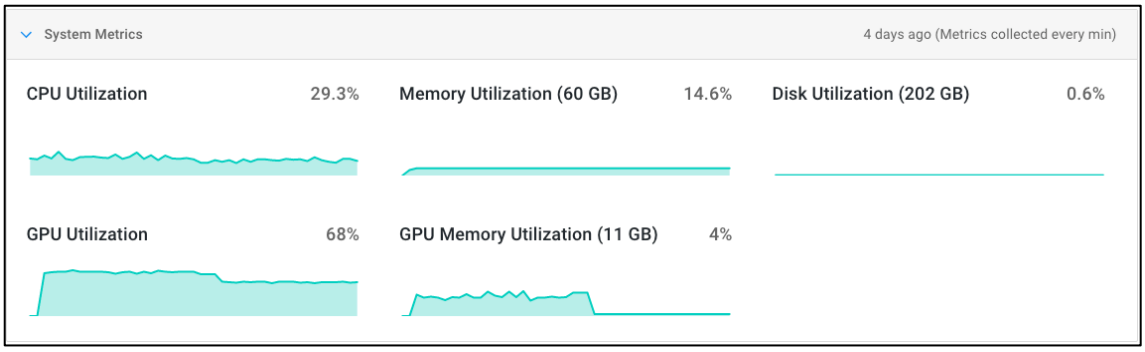
GPU üzerinde gerçekleştirilen testler, CPU üzerinde gerçekleştirilen testlere kıyasla yaklaşık **3,20** kat daha hızlı çalışmıştır. Aynı batch boyutu ile tek katmanlı mimaride yapılan GPU sonuçları (4,6 dk – 0,33 dk) ile 5 katmanlı mimari kullanılan bu test arasında **5,8** kat hız farkı olduğu tespit edilmiştir. Katman sayısının artmasının çalışma sürelerinde doğrusala yakın bir artış etkisi olduğunu göstermiştir.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.10'de verilmiştir.

Çizelge 3.10. Deney 5 Kaynak Kullanım Değerleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 2600 Mhz 16 çekirdek	%60 – % 70	--	12 – 13 GB Ram
CPU / 2600 Mhz 24 Çekirdek	%60 – % 70	--	~13 GB Ram
GPU	%30 – %35	%70 – %80	9 GB Ram ~ 4 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.8'de verilmiştir.



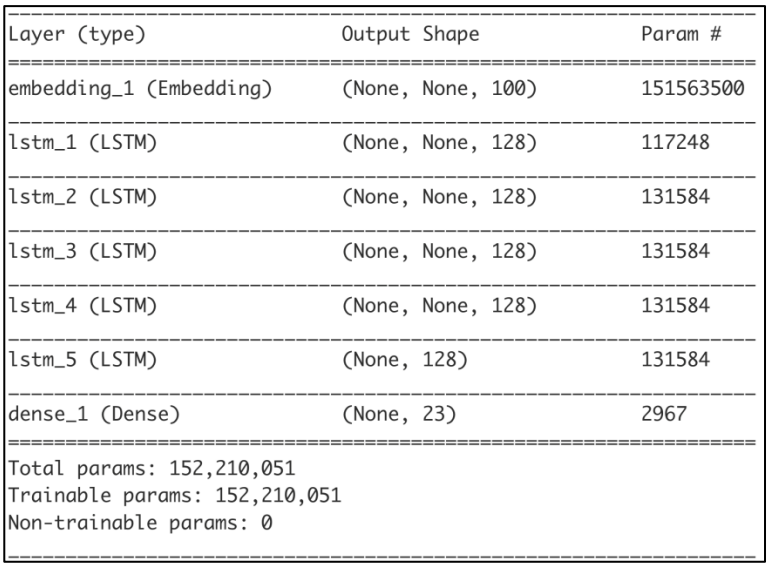
Şekil 3.8. Deney 5 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.6 Performans Analizi Deney 6

Bu deneyde kelime vektörlerinin öğrenildiği sistem üzerinde analiz yapılmıştır. Kelime vektörlerinin öğrenilmesi yüksek işlem gücü ve bellek tüketen bir işlemdir. Bu deney için kullanılan hiperparametreler şu şekildedir;

- Kelime Vektörleri öğrenilmesi gerçekleştirilmiştir.
- Öğrenilen kelime vektörü boyutu 100'dür.
- CPU testlerinde 16 ve 24 çekirdek kullanılmıştır.
- 128 Nörona sahip 5 gizli katman kullanılmıştır (Şekil 3.3).
- Batch Boyutu **500** olarak belirlenmiştir.

Test edilen derin öğrenme mimarisi Şekil 3.9'da verilmiştir.



Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 100)	151563500
lstm_1 (LSTM)	(None, None, 128)	117248
lstm_2 (LSTM)	(None, None, 128)	131584
lstm_3 (LSTM)	(None, None, 128)	131584
lstm_4 (LSTM)	(None, None, 128)	131584
lstm_5 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 23)	2967

Total params: 152,210,051
Trainable params: 152,210,051
Non-trainable params: 0

Şekil 3.9. Deney 6 Derin Öğrenme Mimarisi

Şekil 3.9'da görüleceği üzere öğrenilecek parametre sayısı 152,210,051 adettir. Öğrenme transferi kullanılmadığı için hazır kullanılan (eğitilmeyecek) parametre yoktur.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.11'de verilmiştir.

Çizelge 3.11. Deney 6 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz 16 çekirdek	1000	104 dk	6,03 dk
CPU / 2600 Mhz 24 Çekirdek	1000	85 dk	4,7 dk
GPU	1000	25,1 dk	1,55 dk

Çizelge 3.11'de verilen bilgilere göre CPU üzerinde gerçekleştirilen testlerde işlem gücünün 1,5 katına çıkması, CPU üzerinde yapılan testlerde eğitim süresinde **1,22 kat,** test süresinde ise **1,28 kat** hızlanma sağlamıştır.

GPU üzerinde gerçekleştirilen testler, 16 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla yaklaşık **4,14** kat, 24 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla **3,38** kat daha hızlı çalışmıştır.

Deney 6, aynı batch boyutu ile gerçekleştirilen Deney 5 ile kıyaslandığında kelime vektör boyutu üçte birine düşmüştür. Deney 5'te öğrenme transferi ile hazır kullanılan

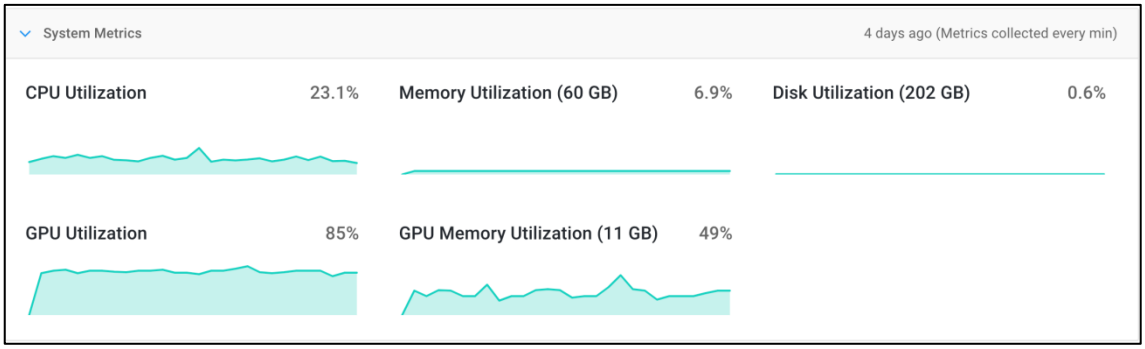
kelime vektörü boyutu 300 iken, bu deneyde kendi öğrendiğimiz kelime vektörleri 100 boyutlu olarak öğrenilmiştir. Kelime vektörü sayısının üçte birine düşmesine rağmen çalışma süreleri Deney 5'e yakın çıkmıştır. Ancak, 300 boyutunda kelime vektörlerinin öğrenilmesi için çalıştırılan testlerde GPU bellek miktarı (12 GB) yetersiz kalmış bu nedenle Bellek Hatası alınmıştır. Hata almadan kelime vektörlerinin öğrenilebilmesi için vektör boyutunun düşürülmesi gerekmiştir. Bu deneyde öğrenme transferi ile okunan kelime vektörleri boyutunca (300) öğrenme işlemi gerçekleştirilse çalışma sürelerinin daha yüksek olacağı düşünülmektedir.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.12'de verilmiştir.

Çizelge 3.12. Deney 6 Kaynak Kullanım Değerleri

Donanım	**CPU Kullanım Yüzdesi**	**GPU Kullanımı Yüzdesi**	**Bellek Kullanım Miktarı**
CPU / 2600 Mhz 16 çekirdek	%60 – % 70	--	13-15 GB Ram
CPU / 2600 Mhz 24 Çekirdek	%60 – % 70	--	13-15 GB Ram
GPU	%30 – %35	%85 – %95	6 GB Ram 3-11 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.10'de verilmiştir.



Şekil 3.10. Deney 6 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.7 Performans Analizi Deney 7

Bu deney için kullanılan hiperparametreler şunlardır.

- Kelime Vektörleri öğrenilmesi gerçekleştirilmiştir.
- Öğrenilen kelime vektörü boyutu 100.
- CPU testlerinde 16 ve 24 çekirdek kullanılmıştır.
- 128 Nörona sahip 5 gizli katman kullanılmıştır (Şekil 3.3).
- Batch Boyutu **500** olarak belirlenmiştir.

