T.C.
EGE ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

SORU YANITLAMA SİSTEMLERİ İÇİN HİBRİD MAKİNE ÖĞRENMESİ TEKNİKLERİNE DAYALI BİR YÖNTEM TASARIMI VE GERÇEKLEŞTİRİMİ

Doktora Tezi

Sinem ÇINAROĞLU

Bilgisayar Mühendisliği Anabilim Dalı

İzmir
2022

T.C.
EGE ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

SORU YANITLAMA SİSTEMLERİ İÇİN HİBRİD MAKİNE ÖĞRENMESİ TEKNİKLERİNE DAYALI BİR YÖNTEM TASARIMI VE GERÇEKLEŞTİRİMİ

Sinem ÇINAROĞLU

Danışman: Doç. Dr. Hasan BULUT

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Doktora Programı

İzmir
2022

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum "**SORU YANITLAMA SİSTEMLERİ İÇİN HİBRİD MAKİNE ÖĞRENMESİ TEKNİKLERİNE DAYALI BİR YÖNTEM TASARIMI VE GERÇEKLEŞTİRİMİ**" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

25/ 02/ 2022

Sinem ÇINAROĞLU

ÖZET

SORU YANITLAMA SİSTEMLERİ İÇİN HİBRİD MAKİNE ÖĞRENMESİ TEKNİKLERİNE DAYALI BİR YÖNTEM TASARIMI VE GERÇEKLEŞTİRİMİ

ÇINAROĞLU, Sinem

Doktora Tezi, Bilgisayar Mühendisliği Anabilim Dalı
Tez Danışmanı: Doç. Dr. Hasan BULUT
Şubat 2022, 140 sayfa

Doğal dilde insan-bilgisayar etkileşiminin başlıca araştırma alanlarından biri olan Soru Yanıtlama Sistemleri, doküman uzayını kullanarak doğal dilde insanlar tarafından sorulan sorulara otomatik olarak cevaplar vermek için geliştirilen bir mimaridir. Ancak, Doğal Dil İşleme'de kullanılan öznitelik uzayının seyrekliği, yüksek boyutluluğu ve gereksiz özniteliklerin varlığı, kullanıcı sorularına doğru cevabı sunma konusunda önemli bir problem teşkil etmektedir. Bu zorluklarla mücadele etmek için sıklıkla başvurulan çözümlerin başında Makine Öğrenmesi yöntemleri gelmektedir.

Bu tez çalışması kapsamında da, verileri en doğru şekilde temsil edecek, makine öğrenmesi tekniklerine dayalı etkin bir öznitelik seçim yöntemi önerilmiştir. Ayrıca bilinen çalışmalardan farklı olarak, soru yanıtlama problemi, sınıflandırma problemi olarak ele alınmış, bu problemin çözümü için derin öğrenme tabanlı soru yanıtlama sistemlerini kullanan melez bir sistem tasarımı geliştirilmiştir. Daha sonra, önerilen bu yöntemler soru yanıtlama çalışmalarında sıklıkla kullanılan TREC ve SQuAD veri seti üzerinde test edilmiş, temel makine öğrenmesi yöntemlerinin bireysel performansları ile karşılaştırılmıştır. Geliştirilen yöntemlerin performansı doğruluk ve F1-Skor ölçütleri kullanılarak değerlendirilmiştir. Yapılan deneysel çalışmalar sonucunda, geliştirilen yöntemler ile temel makine öğrenmesi yöntemlerinin soru yanıtlama ve sınıflandırma üzerindeki başarılarının arttırıldığı gözlemlenmiştir.

Anahtar Sözcükler: Soru yanıtlama sistemleri, soru sınıflandırma, topluluk öğrenmesi, çoğunluk oylama, öznitelik seçimi, derin öğrenme tabanlı öznitelik temsili.

ABSTRACT

THE DESIGN AND IMPLEMENTATION OF A METHOD FOR QUESTION ANSWERING SYSTEMS BASED ON HYBRID MACHINE LEARNING TECHNIQUES

ÇINAROĞLU, Sinem

PhD in Computer Engineering
Supervisor: Assoc. Prof. Dr. Hasan BULUT
February 2022, 140 pages

Question Answering Systems is one of the main research areas of human-computer interaction in natural language. These systems are architectures that use document space and generate automatic answers of questions asked by people in natural language. Nevertheless, the sparseness, high dimensionality and redundancy of a feature space used in Natural Language Processing cause a significant problem in serving the correct answer to user questions. At this point, machine learning methods emerge as one of the most frequently used solutions to cope with this difficulty.