Test edilen derin öğrenme mimarisi Şekil 3.11'de verilmiştir.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 100)         151563500
_________________________________________________________________
lstm_1 (LSTM)                (None, None, 128)         117248
_________________________________________________________________
lstm_2 (LSTM)                (None, None, 128)         131584
_________________________________________________________________
lstm_3 (LSTM)                (None, None, 128)         131584
_________________________________________________________________
lstm_4 (LSTM)                (None, None, 128)         131584
_________________________________________________________________
lstm_5 (LSTM)                (None, 128)               131584
_________________________________________________________________
dense_1 (Dense)              (None, 23)                2967
=================================================================
Total params: 152,210,051
Trainable params: 152,210,051
Non-trainable params: 0
_________________________________________________________________
```

Şekil 3.11. Deney 7 Derin Öğrenme Mimarisi

Şekil 3.11'de görüleceği üzere öğrenilecek parametre sayısı 152,210,051 adettir. Öğrenme transferi kullanılmadığı için hazır kullanılan (eğitilmeyecek) parametre yoktur.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.13'de verilmiştir.

Çizelge 3.13. Deney 7 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz 16 çekirdek	500	129,35 dk	6,08 dk
CPU / 2600 Mhz 24 Çekirdek	500	106 dk	5,06 dk
GPU	500	34,1 dk	2,11 dk

Çizelge 3.13'de verilen bilgilere göre CPU üzerinde gerçekleştirilen testlerde işlem gücünün 1,5 katına çıkması, CPU üzerinde yapılan testlerde eğitim süresinde **1,22 kat,** test süresinde ise **1,20 kat** hızlanma sağlamıştır.

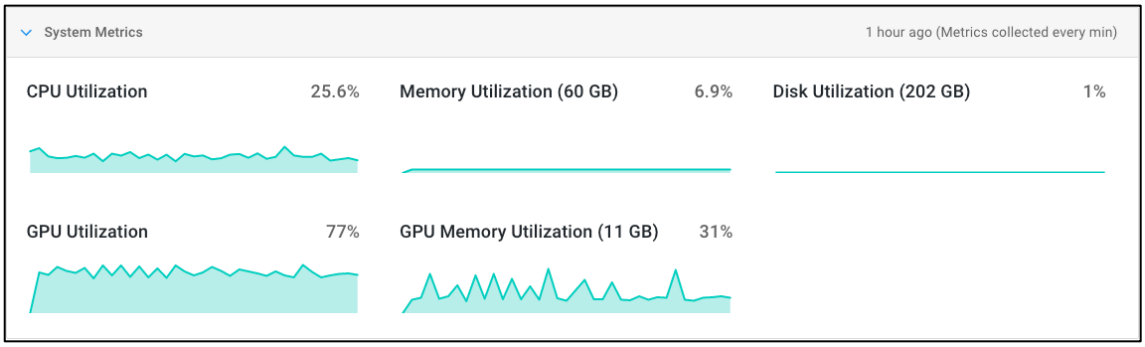
GPU üzerinde gerçekleştirilen testler, 16 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla yaklaşık **3,79** kat, 24 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla **3,10** kat daha hızlı çalışmıştır.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.14'de verilmiştir.

Çizelge 3.14. Deney 7 Kaynak Kullanım Değerleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 2600 Mhz 16 çekirdek	%60 – % 70	--	13-15 GB Ram
CPU / 2600 Mhz 24 Çekirdek	%60 – % 70	--	13-15 GB Ram
GPU	%30 – %35	%85 – %95	6GB Ram 3-11 GB vMemory

GPU üzerinde gerçekleştirilen teste ilişkin sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.12'de verilmiştir.



Şekil 3.12. Deney 7 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.8 Performans Analizi Deney 8

Bu deneyde, Deney 7'de olduğu gibi kelime vektörlerinin öğrenildiği sistem üzerinde analiz yapılacaktır. Bu deneyde öğrenilen kelime vektörü boyutu Deney 7'ye kıyasla artırılmıştır. Bu deney için kullanılan hiperparametreler şu şekildedir;

- Kelime Vektörleri öğrenilmesi gerçekleştirilmiştir.
- Öğrenilen kelime vektörü boyutu 200'dür.
- CPU testlerinde 16 ve 24 çekirdek kullanılmıştır.
- 128 Nörona sahip 5 gizli katman kullanılmıştır (Şekil 3.3).
- Batch Boyutu **500** olarak belirlenmiştir.

Test edilen derin öğrenme mimarisi Şekil 3.13'te verilmiştir.

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 200)	303127000
lstm_1 (LSTM)	(None, None, 128)	168448
lstm_2 (LSTM)	(None, None, 128)	131584
lstm_3 (LSTM)	(None, None, 128)	131584
lstm_4 (LSTM)	(None, None, 128)	131584
lstm_5 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 23)	2967

Total params: 303,824,751
Trainable params: 303,824,751
Non-trainable params: 0

Şekil 3.13. Deney 7 Derin Öğrenme Mimarisi

Şekil 3.10'da görüleceği üzere öğrenilecek parametre sayısı 303,824,751 adettir. Bu rakam Deney 6'da öğrenilen parametrelerin yaklaşık 2 katıdır. Öğrenme transferi kullanılmadığı için hazır kullanılan (eğitilmeyecek) parametre yoktur.

Test sonucu elde edilen çalışma sürelerine ilişkin bilgiler Çizelge 3.15'de verilmiştir.

Çizelge 3.15. Deney 7 Sonucu Çalışma Süreleri Karşılaştırması

Donanım	Batch Boyutu	Eğitim Süresi	Test Süresi
CPU / 2600 Mhz 16 çekirdek	500	160 dk	6,5 dk
CPU / 2600 Mhz 24 Çekirdek	500	141 dk	5,53 dk
GPU	500	54,25 dk	2,16 dk

Çizelge 3.15'de verilen bilgilere göre CPU üzerinde gerçekleştirilen testlerde işlem gücünün 1,5 katına çıkması, CPU üzerinde yapılan testlerde eğitim süresinde **1,13 kat,** test süresinde ise **1,17 kat** hızlanma sağlamıştır.

GPU üzerinde gerçekleştirilen testler, 16 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla yaklaşık **2,94** kat, 24 çekirdekli CPU üzerinde gerçekleştirilen testlere kıyasla **2,59** kat daha hızlı çalışmıştır.

Bu deney, aynı batch boyutu ile gerçekleştirilen Deney 7 ile kıyaslandığında öğrenilen kelime vektörü 100 iken 200'e çıkarılmıştır. Öğrenilmesi gereken kelime vektörü boyutunun iki katına çıkması çalışma süresini de artırmıştır. 100 boyutlu kelime vektörlerinin öğrenildiği Deney 7'de GPU üzerinde eğitim ve test süreleri **34,1 dk** ve **2,11 dk** iken 200 boyutlu kelime vektörlerinin öğrenildği bu deneyde çalışma süreleri **54,25 dk** ve **2,16 dk** sürmüştür. Bu değerlerden anlaşıldığı üzere öğrenilen kelime vektörlerin boyutunun iki katına çıkması eğitim süresinin **1,59 katına** çıkmasına neden olmuştur. Test sürelerinde kayda değer bir artış gözlenmemiştir.

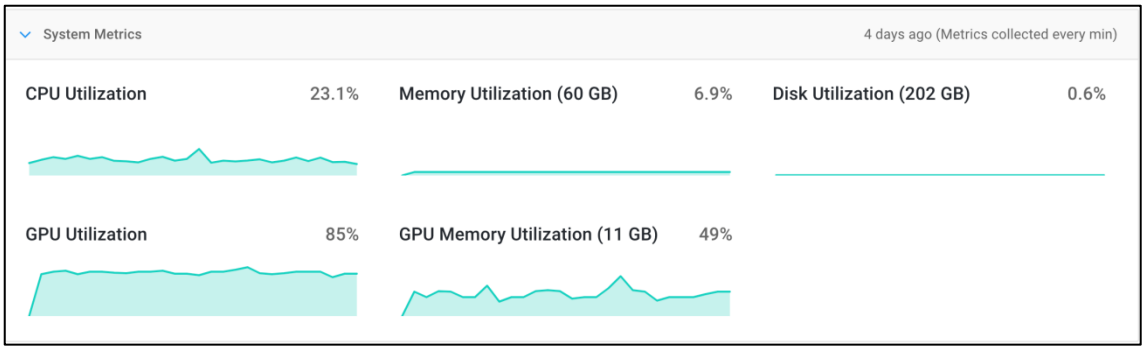
Deney 7'de 24 çekirdekli CPU üzerinde gerçekleştirilen testlerde eğitim ve test süreleri **106 dk – 5,06 dk** iken, öğrenilen kelime vektör boyutu iki katına çıkarılan bu deneyde eğitim ve test süreleri **141 dk – 5,53 dk** olmuştur. Öğrenilen kelime vektörlerinin iki katına çıkması CPU üzerinde gerçekleştirilen testlerde de **1,33 kat** yavaşlamaya neden olmuştur.

Testler esnasında kullanılan sistem kaynaklarına ilişkin bilgiler Çizelge 3.16'da verilmiştir.

Çizelge 3.16. Deney 7 Kaynak Kullanım Değerleri

Donanım	CPU Kullanım Yüzdesi	GPU Kullanımı Yüzdesi	Bellek Kullanım Miktarı
CPU / 2600 Mhz 16 çekirdek	%60 – % 70	--	13-15 GB Ram
CPU / 2600 Mhz 24 Çekirdek	%60 – % 70	--	13-15 GB Ram
GPU	%30 – %35	%85 – %95	6GB Ram 3-11 GB vMemory

GPU üzerinde gerçekleştirilen teste ait sistem kaynaklarının kullanımına ilişkin detaylı bilgiler Şekil 3.14'te verilmiştir.



Şekil 3.14. Deney 7 GPU Üzerinde Sistem Kaynakları Tüketimi

3.1.9 Performans Analizi Test Süreleri Karşılaştırmaları

Deney 1-7 arası gerçekleştirilen testlerde sistem eğitim aşamasını takiben test aşaması çalıştırılmıştır. Test aşamasında veri örnekleri vektörize edilmiş halleri üzerinden yapılmaktadır. Ancak, web sayfaları kategorize edilirken önce bazı ön işlemlerin uygulanması, vektörize edilmesi ve ardından kategorize edilmesi gerekmektedir. Bu nedenle eğitilmiş sistem üzerinde gerçekleştirilen test süreleri Deney 1-7'de elde edilen sonuçlara kıyasla farklı olabilmektedir. Bu nedenle eğitilmiş sistem üzerinden sadece test işlemlerinin çalıştırıldığı durumlar ayrıca incelenmiştir.

Çizelge 3.17'de GPU ve CPU üzerinde 887,195 adet web sayfası, başlık, açıklama ve anahtar kelime bilgilerine göre kategorize edilmesine ilişkin çalışma süreleri verilmiştir. Testler, tek katmanlı öğrenilmiş mimari ve 5 katmanlı öğrenilmiş mimari üzerinde

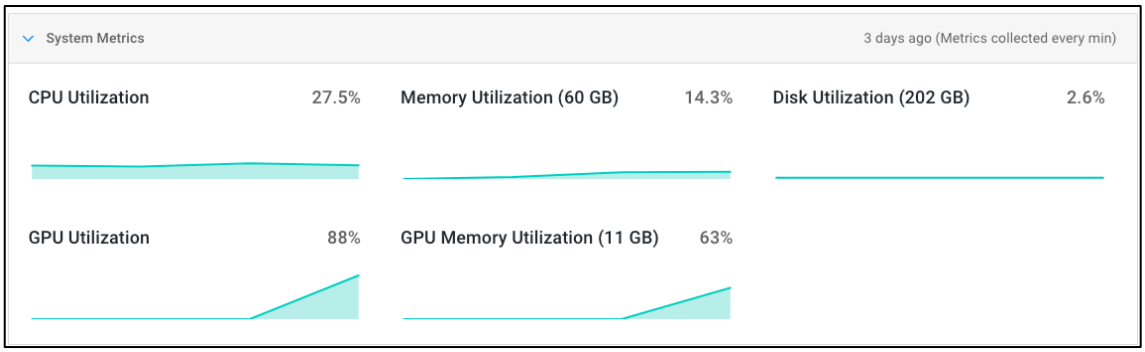
uygulanmıştır. Ayrıca, farklı batch boyutunun çalışma süresine etkisi de incelenmiştir. Test edilen tek katmanlı mimari Şekil 3.2'de, 5 katmanlı mimari ise Şekil 3.3'te verilmiştir.

Çizelge 3.17. Test Süreleri Karşılaştırması

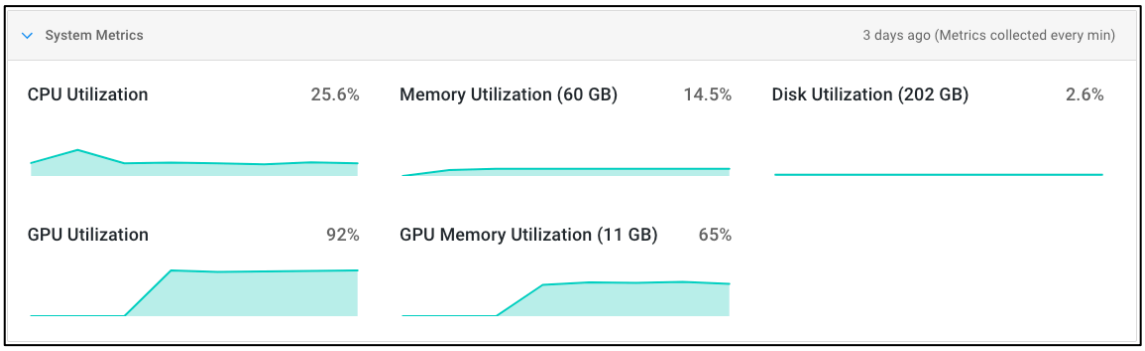
Test Durumu	**CPU (24 Çekirdek)**		**GPU**	
	Vektörizasyon	**Kategorizasyon**	**Vektörizasyon**	**Kategorizasyon**
1 Katman, batch: 5000	0,87 dk	7 dk	1,0 dk	1,25dk
5 Katman, batch: 5000	0,92 dk	28 dk	1,0 dk	5,41 dk
5 Katman, batch: 1000	1,03 dk	27 dk	1,0 dk	7,8 dk

Çizelge 3.15'teki verilere göre katman sayısının birden beşe çıkarılması CPU üzerinde gerçekleştirilen testlerde **4 kat**, GPU üzerinde gerçekleştirilen testlerde ise **4,32 kat** yavaşlamaya neden olmaktadır. Batch boyutunun 5000 yerine 1000 olarak ayarlandığı testte ise CPU üzerinde gerçekleştirilen testlerde kayda değer bir yavaşlama olmadığı, GPU üzerinde gerçekleştirilen testlerde ise 1,44 kat yavaşlamaya neden olmuştur. Vektörizasyon işlemi derin öğrenme mimarisinden bağımsız bir işlem adımı olduğu için tüm testlerde vektörizasyon süreleri birbirlerine yakın sürelerde gerçekleşmiştir.

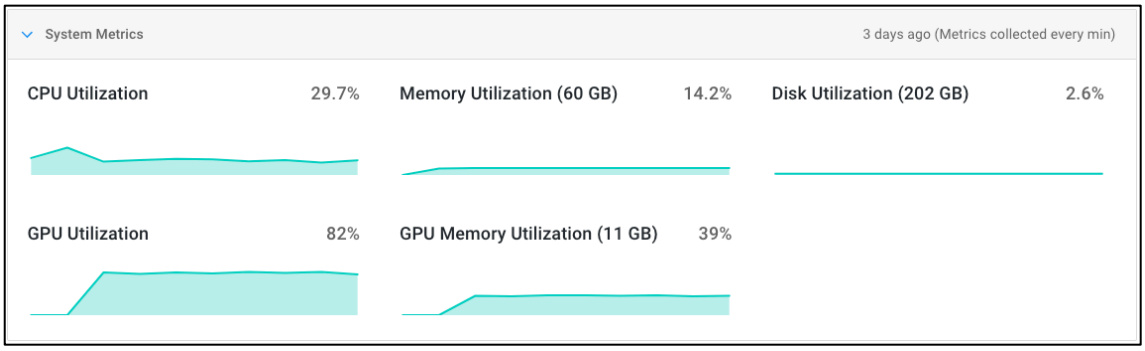
Karşılaştırılan 3 farklı test için CPU üzerinde gerçekleştirilen testlerde yaklaşık 10-12 GB Ram kullanımı olduğu, kullanılan GPU belleği kullanımının ise %50-%60 civarlarında olduğu görülmüştür. CPU kullanımının daha fazla artmamasının nedeni testler esnasında kullanılabilecek çalışma frekansına 37500 Mhz limiti konmasıdır. GPU üzerinde kaynak tüketimleri Şekil 3.15, Şekil 3.16, Şekil 3.17'de verilmiştir.



Şekil 3.15. Katman Sayısı 1, Batch Boyutu 5000 Olan Testte GPU Kaynakları



Şekil 3.16. Katman Sayısı 5, Batch Boyutu 5000 olan Testte GPU Kaynakları



Şekil 3.17. Katman Sayısı 5, Batch Boyutu 1000 olan Testte GPU Kaynakları

3.1.10 Performans Analizi Değerlendirmesi

Bu çalışmada derin öğrenme uygulamalarında GPU ve CPU arasında karşılaştırma yapılmıştır. Karşılaştırma işlemi için birçok farklı durum test edilmiş ve çalışma kapsamında 4 farklı donanım senaryosu test edilmiştir. Test edilen senaryolarda CPU'üzerindeki çekirdek sayısı, bir çekirdeğin sahip olduğu çalışma frekansının değişimi gibi durumlar test edilmiştir. Detaylı deney sonuçları Bölüm 3.1.1 – 3.1.9'da verilmiştir. Bu başlık altında yer alan bilgiler özet bilgilerdir.

CPU'lar için test edilen 3 farklı donanım senaryosu şu şekildedir.

- 16 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 25000 Mhz. Her bir çekirdek için çalışma frekansı: 1300 Mhz
- 16 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 25000 Mhz. Her bir çekirdek için çalışma frekansı: 2600 Mhz
- 24 çekirdekli CPU testleri için maksimum çalışma frekansı limiti: 37500 Mhz. Her bir çekirdek için çalışma frekansı: 2600 Mhz

- 2496 çekirdekli GPU. Her çekirdeğin çalışma frekansı 562 Mhz.

3.1.10.1 **CPU Karşılaştırması Değerlendirmesi**

Test edilen ilk durumda çekirdek çalışma frekansının çalışma süresine etkisi incelenmiştir. Performans değerlendirmesi için çalışma sürelerine ilişkin bilgilerle birlikte hızlanma/yavaşlama miktarları da verilmiştir. Çekirdek çalışma frekansı için gerçekleştirilen testler 16 çekirdekli CPU üzerinde 2 farklı batch boyutu üzerinde gerçekleştirilmiştir. Gerçekleştirilen bu test tek katmanlı mimari üzerinde test edilmiştir. Elde edilen sonuçlar Çizelge 3.18'de verilmiştir. Çizelgedeki veriler sistemin eğitim aşaması için hesaplanan sürelerdir.

Çizelge 3.18. Çekirdek Çalışma Frekansı ve Batch Boyutu Değerlendirmesi

16 Çekirdekli CPU	**1300 Mhz**	**2600 Mhz**	**Hızlanma Oranı**
Batch: 64	71,3 dk	36 dk	~2 kat
Batch: 500	44,4 dk	21,3 dk	~2 kat

Elde edilen sonuçlara göre çekirdek çalışma frakansının iki katına çıkması sistem çalışma süresini yarıya düşürmektedir. Batch boyunun artırılması da çalışma süresini azaltmaktadır.

Gerçekleştirilen ikinci testte çekirdek çalışma frekansı 2600 Mhz olan 16 ve 24 çekirdekli CPU durumları test edilmiştir. İki durum arasında çekirdek sayısı ve maksimum çalışma frekansı değerleri 1,5 katına çıkarılmıştır. Testler esnasında 5 katmanlı derin öğrenme mimarisi denenmiştir. Testler esnasında öğrenme transferi kullanılmamış kelime vektörleri sistem tarafından öğrenilmiştir. Batch boyutu 500 olarak belirlenmiştir.

Çizelge 3.19. Farklı Çekirdek Sayılarına Sahip CPU'ların Değerlendirilmesi

Kelime Vektör Boyutu	**16 Çekirdekli**	**24 Çekirdekli**	**Hızlanma Oranı**
100	129, 35 dk	106 dk	~1,22 kat
200	160 dk	141 dk	~1,13 kat

Çekirdek sayısının artırılmasının çalışma sürelerini azalttığı görülmüştür.

3.1.10.2 **CPU vs GPU Karşılaştırmaları**

CPU ve GPU karşılaştırmaları için gerçekleştirilen ilk testte kelime vektörlerinin hazır kullanılmadığı, sistem tarafından öğrenildiği durum incelenmiştir. Bu testte kullanılan hiperparametreler ile Bölüm 3.1.20.1'deki CPU karşılaştırmalarındaki çekirdek sayısının test edildiği durumdaki hiperparametreler aynıdır. 100 ve 200 uzunluğundaki kelime vektörleri test edilmiştir. Kullanılan batch boyutu da 500'dür. Elde edilen çalışma süreleri Çizelge 3.20'de verilmiştir. Testler 5 katmanlı derin öğrenme mimarisinde çalıştırılmıştır.

Çizelge 3.20. CPU ve GPU Karşılaştırması 1

Kelime Vektör Boyutu	**CPU** 24 çekirdek – 2600 Mhz	**GPU**	**Hızlanma Oranı**
100	106 dk	34,1 dk	~3,10 Kat
200	141 dk	54,25 dk	~2,60 Kat

Çizelge 3.20'de elde edilen verilere göre kelime vektörü boyutunun artması çalışma sürelerinin artmasına neden olurken, testlerin GPU üzerinde gerçekleştirilmesi çalışma sürelerinde azalmaya neden olmuştur.

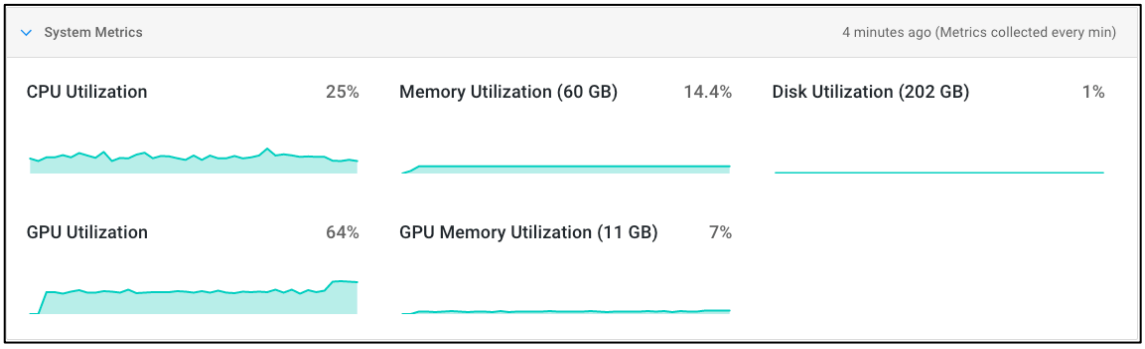
Test edilen bir diğer durumda batch boyutunun çalışma süreleri üzerinde etkisi incelenmiştir. Bu testler tek katmanlı mimaride öğrenme transferi kullanılarak çalıştırılmıştır.

Çizelge 3.21. Farklı Batch Boyutlarının Değerlendirilmesi

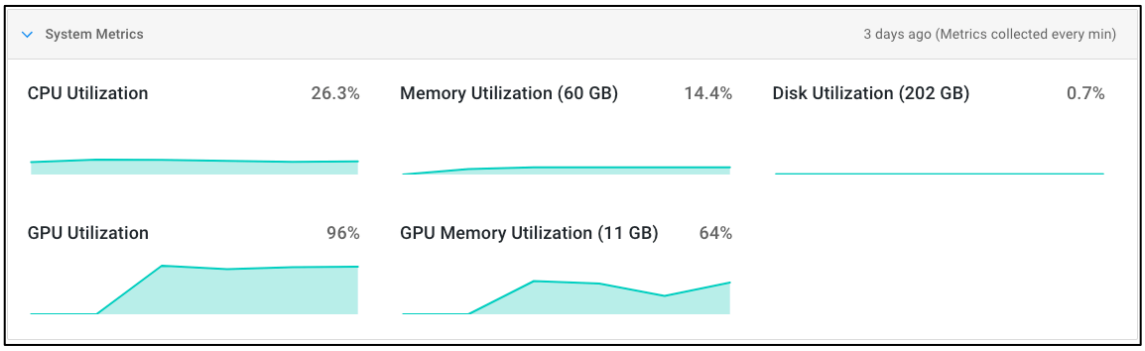
Batch Boyutu	**CPU** 16 çekirdek – 2600 Mhz	**GPU**	**Hızlanma Oranı**
64	36 dk	37,3 dk	yok
500	21,3 dk	6,4 dk	~3,32 Kat
1000	20,61 dk	4,6 dk	~4,48 Kat
5000	20,86 dk	3,8 dk	~5,49 Kat

Batch boyutunun artması matematiksel işlemlerin daha yüksek verim ile paralelleştirilmesine olanak tanımaktadır. Bu nedenle Çizelge 3.21'de elde edilen sonuçlara göre batch saysının artması CPU üzerinde ciddi hızlanmalara neden olmazken, GPU üzerindeki testlerin hızlanmasına neden olmaktadır. Batch sayısının daha fazla artırılması GPU üzerinde sistemin daha hızlı çalışmasını sağlayacaktır. Ancak, batch boyutunun artması kullanılan bellek miktarını da artırmaktadır. Kiralanan GPU sunucumuzun bellek kapasitesi 12 GB'tır. Batch boyutu 10000 olan durum denenmek istendiğinde memory hatası alınmıştır.

Batch boyutu 64 iken GPU testlerinde tüketilen sistem kaynakları Şekil 3.18'de, batch boyutu 5000 iken, GPU testlerinde tüketilen sistem kaynakları ise Şekil 3.19'da verilmiştir.



Şekil 3.18. Batch boyutu 64 iken GPU Testindeki Kaynak Tüketimleri



Şekil 3.19. Batch Boyutu 5000 iken GPU testinde Tüketilen Sistem Kaynakları

Katman sayısını arttığı durumlarda öğrenilecek parametre sayıları da artmaktadır. Öğrenilecek parametre sayılarının artması çalışma sürelerininde artmasına neden olmaktadır. Gerçekleştirilen bir diğer testte artan katman sayısı ve parametrelerin çalışma süresine etkisi incelenmiştir. Testler esnasında öğrenme transferi kullanılmıştır. Karşılaştırma işlemi Şekil 3.2'de verilen tek katmanlı mimari ile Şekil 3.3'te verilen 5

katmanlı mimari arasında yapılmıştır. Katman sayısı artırılacağı zaman her katmanda yer alacak nöron sayısı ayrı ayrı belirlenir. Test edilen 5 katmanlı mimaride her katmandaki nöron sayısı tek katmanlı mimarideki nöron sayısı ile aynı olarak belirlenmiştir. Bu sayede artan katman sayısı ile parametre sayısı da artmıştır. Test sonucunda hesaplanan çalışma süreleri Çizelge 3.22'de verilmiştir. Testte kullanılan batch boyutu 1000'dir.

Çizelge 3.22. Artan Katman Sayısının Çalışma Sürelerine Etkisi

Katman Sayısı	**CPU** 16 çekirdek – 2600 Mhz	**GPU**	**Hızlanma Oranı** (GPU ile)
1	20,61 dk	4,6 dk	~4,48 Kat
5	83,8 dk	26,7 dk	~3,13 Kat
Yavaşlama (Katman sayısı sonucu)	~4,06 Kat	~5,8 Kat	--

Sistem geliştirilmesi aşamasında eğitim süreleri çok önemli iken gerçek zamanlı çalışan bir sistem için test süreleri de oldukça önemlidir. Çizelge 3.23'te 1,210,967 adet web sayfasının sınıflandırılması için geçen test süreleri verilmiştir.

Çizelge 3.23. Test Süreleri Karşılaştırması

Katman/batch boyutu	**CPU** 24 çekirdek – 2600 Mhz	**GPU**	**Hızlanma Oranı**
1 katman / batch: 5000	7 dk	1,25 dk	~5,6 Kat
5 katman / batch: 5000	28 dk	5,41 dk	~5,17 Kat
5 katman / batch: 1000	27 dk	7,8 dk	~3,46 Kat

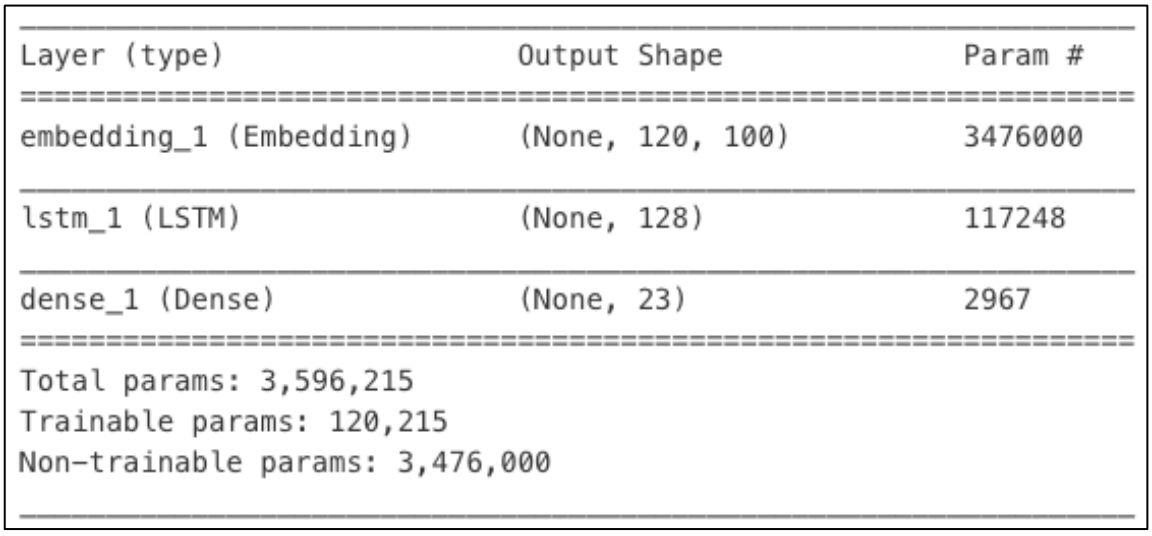
Batch sayısının artması test sürelerinin kısalmasına neden olmaktadır. GPU kapasitemiz artırıldığında tüm eğitim ve test süreleri kısaltılabilecektir.

3.1.10.3 **Öğrenme Transferi Karşılaştırması**

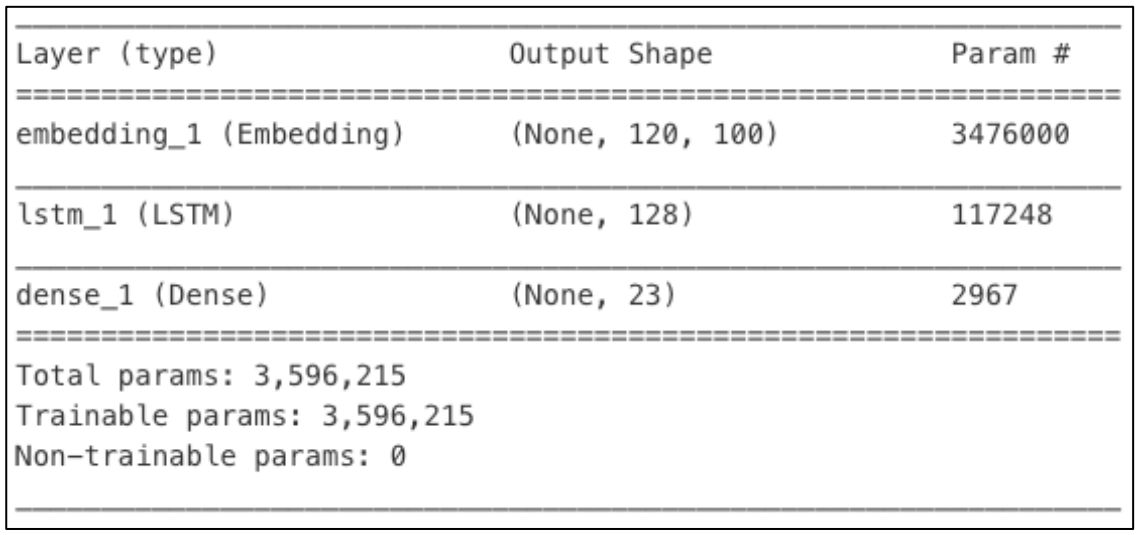
Bu çalışmada öğrenme Transferinin kullanıldığı ve kullanılmadığı durumar için çalışma süreleri ve başarı değerleri karşılaştırılmıştır.

Öğrenme transferinin kullanılması öğrenilen parametre sayısı azaltmaktadır. Karşılaştırma yapılabilmesi için gerçekleştirilen testler tek katmanlı YiSA mimarisi ile