Within the scope of this thesis, an effective feature selection method based on machine learning techniques has been proposed to represent the data in the most accurate way. In addition, unlike the known studies, the question answering problem is handled as a classification problem, and a hybrid system design using deep learning-based question answering systems is developed to solve this problem. Then, these proposed methods are tested on the TREC and SQuAD datasets, which are frequently used in question answering studies, and compared with the individual performances of classical machine learning methods. The performance of the developed methods is evaluated using the Accuracy and F1-Score metrics. As a result of experimental studies, it has been observed that the proposed methods increase the successes of classical machine learning methods on both question answering and classification tasks.

Keywords: Question answering systems, question classification, ensemble learning, majority voting, feature selection, deep learning-based feature representation.

ÖNSÖZ

Bu tezde, soru yanıtlama (question answering) için en iyi hibrit sınıflandırıcı topluluğu kombinasyonu oluşturmak üzerinde odaklanılmıştır. Mimarinin oluşturulmasında Destek Vektör Makineleri, Lojistik Regresyon, Naif Bayes, K-En Yakın Komşu, Karar Ağaçları gibi temel makine öğrenmesi yöntemleri ve Rastgele Orman, Bagging, Adaboost ve Oylama topluluk öğrenme yöntemleri sınıflandırıcı olarak kullanılmıştır. Soru yanıtlama problemlerinde kullanılan veri seti üzerinde deneysel çalışmalar gerçekleştirilmiştir. Mimarinin değerlendirilebilmesi için, performans değerlendirme ölçütleri kullanılmıştır. Son olarak, temel makine öğrenmesi yöntemlerinin sınıflandırma başarısı, topluluk öğrenmesi yöntemlerinin sınıflandırma başarısı ile karşılaştırılmıştır. Tez çalışmasında, soru yanıtlama problemi için, topluluk öğrenmesi yöntemlerinde farklı birleştirme kombinasyonları oluşturularak en iyi kombinasyonun bulunmasına yönelik bir bakış açısı kazandırılmıştır.

İZMİR

25 / 02 / 2022 Sinem ÇINAROĞLU

İÇİNDEKİLER

Sayfa

ŞEKİLLER DİZİNİ

Şekil Sayfa

ŞEKİLLER DİZİNİ (devam)

Şekil Sayfa

ŞEKİLLER DİZİNİ (devam)

Şekil Sayfa

TABLOLAR DİZİNİ

Tablo Sayfa

TABLOLAR DİZİNİ(devam)

Tablo Sayfa

TABLOLAR DİZİNİ(devam)

Tablo Sayfa

TABLOLAR DİZİNİ(devam)

Tablo Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler	Açıklama
C_m	Karmaşıklık matrisi
$P_{m,i}$	m yönteminin i sınıfı için kesinlik değeri
SV_N	Negatif destek vektörü
SV_P	Pozitif destek vektörü
SV_S	Destek vektörleri
X^2	Ki-kare

Kısaltmalar

ACC	Accuracy
AI	Artificial Intelligence
AR	Auto-regressive
AW	Ödül Ağırlık
BiLM	Bidirectional Language Model
BiLSTM	Bidirectional Long Short-Term Memory
BoW	Bag of Words,
BPE	Byte Pair Encoding
CBOW	Continuous Bag-of-Words

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
DF	Document Frequency
DT	Decision Tree
FN	False Negative
FP	False Positive
IE	Information Extraction
IR	Information Retrieval
IRAD	Information Radius
JSD	Jensen-Shannon Divergence
KLD	Kullback-Leibler Divergence
KNN	K- Nearest Neighbour
LR	Logistic Regression
LSTM	Long Short-Term Memory
ML	Machine Learning
MLM	Masked Language Model
MT	Machine Translation
NB	Naive Bayes

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
NER	Named Entity Recognition
NLI	Natural Language Inference
NLP	Natural Language Processing
NN	Neural Network
NP	Noun Phrase
NSP	Next Sentence Prediction
OHE	One-Hot Encoding
OOV	Out-of-Vocabulary
OSH	Optimal Separating Hyper-Plane
P	Precision
POS	Part-of-Speech
PR	Paragraph Retrieval
QA	Question Answering
QAS	Question Answering Systems
R	Recall
RF	Random Forest

SİMGELER VE KISALTMALAR DİZİNİ (devam)

Kısaltmalar	Açıklama
SBERT	Sentence-BERT
SVM	Support Vector Machines
TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
TLM	Translation Language Modeling
TN	True Negative
TP	True Positive
VP	Verb Phrase