denenmiştir. Öğrenme transferi kullanılan teste ilişkin model yapısı Şekil 3.20'de öğrenme transferi kullanılmadan gerçekleştirilen teste ilişkin model yapısı Şekil 3.21'de verilmiştir. Her iki test için kullanılan sözlük boyutu ve sözlük içerisindeki kelimeler aynıdır.

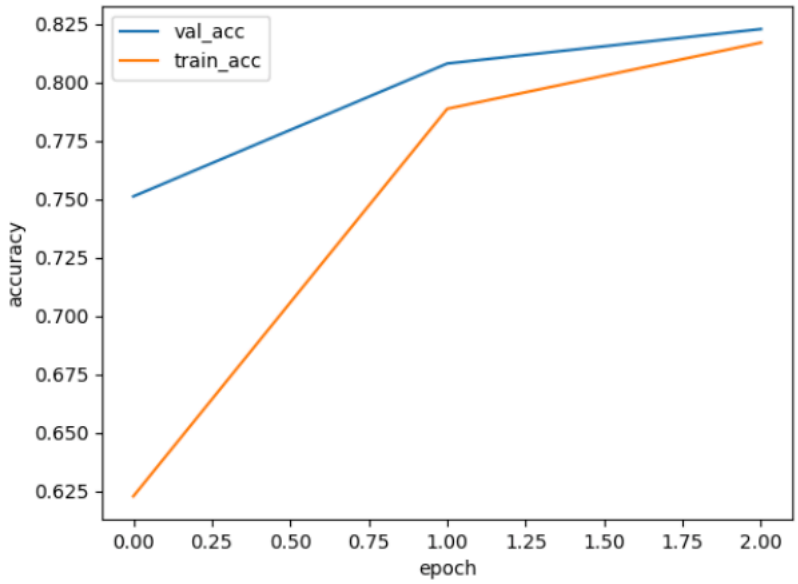


Şekil 3.20. Öğrenme Transferi Kullanılan Test için Model Yapısı

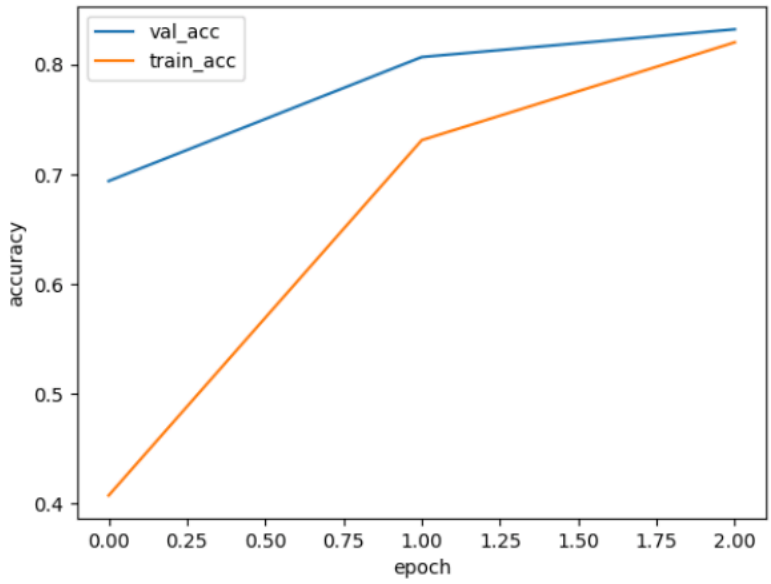


Şekil 3.21. Öğrenme Transferi Kullanılmayan Test için Model Yapısı

Gerçekleştirilen testlere ilişkin validasyon başarı oranı (val_acc) ve eğitim başarı oranı (train_acc) değerleri Şekil 3.22 ve Şekil 3.23'de verilmiştir.



Şekil 3.22. Öğrenme Transferi Kullanılan Teste İlişkin Başarı Değerleri



Şekil 3.23. Öğrenme Transferi Kullanılmayan Teste İlişkin Başarı Değerleri

Elde edilen sonuçlara göre transferi kullanılan test ile kullanılmayan testler arasında başarı oranlarında ciddi bir farklılık gözlenmemiştir. Gerçekleştirilen testler 3 epoch boyunca çalıştırılmış olup her epoch için çalışma süreleri ortalamaları şu şekildedir;

- Öğrenme transferi kullanılan test: 597,9 sn
- Öğrenme Transferi kullanılmayan test: 712 sn

Sonuç olarak öğrenme transferi kullanmak başarı oranında ciddi bir etki oluşturmazken çalışma süresinin kısalması yönünde bir etki oluşturmuştur.

3.2 Web Sayfalarının Sınıflandırılması için Model Denemeleri Sonuçları

Bir web sayfasına ilişkin işlenecek veri 3 türdür. Bunlar;

- Başlık
- Açıklama
- Anahtar Kelimeler

Metinsel içerik bilgisi diğer 3 veri türüne kıyasla oldukça fazla kelime içermektedir. Sistem eğitim aşamasında öncelikle başlık, anahtar kelime ve açıklama bilgilerini kullanacaktır.

Bir web sayfası işlenirken veriler birleştirilerek derin öğrenme mimarisine girdi olarak verilmiştir. Verilerin eklenme sırası başlık, açıklama, anahtar kelime şeklindedir.

Gerçekleştirilen testler, üzerinde CPU donanımı bulunan bir sunucu üzerinde gerçekleştirilmiştir. Sunucu üzerinde bulunan donanımsal özellikler şunlardır;

- 36 Çekirdek Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz
- 128 GB Ram
- 1 TB SSD Disk

Gerçekleştirilen testlerde iki farklı derin öğrenme mimarisi test edilmiştir. Test edilen mimarilerden birisi YiSA, diğeri ise İleri Beslemeli Yapay Sinir Ağıdır (İBYSA). Gerçekleştirilen testler her iki mimaride ayrı ayrı gerçekleştirilmiştir. Her mimari için farklı katman sayıları oluşturulmuş modeller test edilmiştir. Bunların yanı sıra tüm testler Öğrenme Transferi kullanılarak ve kullanılmadan gerçekleştirilmiştir. Analiz edilen durumlar 4 farklı başlıkta değerlendirilebilir.

- Farklı katman sayılarına sahip mimarilerin denenmesi: 1, 3, 5 katmanlı
- Farklı derin öğrenme mimarilerinin denenmesi: YiSA, İBYSA
- Öğrenme Transferinin denenmesi

- Geleneksel yöntemlerin derin öğrenme yöntemleri ile karşılaştırılması

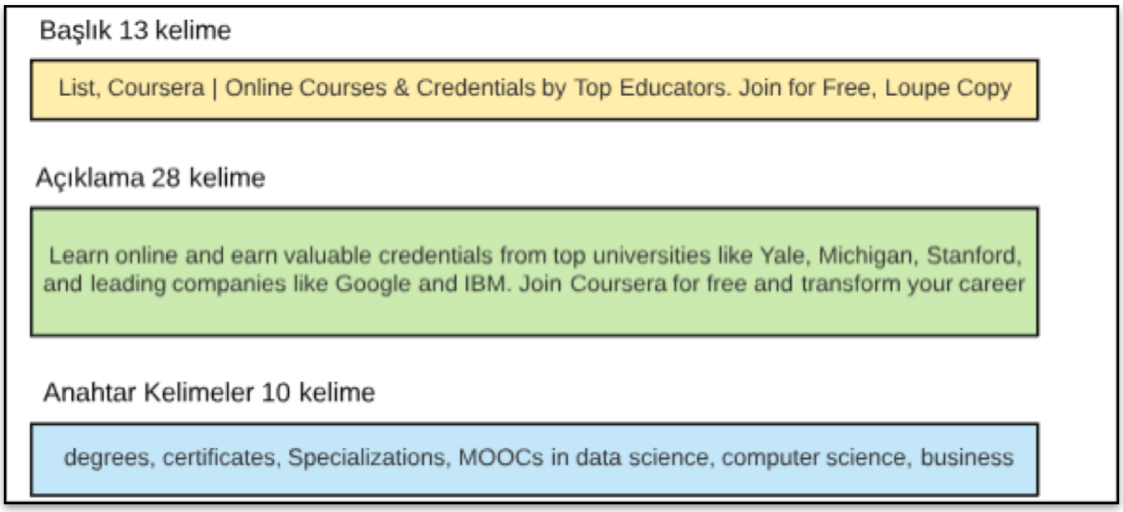
Performans analizi için başarı değerlerinin yanı sıra çalışma sürelerinin karşılaştırılması da oldukça önemlidir. Gerçekleştirilen testlerde başarı değerleri ve çalışma süreleri kıyaslanmış ve değerlendirmelerde bulunulmuştur.

Metinsel verilerin işlenebilmesi için algoritmalar tarafından anlamlı bir şekilde vektörize edilmesi gerekmektedir. Bu çalışmada her metin alanının birbirinden farklı ağırlıklara sahip olabileceği gözetilerek algoritmanın bu ağırlıklandırmayı yapabileceği bir vektörizasyon yöntemi uygulanmaya çalışılmıştır.

Vektörizasyon işlemi metnin belli bir standarta uydurulmasını sağlayan bir işlemdir. Metinler, veri örneğinden veri örneğine boyutun değişebildiği bir veri türüdür. Bu nedenle metinsel verilerde boyutun standartlaştırılması gerekmektedir. Metin alanlarının vektörize edilip kullanılabilmeleri için bir limit (n kelime) belirlenir. Metin bu limitten daha uzunsa ilk n adet kelime alınır gerisi atılır. Metin n kelimeden daha az kelime içeriyorsa n adet kelimeye tamamlanacak şekilde metne ekleme yapılır. Bu çalışmada her metin alanı için farklı bir limit belirlenmiştir. Bu limitler şu şekildedir;

- Başlık: 20 kelime
- Açıklama: 50 kelime
- Anahtar kelimeler: 50 kelime

Bu değerler, veri kümesinin incelenmesi sonucu belirlenmiştir. Metin alanları bu şekilde sınırlandırıldıktan sonra uç uca eklenerek girdi vektörü oluşturulur. Girdi metnine ilişkin örnek Şekil 3.24'te, oluşturulan girdi vektörüne ilişkin temsili görsel ise Şekil 3.25'te verilmiştir. Şekilde yer alan örnekte boş kalan kısımların n kelimeye tamamlanması için 0 numaralı kelime ekleme ile yapılmıştır. Kelime numaraları temsilidir.



Şekil 3.24. Veri Örneği

Şekil 3.25. Vektör Örneği

Oluşturulan bu vektör derin öğrenme mimarilerine girdi olarak verilmektedir. Vektör oluşturulması aşamasından önce durak kelimeleri (stopwords) metinden temizlenmiştir. 3 farklı durumun analiz edilmesi için toplamda 12 adet test gerçekleştirilmiştir. Gerçekleştirilen testlere ilişkin özet ve testlerin kısaltmaları şu şekildedir. İlerleyen açıklamalarda ve grafiklerde bu kısaltmalar kullanılmıştır.

- ibysa_1_ot: Tek katmanlı İBYSA, öğrenme transferi kullanıldı.
- ibysa_3_ot: 3 katmanlı İBYSA, öğrenme transferi kullanıldı.
- ibysa_5_ot: 5 katmanlı İBYSA, öğrenme transferi kullanıldı.
- yisa_1_ot: Tek katmanlı YiSA, öğrenme transferi kullanıldı.
- yisa_3_ot: 3 katmanlı YiSA, öğrenme transferi kullanıldı.
- yisa_5_ot: 5 katmanlı YiSA, öğrenme transferi kullanıldı.
- ibysa_1_n-ot: Tek katmanlı İBYSA, öğrenme transferi kullanılmadı.

- ibysa_3_n-ot: 3 katmanlı İBYSA, öğrenme transferi kullanılmadı.
- ibysa_5_n-ot: 5 katmanlı İBYSA, öğrenme transferi kullanılmadı.
- yisa_1_n-ot: Tek katmanlı YiSA, öğrenme transferi kullanılmadı.
- yisa_3_n-ot: 3 katmanlı YiSA, öğrenme transferi kullanılmadı.
- yisa_5_n-ot: 5 katmanlı YiSA, öğrenme transferi kullanılmadı.

Tüm testlerde ortak kullanılan hiperparametreler şunlardır;

- Kelime Vektör Boyutu: 100
- Batch Boyutu: 10,000
- Epoch Sayısı: 10
- Test Seti Bölme Oranı: 0.2, Toplam Örnek Sayısı: 887,184 Eğitim Örneği Sayısı: 709,738, Test Örneği Sayısı: 177,446
- Kullanılan Öğrenme Transferi Kaynağı: Glove6B100d [59]
- Girdi Vektörü Boyutu: 120
- Sözlükte bulunan kelime sayısı: 34,759
- Kayıp (Loss) Fonksiyonu: Categorical_Crossentropy
- Optimizer: RMSProp, lr:0.01, rho:0.9, decay: 0
- Ağırlık ilklendirme yöntemi: Rastgele

Bu çalışma kapsamında derin öğrenme tabanlı İBYSA ve YiSA mimarileri test edilmiş ve kıyaslanmıştır. Bu mimariler, yapıları gereği farklı sayıda parametrenin öğrenilmesi ile çalışırlar. Kullanılan diğer hiperparametreler için Keras kütüphanesindeki [79] varsayılan parametreler kullanılmıştır.

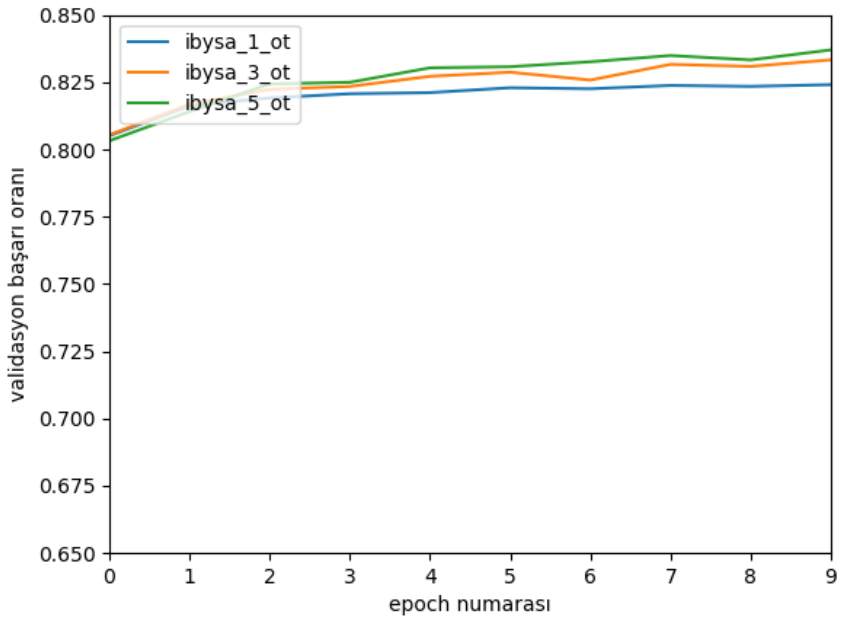
Katman sayılarının başarı değerlerine etkisi incelenirken gerçekleştirilen testlerde öğrenme transferi kullanılmıştır. Öğrenme transferinin kullanılması, eğitim aşamasında öğrenilmesi gereken parametre sayısını azaltmıştır. Çizelge 3.24 her iki mimari için toplam ve eğitim esnasında öğrenilmesi gereken parameter sayılarını göstermektedir. YiSA ve İBYSA mimarilerinde bir parametrenin öğrenilmesi için yapılması gereken

hesaplama miktarları aynı değildir. YiSA'da bir parametrenin öğrenilmesi için daha fazla işlem yapılması gerekirken İBYSA'da daha az hesaplama ile parametreler öğrenilir. Bu nedenle parametre sayıları ile çalışma süreleri arasında doğrusal bir ilişki bulunmamaktadır.

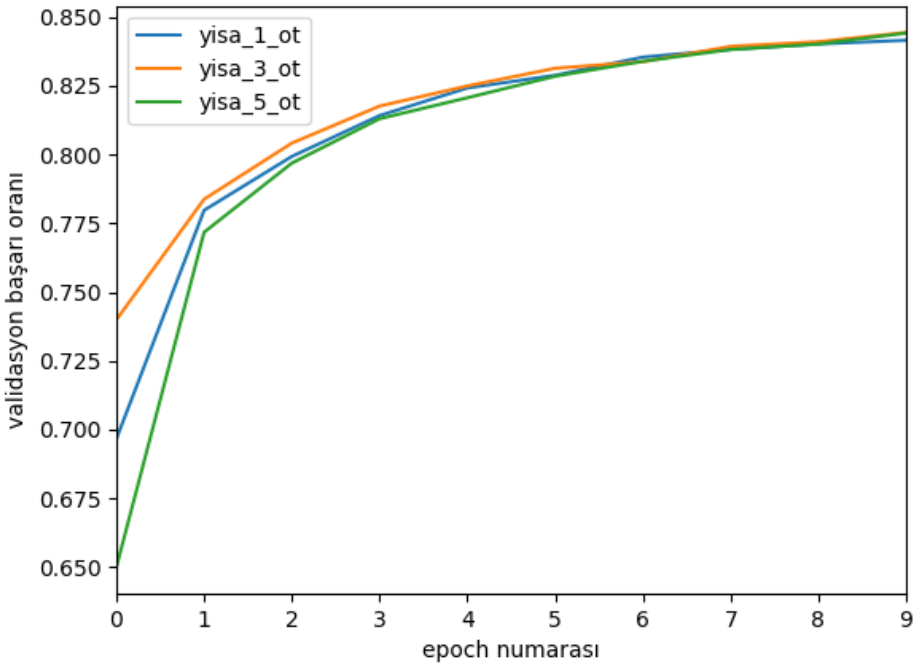
Çizelge 3.24. Parametre Sayıları

Katman Sayısı	**İBYSA**		**YiSA**	
	Toplam Parametre Sayısı	Öğrenilen Parametre Sayısı	Toplam Parametre Sayısı	Öğrenilen Parametre Sayısı
1	3,842,231	366,231	3,596,215	120,215
3	3,875,255	399,255	3,859,383	383,383
5	3,908,279	432,279	4,122,551	646,551

Katman sayısının fazla olması öğrenilecek parametre sayısını artırmaktadır. İBYSA için farklı katman sayılarına yönelik gerçekleştirilen validasyon başarı oranları (Validation Accuracy) değerleri Şekil 3.26'da verilmiştir. YiSA için farklı katman sayılarına ilişkin yapılan testlerde elde edilen validasyon başarı oranları Şekil 3.27'de verilmiştir.



Şekil 3.26. Farklı Katman Sayıları için İBYSA Sonuçları



Şekil 3.27. Farklı Katman Sayıları için YiSA Sonuçları

Şekil 3.26 ve Şekil 3.27'de görüldüğü üzere katman sayısının artması başarı değerlerinde ciddi artışlara neden olmamıştır. Elde edilen değerlere göre katman sayısının artırılmasının başarı değerlerine olumlu yönde ciddi bir artışa neden olmadığı görülmektedir. Bu nedenle problemin çözümü için az katmanlı mimarilerin tercih edilmesinin web sayfası sınıflandırma probleminin çözümü için yeterli olduğu düşünülmektedir. Az katmanlı mimarilerin kullanılmasının performansa da olumlu etkileri olması beklenmektedir.

YiSA mimarisinde başarı değerinin yakınsaması için İBYSA'na kıyasla daha fazla sayıda epocha ihtiyaç duyulmuştur. Elde edilen sonuçlara göre YiSA mimarisinde web sayfası sınıflandırması probleminin öğrenilmesi İBYSA'ya kıyasla daha yavaş gerçekleşmiştir. Şekil 3.26 ve Şekil 3.27'de görüldüğü üzere validasyon başarı oranı her iki mimari de %85 başarı değerlerine yakınsamıştır. Bu sonuçlara göre başarı değerleri kıyaslandığında web sayfası sınıflandırma problemi için test edilen bir mimarinin diğerinden daha başarılı olduğunun söylenmesi mümkün değildir.

Validasyon başarı değerleri test edilen modellerin analiz edilmesi için önemli değerlerdir. Ancak, model hakkında daha fazla değerlendirmede bulunabilmesi için

başka metriklerin de değerlendirilmesi gerekmektedir. Validasyon başarı değerleri (validation accuracy) ile birlikte eğitim başarı değerlerinin (train accuracy) birlikte analiz edilmesi sistemin Ezberleme (Overfitting) yapıp yapmadığının anlaşılmasını sağlar. Gerçekleştirilen testler için validasyon başarı oranı ile birlikte eğitim başarı oranının birlikte verildiği çıktılar Şekil 3.28'de verilmiştir. Validasyon başarı oranı ile eğitim başarı oranı arasında bir miktar fark olması normal karşılanabilir. Ancak, farkın yüksek olması sistemin ezberleme yaptığının göstergesi olabilir. Şekil 3.28'te YiSA mimarisinin ezberlemeye karşı İBYSA'ya kıyasla daha dayanıklı olduğu görülmektedir.

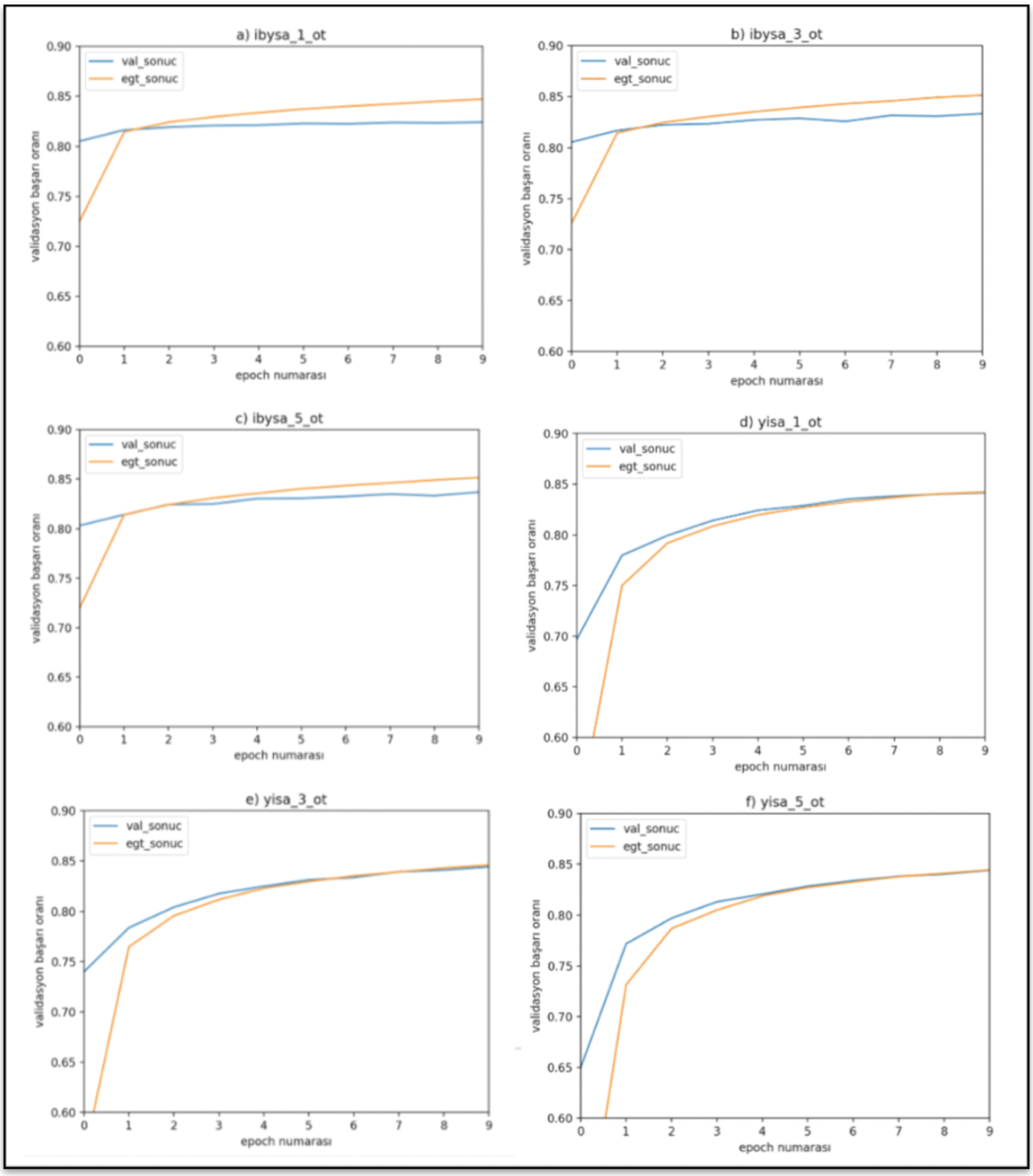
Gerçekleştirilen testlerin son epochlarında elde edilen validasyon başarı oranları ve eğitim başarı oranları Çizelge 3.25'de verilmiştir.

Çizelge 3.25. Testler için Başarı Oranları

Test İsmi	**Validasyon Başarı Oranı**	**Eğitim Başarı Oranı**
ibysa_1_ot	0.8240	0.8470
ibysa_3_ot	0.8332	0.8516
ibysa_5_ot	0.8396	0.8516
yisa_1_ot	0.8415	0.8422
yisa_3_ot	0.8343	0.8459
yisa_5_ot	**0.8441**	**0.8444**

Geliştirilen sistemde 23 farklı kategori yer almaktadır. Kategorilerde yer alan veri örneği sayıları çok değişkendir. Veri kümesi dengesiz bir sınıf dağılımına sahiptir. Bu nedenle sonuçların sadece başarı oranlarına (accuracy) dayanarak yapılması yeterli olmamaktadır. Her sınıf için ayrı ayrı değerlendirmelerde bulunulması daha doğru olacaktır. Sınıfların başarı durumlarının değerlendirilebilmesi için tutturma (Precision), hassaslık (Recal) ve f-skor (f-measure) değerleri kullanılmıştır (bkz. Bölüm 2.6). Çizelge 3.25'te yer alan testler içerisinde validasyon başarı oranına göre en başarılı test "*yisa_5_ot*"'dır. Bu teste ilişkin her sınıf için tutturma, hassaslık ve f-skor değerleri

Çizelge 3.26'da verilmiştir. Testlere ilişkin eğitim ve validasyon sonuçlarına ilişkin grafikler Şekil 3.28'de verilmiştir.



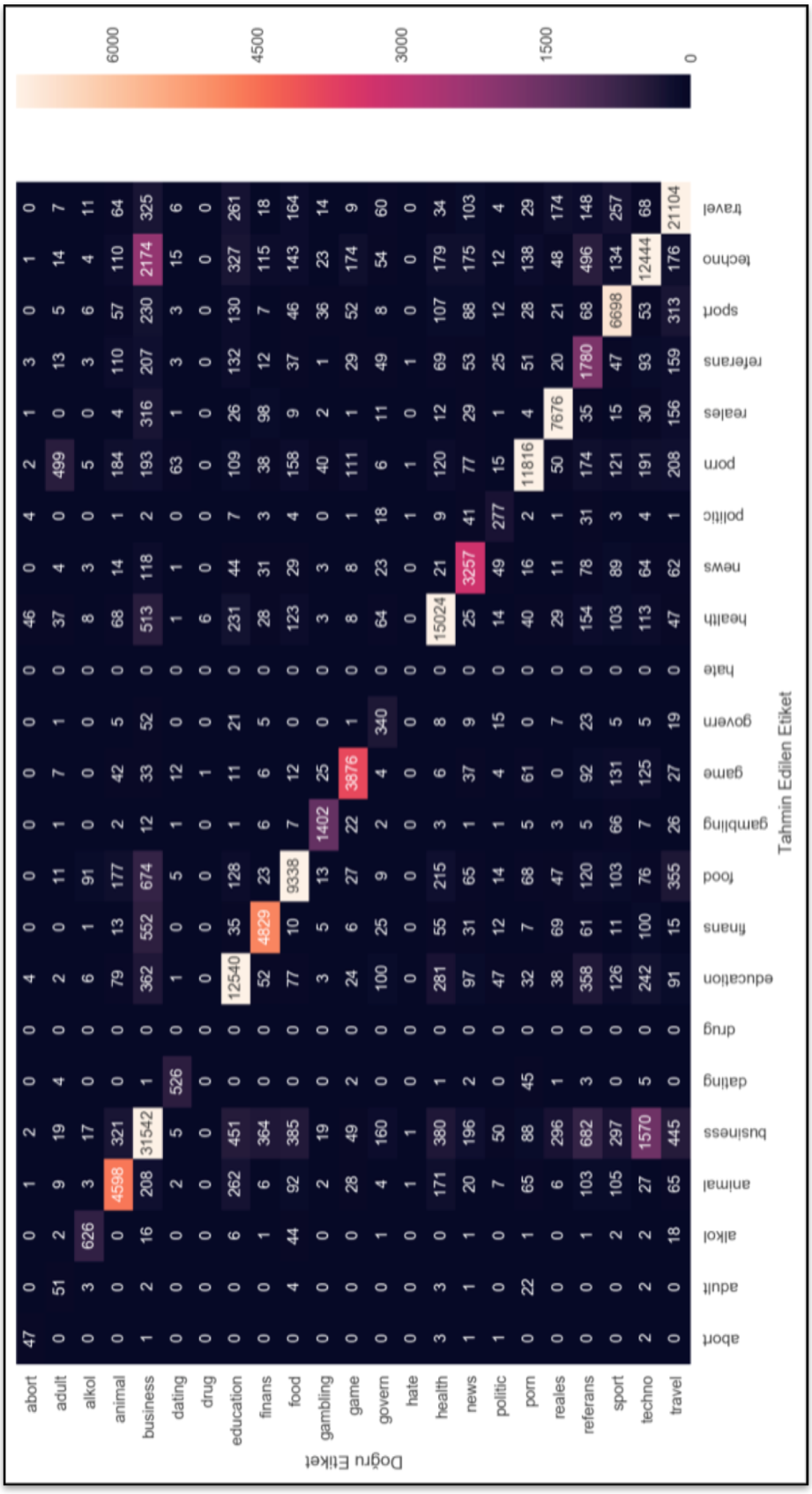
Şekil 3.28. Testlere İlişkin Validasyon ve Eğitim Başarı Oranları

Çizelge 3.26. yisa_5_ot Sınıf Bazlı Başarı Değerleri

Sınıf İsmi	Tutturma	Hassaslık	F-Skor	Test Örneği Sayısı
abort	0.85	0.42	0.57	111
adult	0.58	0.07	0.13	686
alkol	0.87	0.80	0.83	787
animal	0.79	0.79	0.79	5849
business	0.84	0.84	0.84	37533
dating	0.89	0.82	0.85	645
drug	0.00	0.00	0.00	7
education	0.86	0.85	0.86	14722
finans	0.83	0.86	0.84	5642
food	0.81	0.87	0.84	10682
gambling	0.89	0.88	0.89	1591
game	0.86	0.88	0.87	4428
govern	0.66	0.36	0.47	938
hate	0.00	0.00	0.00	5
health	0.90	0.90	0.90	16701
news	0.83	0.76	0.79	4309
politic	0.68	0.49	0.57	560
porn	0.83	0.94	0.89	12518
reales	0.91	0.90	0.91	8497
referans	0.61	0.40	0.49	4412
sport	0.84	0.81	0.82	8313
techno	0.73	0.82	0.77	15223
travel	0.92	0.91	0.91	23287
Ortalama / Toplam	**0.84**	**0.84**	**0.84**	**177446**

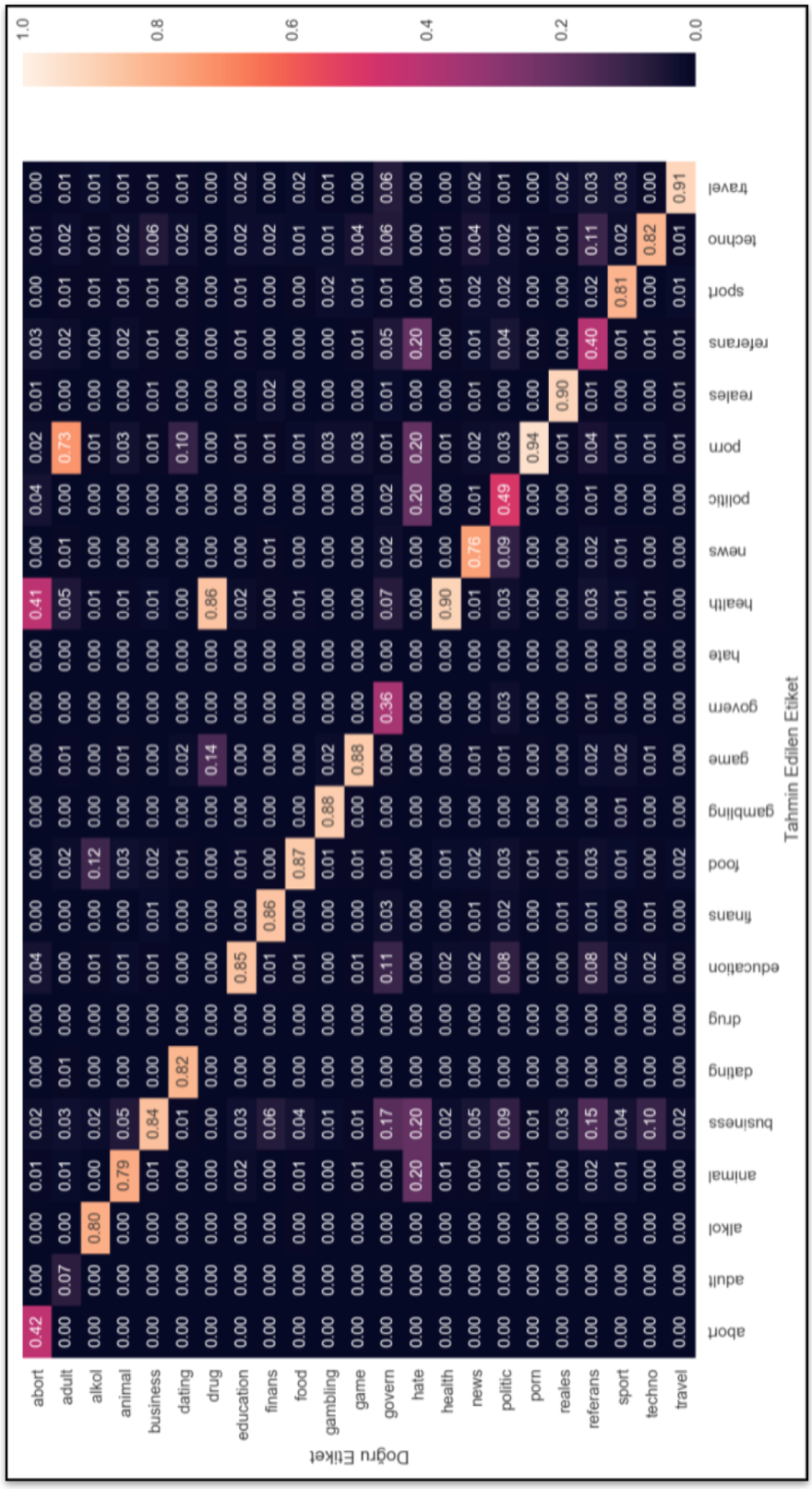
Çizelge 3.26'daki verilere göre bazı sınıflar başarılı bir şekilde tanınmışken bazı sınıflar iyi tanınamamıştır. Örneğin, "drug" ve "hate" sınıfları için veri örneği miktarı çok az olduğu için bu sınıflar için başarı sonuçları çok kötü çıkmıştır. Veri örneği sayısı yeterli olan bazı kategoriler oldukça iyi tanınabilmektedir. Örneğin; "travel", "reales", "health" gibi. Bazı kategorlerin ise tutturma oranı yüksek iken, hassaslık oranları düşük olabilmektedir. Örneğin "abort", "referens" gibi.

Tutturma, hassaslık ve f-skor ortalamaları 0.84 olsa da geliştirilen sistem bazı sınıfları ayırt etmekte başarılı olamamaktadır. Tutturma ve hassaslık oranlarındaki düşüklüğün analiz edilebilmesi için birbirleri ile karışan sınıfların tespit edilmesi gerekmektedir. Bu tespit işlemi için Karışıklık Matrisi kullanılır. *yisa_5_ot* testi için çıkarılmış olan karışıklık matrisi Şekil 3.29'de verilmiştir.



Şekil 3.29. yisa_5_ot için Karışıklık Matrisi

Şekil 3.29’da verilen karışıklık matrisi sıcaklık haritası olarak çizdirilmiştir. Bu matrisde yer alan bir hücredeki örnek sayısı yüksek ise o hücrenin rengi daha açık olarak boyanmıştır. En açık renk beyaza yakın olan renktir. En koyu renk ise mora yakın olan renktir. Renk sıkalası matrisin yanında görülmektedir. Test edilen sınıflar için farklı test sayıda test örnekleri olduğu için matrisde karışıklıkların belirgin görebilmesi için açık renk boyanması için maximum değer 7000 olarak girilmiştir. İdeal sistemde diyagoneldeki hücrelerin açık renk, diğer hücrelerin koyu renk olmaları beklenir. Ancak, Şekil 3.25’e göre, elde edilen sonuçlarda sınıflar arası karılıkşıklar olduğu görülmektedir. Örneğin, adult kategorisi porn kategorisi ile karışabilmektedir. Veri kümemizde dengesiz sınıf dağılımı olduğu için veri örneği sayıları ile karışıklık matrisi bize yanıltıcı bilgiler verebilir. Örneğin, adult kategorisinin çok büyük kısmı porn kategorisi ile karışmıştır. “adult” olmasına rağmen “adult” olarak tanınan örnek sayısı 51 iken “adult” olmasına rağmen “porn” tanınan örnek sayısı 499’dur. Ancak, iki hücrede birbirine yakın renkte boyanmıştır. Şekil 3.25 de verilen grafik bize sınıf karışıklıkları hakkında birtakım bilgiler verse de daha doğru analizler yapılabilmesi için karışıklık matrisinin normalize edilmiş hali kullanılır. Yisa_5_ot için normalize edilmiş karışıklık matrisi Şekil 3.30’da verilmiştir (Çizelgedeki değerlere sadece virgülden sonraki ilk iki basamak yazdırılmıştır).



Şekil 3.30. yisa_5_ot için Normalize Edilmiş Karışıklık Matrisi

Normalize edilmiş karışıklık matrisi, veri örnekleri yerine yüzdesel ifadelerden oluşur. Örneğin, “alkol” sınıfının %80’i “alkol” olarak sınıflandırılmıştır. “adult” kategorisi ile “porn” kategorisi arasındaki karışıklık miktarını analiz etmek istediğimizde ilgili hücrenin renginin çok açık olduğunu görülmektedir. Bu sayede “adult” kategorisi ile “porn” kategorisinin karıştığını daha net görebiliyoruz. Başka bir dikkat çekici örnek ise “abort” kategorisi ile “health” kategorisi arasındaki karışıklıktır. “abort” kategorisi kürtaj ile alakalı bir kategoridir, “health” kategorisi ise sağlık ile ilgili bir kategoridir. Bu kategoriler yapıları gereği birbirine benzer veri örnekleri içerebilir. Nitekim, Şekil 3.26’daki normalize edilmiş karışıklık matrisi incelendiğinde “abort” kategorisinin %41 oranında “health” kategorisi ile karıştırıldığı görülmektedir. Normalize edilmiş karışıklık matrisinin incelenmesi birbirine yakın olan sınıfların tespit edilebilmesine olanak tanır. Bu sayede birbiri ile çok karışan sınıfların birbirlerinden ayırt edilebilmesi için ek önlemler alınabilir ya da bazı kategoriler birleştirilerek tek kategori haline getirebilir.

Test edilen her iki mimari için hesaplamasal yük farklıdır. YiSA’da bir parametrenin öğrenilmesi için daha fazla matematiksel işlem gerekirken, İBYSA’da işlemler daha basit gerçekleştirilmektedir. Gerçekleştirilen tüm testler 10 epoch boyunca çalıştırılmıştır. Tüm testler için eğitim aşamasında bir epochun ortalama çalışma sürelerine ilişkin değerler Şekil 3.31’de verilmiştir.

Test İsmi ▲	Ortalama Epoch Süreleri
ibysa_1_ot	54.95
ibysa_3_ot	115.3
ibysa_5_ot	177.8
yisa_1_ot	459.8
yisa_3_ot	1189.5
yisa_5_ot	1913

Şekil 3.31. Eğitim Aşamasında Bir Epcoh için Çalışma Süreleri (Saniye Cinsinden)