1. GİRİŞ

İnternet teknolojisinin hızla ilerlemesiyle birlikte günümüzde bilgiye ulaşmak için en çok tercih edilen yollardan biri arama motorlarıdır. Günümüzdeki arama motorları dikkat çekici özelliklere sahip olmasına rağmen, bilginin çokluğu ve kullanıcıların arayacakları bilgiler hakkında doğru veriler kullanmaması durumunda, arama motorları kullanıcının sorusuna doğru ve kesin bir cevap verememektedir. Bu nedenle, kullanıcı sorusuna doğru ve kesin cevabı otomatik olarak verebilecek araçlara ihtiyaç duyulmaktadır. Bu ihtiyacı karşılamak için geliştirilen ve Doğal Dil İşleme (Natural Language Processing - NLP) alanının alt uygulamalarından biri olan Soru Yanıtlama Sistemleri (Question Answering Systems - QAS), günümüzde başarılı bir şekilde kullanılmakta ve güncelliğini korumaktadır.

QAS, girdi olarak verilen sorudaki bilgi ihtiyacını belirleyerek, gerekli bilgileri (anahtar kelimeler, soru türü gibi) bulabilmekte ve bu bilgiyi çıkarabilmektedir. Çıkarılan bilgilerden bir cevap üreterek, sorudaki gerekliliklere göre bu cevabı kullanıcıya sunabilmektedir (Indurkhya and Damerau, 2010). Arama motorları, kullanıcı tarafından sorulan sorunun cevabını içeren web sitesi listesini sunarken; QAS, arama motorlarının aksine sadece istenilen bilgileri sunmaktadır. Kullanıcı, arama motorlarının sunmuş olduğu ilgili bağlantılara tek tek girerek, sorduğu sorunun cevabını kendisi bulmak zorunda kalmaktadır. Bu da kullanıcı için ekstra bir çabaya ve zaman kaybına neden olmaktadır. Ek olarak, sunulan web sitelerinde bazen istenilen bilgilere de ulaşılamamaktadır. Bu nedenle QAS, kesin cevabı bulmak ve kullanıcıya doğrudan cevabı sunmak için Belge Alımı (Document Retrieval) işlemini gerçekleştirerek, kullanıcıya düşen bu yükü azaltmayı amaçlamaktadır (Strzalkowski and Harabagiu, 2006).

Uzun zamandır çalışılan QAS'in sayısı, farklı alanlar, farklı veri kaynakları, farklı soru türleri ve farklı cevap biçimleri olduğu için oldukça fazladır (Mishra and Jain, 2016). Soru yanıtlama ile ilgili ilk çalışma, 1961 yılında Green ve arkadaşları tarafından, Amerika'da oynanan beysbol ligiyle ilgili tarih, yer vb. bilgileri sağlayan BASEBALL isimli bir çalışmadır (Green et al., 1961). Aslında QAS'in temeli, 1965 yılında Simmons tarafından yapılan İngilizce dilinde soru yanıtlama

girişimlerini inceleyen bir araştırma ile atılmıştır (Simmons, 1965). Ay'dan alınan toprak ve ay taşı numunelerinin kimyasal analizi hakkında bilgiler sunan LUNAR, kullanıcıların sorularını veri tabanı sorgularına dönüştürerek cevap üreten başka bir soru yanıtlama sistemidir (Woods, 1973). BASEBALL ve LUNAR sistemleri veri tabanları üzerinde çalışan kısıtlı ya da kapalı alan (restricted /closed domain) sistemleridir (Ojokoh and Adebisi, 2019).

1990'lı yılların sonlarına doğru TREC (Text REtrieval Conference) soru yanıtlama kayıtlarının kullanılmasıyla birlikte açık alan (open domain) soru yanıtlama araştırmaları başlamıştır (Indurkhya and Damerau, 2010; Mishra and Jain, 2016). Web teknolojisinin ilerlemesi ile birlikte kullanıcıların sorularını yanıtlayabilecek, webi bilgi tabanı olarak kullanan soru yanıtlama sistemleri geliştirilmiştir (Soricut and Brill, 2006; Li and Roth, 2002; Guda et al., 2011).

2010 yılında IBM Araştırma Grubu, soru yanıtlama alanında dönüm noktası sayılacak olan Watson (DeepQA) sistemini önermiş, soru yanıtlama testi karşısında gösterdiği başarı ile insan zekası seviyesini yakalamıştır (Ferrucci et al., 2010). Son yıllarda, Siri, Alexa gibi sesli soruları cevaplayan, tavsiyelerde bulunan akıllı asistanlar oldukça popülerdir. Ancak, soru yanıtlama alanında hala birçok zorluk bulunmakta ve QAS'in iyileştirilmesi için birçok çalışma yapılmaktadır.