Şekil 3.31’de görüldüğü üzere katman sayılarının artması çalışma sürelerini de artırmaktadır. YiSA için eğitim sürelerinin İBYSA ya kıyasla çok daha sürdüğü tespit edilmiştir. Bu durumun nedeni YiSA yaklaşımında bir paremetrenin öğrenilmesi için İBYSA’ya kıyasla daha kompleks hesaplamalar yapılmasıdır. İki mimari arasında başarı

kıyaslamasında ciddi farklar gözlenmezken İBYSA, YiSA'dan çok daha hızlı çalışmaktadır. Bu nedenle İBYSA'nın daha performanslı çalıştığı söylenebilir.

Öğrenme transferinin kullanılması öğrenilecek parametrelerin azalmasına neden olmaktadır. Gerçekleştirilen testler için toplam parametre sayıları ve öğrenme transferi sonrası öğrenilmesi gereken parametre sayıları Çizelge 3.24'te verilmiştir. Öğrenilen parametre sayılarının toplam parametre sayılarına kıyasla çok az olduğu görülecektir. Toplam parametre sayısı içerisinde kelime vektörleri için kullanılan parametreler çok yer tutmaktadır. Kelime vektörü parametreleri ile model içerisinde yer alan diğer parametrelerin öğrenilmesi de ayrı hesaplamasal karmaşıklığa sahiptir. Bu nedenle öğrenilen parametre sayılarındaki ciddi düşüş çalışma sürelerinde doğrusal bir düşüşe neden olmaz.

Çizelge 3.25'te başarı oranları verilen teslerin tamamı öğrenme transferi kullanılmadan da gerçekleştirilmiştir. Tüm testler için çalışma sürelerinin kıyaslanmasına ilişkin görsel Şekil 3.32'de verilmiştir.

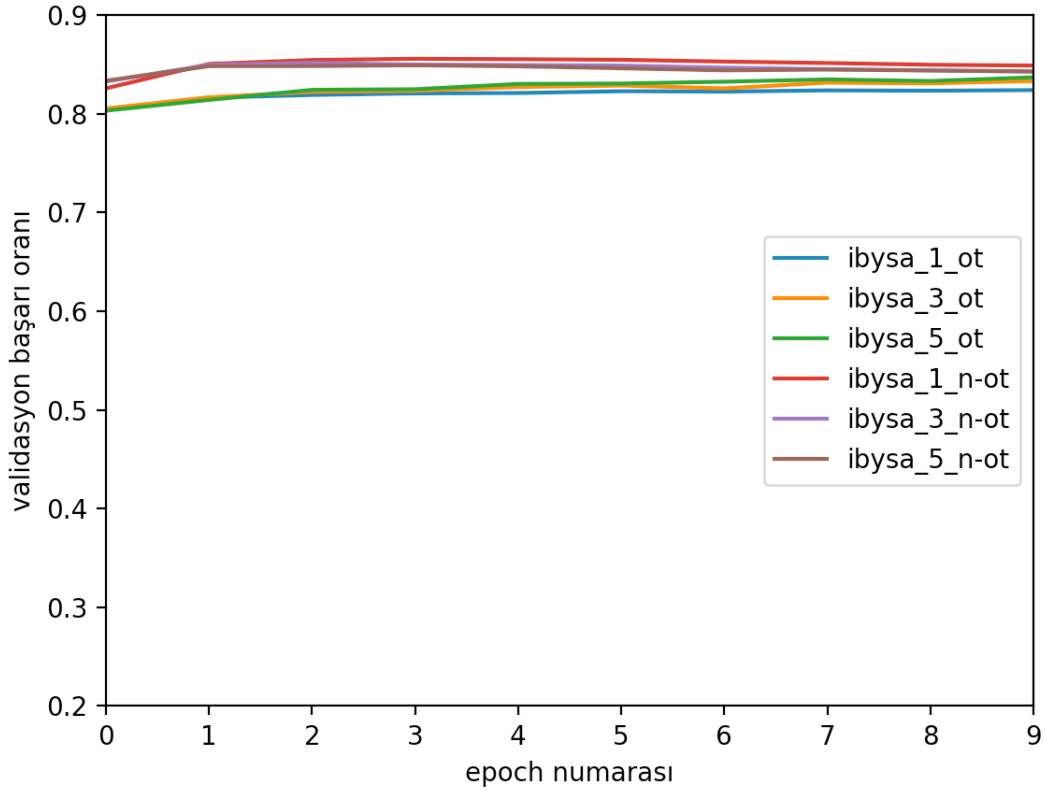
Test İsmi ▲	Öğrenme Transferi	Ortalama Epoch Süreleri
ibysa_1_ot	Var	54.95
ibysa_1_n-ot	Yok	81.11
ibysa_3_ot	Var	115.3
ibysa_3_n-ot	Yok	180.1
ibysa_5_ot	Var	177.8
ibysa_5_n-ot	Yok	283.1
yisa_1_ot	Var	459.8
yisa_1_n-ot	Yok	532.6
yisa_3_ot	Var	1189.5
yisa_3_n-ot	Yok	1557.1
yisa_5_ot	Var	1913
yisa_5_n-ot	Yok	2035

Şekil 3.32. Öğrenme Transferi için Çalışma Süreleri Karşılaştırması

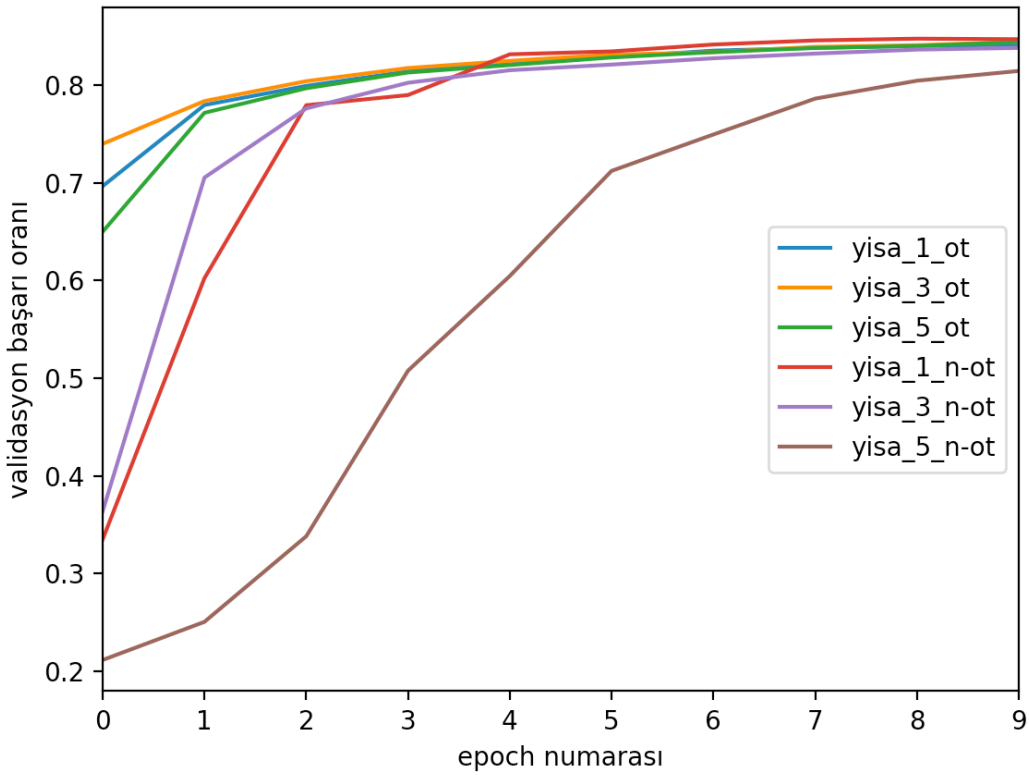
Şekil 3.32'de aynı testin öğrenme transferi kullanılarak gerçekleştirilen hali ve kullanılmadan gerçekleştirilen hali arka arkaya verilmiştir. Sonuçlardan görüleceği

üzere tüm testlerde öğrenme transferi kullanılan testler, kullanılmayan testlerden daha hızlı çalışmıştır. Ancak, çalışma süreleri arasında fark yüksek oranlarda değildir.

Öğrenme transferinin başarı durumlarına etkisi Şekil 3.33'de ve Şekil 3.34'de verilmiştir. Şekil 3.33, İBYSA için elde edilen sonuçları içermektedir. Şekil 3.34 ise YiSA için elde edilen sonuçları içermektedir. Grafikteki değerler validasyon başarı oranlarını göstermektedir.



Şekil 3.33. İBYSA için Başarı Değerleri



Şekil 3.34. YiSA için Başarı Değerleri

Şekil 3.33'deki değerlere göre İBYSA için öğrenme transferinin başarı oranlarında ciddi iyileştirmelere neden olmadığı görülmektedir. Başarı oranlarında iyileşme olmamasına rağmen öğrenme transferi kullanılarak gerçekleştirilen testlerin daha hızlı çalışması önemli bir durumdur.

Şekil 3.34'deki değerlere göre öğrenme transferi kullanılmayan durumlarda öğrenme işlemi daha yavaş gerçekleşmektedir. Ancak, yeterli epoch sayısında eğitim gerçekleştirilirse tüm testler için başarı değerlerinin birbirine yakın değerlere yakınsayacağı görülmektedir. Şekil 3.34'de en başarısız test yisa_5_n-ot olarak görülmektedir. Bu testte öğrenme transferi kullanılmamıştır. Onuncu epoch tamamlandığında bu test için başarı değeri için henüz yakınsamadığı anlaşılmaktadır. Başarı değerleri artmaya meyilli olarak test sonlanmıştır. Epoch sayısı artırılırsa bu testin de diğer testlere yakın bir başarı değerine yakınsacağı düşünülmektedir.

3.2.1 Derin Öğrenme Yaklaşımları ile Geleneksel Yöntemlerin Karşılaştırılması

Bu çalışma kapsamında web sayfası sınıflandırmasına yönelik bir sistem geliştirilmiştir. Geliştirilen sistem ilk aşamada derin öğrenme yaklaşımları ile test edilmiştir. Ardından

web sayfası sınıfladırması işlemi aynı veri seti üzerinde geleneksel algoritmalar üzerinde de çalıştırılmıştır. Ardından elde edilen sonuçlar karşılaştırılmıştır. Test edilen geleneksel algoritmalar şu şekildedir;

- Ağaç Tabanlı Rastsal Orman (Random Forest - RO) Algoritması
- İstatistik Tabanlı Naive Bayes (NB) Algoritması
- Kernel Tabanlı Squential Minimal Optimization (SMO)

Testler esnasında derin öğrenme testlerinde olduğu gibi 0.8 oranında eğitim ve 0.2 oranında test bölümlemesi uygulanmıştır.

Geleneksel algoritmaların temsil edilmesinde Word2Vec yaklaşımı kullanılmıştır. Bir web sayfası için vektör oluşturulurken, o web sayfası için toplanmış kelimelerin kelime vektörlerinin ortalaması alınmıştır. Kelime vektörleri Glove6B100d [67]'den alınmıştır. Ortalama alma işlemi tüm vektörker için belirli indekste yer alan değerlerin ortalamasının alınması şekliyle hesaplanmıştır. Bu sayede her veri örneği 100 boyutlu bir vektör ile temsil edilmiştir.

Derin öğrenme testlerinde kullanılan test verisi ile geleneksel algoritmalarda kullanılan test verileri birebir aynı olarak ayarlanmıştır. Test edilen 3 geleneksel algoritma için test verisi üzerinde elde edilen başarı oranı (accuracy), hassaslık, tutturma ve f-skor değerleri değerleri Çizelge 3.27'de verilmiştir. Derin öğrenme yaklaşımları kullanılarak gerçekleştirilen testlerden en yüksek başarı oranı oranına sahip teste ilişkin değerlendirme metrikleri de karşılaştırma yapılabilmesi için aynı çizelgeye girilmiştir.

Çizelge 3.27. Geleneksel Algoritmalar için Değerlendirme Metrikleri

Algoritma	**Başarı Oranı**	**Tutturma**	**Hassaslık**	**F-Skor**
NB	0.720	0.780	0.720	0.741
RO	0.819	0.824	0.819	0.814
SMO	**0.821**	0.819	**0.821**	**0.818**
yisa_5_ot	**0.844**	**0.840**	**0.840**	**0.840**

Başarı Oranı, hassaslık ve F-Skor değerlerinde SMO algoritmasının diğer iki algoritmadan daha başarılı olduğu görülmüştür. SMO algoritması kullanılarak gerçekleştirilen teste ilişkin sınıf tabanlı hassaslık, tutturma ve F-skor değerleri Çizelge 3.28'de verilmiştir. YiSA yaklaşımı ile gerçekleştirilen testin test edilen tüm geleneksel algoritmaları tüm değerlendirme metriklerinde geçtiği görülmüştür.

Çizelge 3.28. SMO Algoritması Metrikleri

Sınıf İsmi	**Tutturma**	**Hassaslık**	**F-Skor**
abort	0.817	0.570	0.671
adult	0.554	0.051	0.093
alkol	0.856	0.754	0.802
animal	0.785	0.713	0.747
business	0.767	0.858	0.810
dating	0.877	0.828	0.852
drug	0.000	0.000	0.000
education	0.872	0.818	0.844
finans	0.837	0.777	0.806
food	0.863	0.826	0.844
gambling	0.904	0.839	0.870
game	0.908	0.858	0.883
govern	0.593	0.419	0.491
hate	0.000	0.000	0.000
health	0.894	0.882	0.888
news	0.806	0.727	0.765
politic	0.720	0.575	0.639
porn	0.810	0.927	0.865
reales	0.898	0.859	0.878
referans	0.537	0.338	0.415
sport	0.818	0.769	0.793
techno	0.732	0.761	0.746
travel	0.888	0.893	0.890
Ortalama / Toplam	**0.819**	**0.821**	**0.818**

BÖLÜM 4

SONUÇ VE ÖNERİLER

Bu çalışmada web sayfalarının sınıflandırılmasına yönelik derin öğrenme tabanlı bir sistem geliştirilmiştir. Veri kümesi Roksit [74] firmasından elde edilmiştir. Kullanılan veri kümesinde 23 farklı kategori için toplamda 887,195 web sayfasına ilişkin bilgi bulunmaktadır. Web sayfalarının sınıflandırılmasında sayfalarda yer alan meta etiketler kullanılmıştır. Kullanılan meta etiketler başlık, açıklama ve anahtar kelimelerdir. Web sayfalarından toplanan bu bilgiler word2vec yöntemi ile vektörize edilmiştir. Vektörize edilen veri algoritmalara girdi olarak verilmiştir.

Sistem tek sınıflı sınıflandırıcı olacak şekilde tasarlanmıştır. İki farklı mimari test edilmiştir. Test edilen mimariler İBYSA ve YiSA'dır. Gerçekleştirilen testler hem CPU üzerinde hem GPU üzerinde gerçekleştirilmiştir. Her iki donanım ile çalıştırılan testler için performans karşılaştırması yapılmıştır. Gerçekleştirilen testlerde birden fazla CPU konfigürasyonu ve bir adet GPU denenmiştir. Performans karşılaştırmalarında batch boyutunun çalışma sürelerine ve kaynak tüketimlerine etkisi incelenmiştir. Benzer şekilde öğrenme transferi kullanmanın ve kelime vektör boyutunun çalışma sürelerine etkisi farklı donanımlar üzerinde test edilmiştir. Farklı donanımlar üzerinde gerçekleştirilen testlerde YiSA mimarisi kullanılmıştır. Elde edilen sonuçlara göre GPU üzerinde çalıştırılan testler CPU'ya kıyasla ortalama 5 kat daha hızlı çalışmaktadır.

Farklı donanımlarda çalıştırılan test sonuçlarının performansa etkisinin incelenmesinin ardından test edilen durumların başarı değerleri analiz edilmiştir. Sonuçlar

değerlendirilirken performans analizi değerlendirmesinin daha sağlıklı yapılabilmesi için başarı oranlarının yanı sıra çalışma süreleri de kıyaslanmıştır.

Test edilen mimariler için farklı katman sayılarının başarı oranlarına ve çalışma süreleri etkisi incelenmiştir. Elde edilen sonuçlara göre web sayfalarının sınıflandırılması için katman sayının artırılması başarı oranlarında ciddi iyileşmelere neden olmamakla beraber çalışma sürelerinin artmasına neden olmaktadır.

Geliştirilen sistemin ezberleme yapıp yapmadığı da incelenmiştir. Elde edilen sonuçlara yüksek oranda ezberleme gerçekleşmediği görülmüştür. YiSA mimarisinin ezberlemeye İBYSA'dan daha dayanıklı olduğu gözlenmiştir.

Son olarak öğrenme transferi kullanımının performansa etkisi analiz edilmiştir. Öğrenme transferi kullanılması öğrenilecek parametrelerin azalmasına neden olmaktadır. Bu nedenle öğrenme transferi kullanılarak gerçekleştirilen testlerin diğer testlere kıyasla daha hızlı çalıştığı gözlenmiştir. Öğrenme transferi kullanılarak gerçekleştirilen testler ile diğer testler arasında başarı oranlarında da iyileşmeye neden olduğu gözlenmemiştir. Ancak, öğrenme transferinin, eğitim sürelerini kısaltması nedeniyle performansa olumlu etki oluşturduğu gözlenmiştir.

Gerçekleştirilen tüm testlerde elde edilen validasyon başarı oranları %85 civarlarındadır.

Kullanılan veri kümesi dengesiz bir sınıf dağılımına sahip olduğu için sınıf bazlı tutturma, hassaslık ve f-skor değerleri incelenmiştir. İnceleme sonucunda bazı sınıfların oldukça düşük başarı oranları ile tanındığı gözlenmiştir. Düşük başarı oranları ile tanınan sınıfların analiz edilmesi için karışıklık matirisi kullanılmıştır. Karışıklık matrisi ile birbirleri ile yüksek oranda karışan sınıflar tespit edilmiştir.

KAYNAKLAR

[1] Altıngövde, I_. S., Özel, S. A., Ulusoy, Ö., Özsoyoglu, G., and Özsoyoglu, Z. M. (2001). "Topic-centric querying of Web information resources". Lecture Notes in Computer Science, 2113:699–711.

[2] De Bra, P. M. E., and Post, R. D. J. (1994). "Information retrieval in the World Wide Web: Making client-based searching feasible". Computer Networks and ISDN Systems, 27(2):183–192.

[3] Menczer, F., Pant, G., and Srinivasan, P. (2004). "Topical Web crawlers: Evaluating adaptive algorithms". ACM Transactions on Internet Technology, 4(4):378–419.

[4] Qi, X., and Davison, B. D. (2009). "Web page classification: Features and algorithms". ACM Computing Surveys, 41(2)

[5] Chen, R. C., and Hsieh, C. H. (2006). "Web page classification based on a support vector machine using a weighted vote schema". Expert Systems with Applications, 31:427–435.

[6] Selamat, A., and Omatu, S. (2004). "Web page feature selection and classification using neural networks". Information Sciences, 158:69–88.

[7] Wakaki, T., Itakura, H., Tamura, M., Motoda, H., and Washio, T. (2006). "A study on rough set-aided feature selection for automatic web-page classification". Web Intelligence and Agent Systems: An International Journal, 4:431–441.

[8] Chen, R. C., and Hsieh, C. H. (2006). "Web page classification based on a support vector machine using a weighted vote schema". Expert Systems with Applications, 31(2):427-435.

[9] Ozel, S. A. (2011). "A genetic algorithm based optimal feature selection for web page classification". In Innovations in Intelligent Systems and Applications (INISTA), 15-18 June 2011, Istanbul, 282-286.

[10] Meshkizadeh, S., and Dezfuli, M. A. (2010). "Webpage Classification based on URL Features and Features of Sibling Pages". International Journal of Computer Science and Information Security, 8(2):168-173.

[11] Ozel, S. A. (2011). “A web page classification system based on a genetic algorithm using tagged-terms as features”. Expert Systems with Applications, 38(4):3407-3415.

[12] Liu, H., and Huang, S. (2003). “A genetic semi-supervised fuzzy clustering approach to text classification”. Lecture Notes in Computer Science, 2762:173–180.

[13] Pietramala, A., Policicchio, V. L., Rullo, P., and Sidhu, I. (2008). “A genetic algorithm for text classification rule induction”. Lecture Notes in Artificial Intelligence, 5212:188–203.

[14] Qi, D., and Sun, B. (2004). “A genetic k-means approaches for automated Web page classification”. In IEEE international conference on information reuse and integration, 20-24 June 2004. Las Vegas, 241–246.

[15] Ribeiro, A., Fresno, V., Garcia-Alegre, M. C., and Guinea, D. (2003). “Web page classification: A soft computing approach”. Lecture Notes in Artificial Intelligence, 2663:103–112.

[16] Holden, N., and Freitas, A. A. (2004). Web page classification with an ant colony algorithm. In International Conference on Parallel Problem Solving from Nature, 7 September 2004, Berlin, 1092-1102.

[17] DURAL, B., (2012), Türkçe Arama Motoru Sonucu Kümeleme Çalışmaları, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[18] Cover, T. M., and Thomas, J. A. (2012). “Elements of information theory”. John Wiley & Sons.

[19] Cai, L., and Hofmann, T. (2003). “Text categorization byboosting automatically extracted concepts”. The 26th Annual International ACM SIGIR Conference, July 28 – 1 Agu 2003, Toronto, 182–189.

[20] Hingmire, S., Chougule, S., Palshikar, G. K., and Chakraborti, S. (2013). “Document classification by topic labeling”. In SIGIR, July 28 – 1 Agu 2013, Dublin, 877–880.

[21] Dou, S., Zheng, C., Qiang Y., Hua-Jun, Z., Benyu, Z., Yuchang, L., and Wei Ying M. (2004), “Web-page classification through summarization”, In the Proceedings of the 27th annual international ACM SIGIR 04, conference on. Research and Development in Information Retrieval, 12 October 2004, New York, 242- 249.

[22] Attardi, G., Gullì, A., and Sebastiani, F. (1999). “Automatic Web page categorization by link and context analysis”. In Proceedings of THAI, 99(99):105-119.

[23] Kan, M.Y. (2004). “Web page classification without the web page”. In WWW Alt. ’04: Proceedings of the 13th International World Wide Web Conference Alternate Track Papers & Posters, 17-22 May 2004, New York, 262–263.

[24] Qi, X., and Davison, B. D. (2009). “Web page classification: Features and algorithms”. ACM computing surveys (CSUR), 41(2):12.

[25] Kovacevic, M., Diligenti, M., Gori, M., and Milutinovic, V. (2004). “Visual adjacency multigraphs-a novel approach for a web page classification”. In Proceedings of SAWM04 workshop, ECML2004. 21 September 2004, Siena.

[26] Asirvatham, A. P. and K. K. Ravi (2001). “Web page classification based on document structure”. Awarded second prize in National Level Student Paper Contest conducted by IEEE India Council. 14 January 2001, Hyderabad.

[27] Sriurai, W., Meesad, P., & Haruechaiyasak, C. (2010). “Hierarchical web page classification based on a topic model and neighboring pages integration”. 1003.1510.

[28] Menczer, F. (2005). “Mapping the semantics of web text and links”. IEEE Internet Computing, 9(3):27-36.

[29] Davison, B. D. (2000, July). “Topical locality in the Web”. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 24-28 July 2000, New York, 272–279.

[30] Chakrabarti, S., M. M. Joshi, K. Punera, and D. M. Pennock (2002). “The structure of broad topics on the web”. In WWW ’02: Proceedings of the 11th International Conference on World Wide Web, 07-11 May 2002, New York, 251–262.

[31] Chakrabarti, S., B. E. Dom, and P. Indyk (1998). “Enhanced hypertext categorization using hyperlinks”. In SIGMOD ’98: Proceedings of the ACM SIGMOD International Conference on Management of Data, 01-04 June 1998, New York, 307–318.

[32] Ghani, R., S. Slattery, and Y. Yang (2001). “Hypertext categorization using hyperlink patterns and meta data”. In ICML ’01: Proceedings of the 18th International Conference on Machine Learning, 28 June-01 July 2001, San Francisco, 178–185.

[33] Yang, Y., S. Slattery, and R. Ghani (2002). “A study of approaches to hypertext categorization”. Journal of Intelligent Information Systems 18(2-3):219–241.

[34] Slattery, S., and Mitchell, T. (2000, June). “Discovering test set regularities in relational domains”. In ICML, 29 June–02 July 2000, San Francisco, 895-902.

[35] Calado, P., M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M. A. Goncalves (2003). “Combining link-based and content-based methods for web document classification”. In CIKM ’03: Proceedings of the 12th International Conference on Information and Knowledge Management, 03-08 November 2003, New York, 394–401.

[36] Glover, E. J., K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake (2002). “Using web structure for classifying and describing web pages”. In Proceedings of the 11th International Conference on World Wide Web, 07-11 May 2002, New York, 562–569.

[37] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). “A Neural Probabilistic Language Model”. JMLR 3:1137–1155.

[38] Mikolov, T., Yih, W.T., and Zweig, G. (2013). “Linguistic regularities in continuous space word representations”. The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 9-14 June 2013, Atlanta, 746–751.

[39] Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., and Ng, A. Y. (2014). “Grounded compositional semantics for finding and describing images with sentences”. Transactions of the Association of Computational Linguistics, 2(1):207-218.

[40] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. (2011). “Natural languageprocessing (almost) from scratch”. JMLR 12:2493–2537.

[41] Lai, S., Xu, L., Liu, K., and Zhao, J. (2015, January). “Recurrent Convolutional Neural Networks for Text Classification”. In AAAI, 333:2267-2273.

[42] Medsker, L. R., and Jain, L. C. (2001). Recurrent neural networks*. Design and Applications*, 392:*5*.

[43] Tartarus, The Porter Algorithm, https://tartarus.org/martin/PorterStemmer/, 1 Kasım 2017.

[44] Porter, M. F. (1980). “An algorithm for suffix stripping”. Program, 14(3):130-137.

[45] Shanks, V. and H. E. Williams (2001, November). “Fast categorisation of large document collections”. In Proceedings of Eighth International Symposium on String Processing and Information Retrieval (SPIRE) 13-15 November 2001, Laguna de San Rafael, 194–204.

[46] Zelikovitz, S. and H. Hirsh (2001). “Using LSI for text classification in the presence of background text”. In Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM), 6-8 June 2001, New York, 113–118.

[47] Fisher, M. J. and R. M. Everson (2003). “When are links useful? Experiments in text classification”. In Advances in Information Retrieval. 25th European Conference on IR Research, 14-16 April 2003, Pisa, 41–56.

[48] Mangai, J. A., and Kumar, V. S. (2011). “A novel approach for web page classification using optimum”. IJCSNS, 11(5):252.

[49] Roksit, Roksit Easiest Way to be Secure, https://www.roksit.com/domain-categorization/, 1 Kasım 2017

[50] Hurriyetemlak, Hürriyet Emlak – Türkiye'nin En Büyük Emlak Sitesi, https://www.hurriyetemlak.com, 1 Mart 2018.

[51] Hurriyetemlak, Hürriyet Emlak – Türkiye'nin En Büyük Emlak Sitesi view-source:https://www.hurriyetemlak.com/, 1 Mart 2018.

[52] Github, Port of Google's language-detection library to Python, https://github.com/Mimino666/langdetect, 21 Mayıs 2018.

[53] Coursera, Sequence Models, deeplearning.ai, https://www.coursera.org/learn/nlp-sequence-models, 12 Kasım 2018.

[54] Github, The Unreasonable Effectiveness of Recurrent Neural Network, http://karpathy.github.io/2015/05/21/rnn-effectiveness/, 21 Kasım 2018.

[55] Github, Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/, 12 Mayıs 2018.

[56] Github, Pre-trained word vectors trained by Facebookresearch, https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md, 7 Nisan 2018.

[57] Hochreiter, S., and Schmidhuber, J. (1997). "Long short-term memory". Neural computation, 9(8):1735-1780.

[58] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). "Efficient estimation of word representations in vector space". *arXiv:*1301.3781.

[59] Pennington, J., Socher, R., and Manning, C. (2014). "Glove: Global vectors for word representation". In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 25-29 October 2014, Doha, 1532-1543.

[60] Maaten, L. V. D., and Hinton, G. (2008). "Visualizing data using t-SEN". Journal of machine learning research, 9:2579-2605.

[61] Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., and Mikolov, T. (2013). "Devise: A deep visual-semantic embedding model". In Advances in neural information processing systems, 2121-2129.

[62] Coursera, Sequence Models Course on Coursera represented by Andrew NG, https://www.coursera.org/learn/nlp-sequence-models/home/welcome, 1 Nisan 2018.

[63] Tensorflow, Vector Representations of Words, https://www.tensorflow.org/tutorials/word2vec, 1 Nisan 2018.

[64] Analyticsvidhya, An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/, 1 Nisan 2018.

[65] Mccormickml, Word2Vec Tutorial - The Skip-Gram Model, http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/, 1 Nisan 2018.

[66] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). "Distributed representations of words and phrases and their compositionality". In Advances in neural information processing systems, 3111-3119.

[67] Stanford, GloVe: Global Vectors for Word Representation, https://nlp.stanford.edu/projects/glove/, 1 Nisan 2018.

[68] Mccormickml, Google's trained Word2Vec model in Python, http://mccormickml.com/2016/04/12/googles-pretrained-word2vec-model-in-python/, 1 Nisan 2018.

[69] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). "Enriching word vectors with subword information". arXiv preprint arXiv:1607.04606.

[70] Github, fastText Japanese Tutorial, https://github.com/icoxfog417/fastTextJapaneseTutorial, 3 Nisan 2018.

[71] Lmu, Meta-Embeddings: Higher-quality word embeddings via ensembles of Embedding Sets, http://cistern.cis.lmu.de/meta-emb/, 3 Nisan 2018.

[72] Wordpress, Dependency-Based Word Embeddings, https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/, 4 Nisan 2018.

[73] Github, Word Embedding based Semantic Relation Detection for Turkish Language, https://github.com/savasy/TurkishWordEmbeddings, 5 Nisan 2018.

[74] Roksit, DNS Firewall & Malware Detection | The Easiest Way to be Secure, https://www.roksit.com, 11 Nisan 2018.

[75] Floyhub, FloydHub is a zero setup Deep Learning platform for productive data science teams.https://www.floydhub.com/, 11 Nisan 2018.

[76] Intel, Intel® Xeon® Gold 6126 Processor, https://ark.intel.com/products/120483/Intel-Xeon-Gold-6126-Processor-19_25M-Cache-2_60-GHz, 12 Nisan 2018.

[77] Nvidia, NVIDIA TESLA K80, https://www.nvidia.com/en-us/data-center/tesla-k80/, 13 Nisan 2018.

[78] Anandtech, NVIDIA Launches Tesla K80, GK210 GPU, https://www.anandtech.com/show/8729/nvidia-launches-tesla-k80-gk210-gpu, 11 Nisan 2018.

[79] Keras, Keras: The Python Deep Learning Library, https://keras.io/, 21 Kasım 2018.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Ebubekir BÜBER

Doğum Tarihi ve Yeri :21.08.1992, Kayseri/Melikgazi

Yabancı Dili : İngilizce

E-posta : ebubekirbbr@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Y. Lisans	Bilgisayar Müh.	Yıldız Teknik Üniversitesi	2018
Lisans	Bilgisayar Müh.	Yıldız Teknik Üniversitesi	2015
Lise	Sayısal	Kocaeli Körfez Fen Lisesi	2010

İŞ TECRÜBESİ

Yıl	Firma/Kurum	Görevi
2016	Normshield Inc. (1 Yıl)	Data Scientist
2018	Roksit Ltd. Şti. (Halen çalışmaktayım)	Data Scientist

YAYINLARI

Bildiri

1. Buber, E., and Sahingoz, O. K. (2017). “Makine Öğrenmesi sistemi ile görüntü İşleme ve en uygun parametrelerin ayarlanmasi”. In Artificial Intelligence and Data Processing Symposium (IDAP), 16-18 September 2017, Malatya, 1-5.

2. Buber, E., Demir, Ö., and Sahingoz, O. K. (2017). “Feature selections for the machine learning based detection of phishing websites”. In Artificial Intelligence and Data Processing Symposium (IDAP), 16-18 September 2017, Malatya, 1-5.

3. Buber, E., and Guvensan, A. M. (2014). “Discriminative time-domain features for activity recognition on a mobile phone”. In Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 12-14 April 2014, Singapur, 1-6.

4. Buber, E., Dırı, B., and Sahingoz, O. K. (2017). “Detecting phishing attacks from URL by using NLP techniques”. In International Conference on Computer Science and Engineering (UBMK), 7-9 October 2017, Antalya, 337-342.

5. Buber, E., and Dırı, B. (2018). “Performance Analysis and CPU vs GPU Comparison for Deep Learning”. In International Conference on Control Engineering & Information Technology, 4-11 October 2018, Istanbul, 1059-1064.