NLP'de kullanılan verilerin yüksek boyutlu olması ve bu verilerin çok sayıda gereksiz öznitelik içermesi problem teşkil etmektedir (Aggarwal and Zhai, 2012). Soru yanıtlamada bu problemin üstesinden gelebilme için, Yapay Zekâ (Artificial Itelligence -AI)'nın alt dalı olan Makine Öğrenmesi (Machine Learning - ML) yöntemleri sıklıkla kullanılmaktadır. ML yöntemlerini kullanırken verileri temsil edecek özniteliklerin seçilmesi, yöntemlerin performansını doğrudan etkilediğinden öznitelik seçimi yüksek önem taşımaktadır. Kullanıcıların sormuş olduğu sorulara en doğru cevabı verebilmek için verileri mümkün olduğunca doğru bir biçimde temsil etmek ve etkin bir öznitelik seçimi uygulamak, tez çalışmasının temel hedefleri arasındadır.

QAS, soru analizi (question analysis), belge/paragraf alımı (document/passage retrieval) ve cevap çıkarma (answer extraction) olmak üzere üç

ayrı bileşenden oluşmaktadır. Soru sınıflandırma ise cevap türünü bulmamızı sağlayan soru analizinin bir alt görevidir. Cevabın türünü bilmek, kullanıcıya sunulacak olan cevabın kesinliğini belirleyeceği için QAS'de soru sınıflandırma önemli bir rol oynamaktadır. Soru sınıflandırma için temel sınıflandırma algoritmaları kullanılabileceği gibi, daha yüksek bir sınıflandırma başarımı elde edebilmek için topluluk öğrenmesi (ensemble learning) yöntemleri de kullanılmaktadır. Topluluk öğrenmesi yöntemlerinin yüksek bir başarım elde edebilmesi için birleştirilme şekli önem arz etmektedir (Zhang and Ma, 2012). Tez çalışması kapsamında, topluluk öğrenmesi yöntemleri kullanılarak temel sınıflandırma algoritmalarından daha yüksek başarıma sahip bir yöntem geliştirilmesi hedeflenmiştir. Ek olarak, derin öğrenme tabanlı öznitelik gösterimleri ve topluluk öğrenmesi yöntemleri kullanılarak hibrit etkin bir QAS mimarisi tasarlanması hedeflenmiştir. Bu çalışmada, soru sınıflandırma görevi için literatürde sıklıkla kullanılan TREC veri seti, soru yanıtlama görevi için SQuAD veri seti kullanılmıştır.

Tez metninin ikinci bölümünde, QAS ve mimarisi, kullanıcıların sormuş olduğu soru türleri, soru sınıflandırma problemi ve soru yanıtlama ile ilgili önceki çalışmalar hakkında bilgi verilmektedir.

Üçüncü bölümünde, tez çalışması için gerekli olan klasik öznitelik temsil yöntemlerinden, öğrenme tabanlı temsil yöntemlerinden, derin öğrenme tabanlı temsil yöntemlerinden, Makine Öğrenmesi yöntemlerinden ve topluluk öğrenme yöntemlerinden bahsedilmektedir.

Dördüncü bölümde, soru sınıflandırma için geliştirilen öznitelik seçimi ve hibrit yöntem ile QAS için geliştirilen soru yanıtlama mimarisi ayrıntılı olarak sunulmaktadır.

Beşinci bölümde, tez kapsamında yapılan deneysel çalışmalar hakkında bilgiler verilmekte ve geliştirilen yöntemler ile elde edilen sonuçlar sunulmaktadır.

Tezin son bölümünde ise tez çalışmasının katkılarına, sonuçlarına ve gelecekte bu alanda çalışacak olan araştırmacılar için önerilere yer verilmiştir.

2. SORU YANITLAMA SİSTEMLERİ

Bu bölümde Soru Yanıtlama Sistemleri'nin genel mimarisine, kullanıcıların sormuş olduğu soru türlerine, soru sınıflandırma problemi ve soru yanıtlama ile ilgili önceki çalışmalara değinilmiştir.

2.1 QAS ve Mimarisi

Soru yanıtlama (Question Answering - QA), Bilgi Alımı (Information Retrieval - IR) ve Bilgi Çıkarımı (Information Extraction - IE) alanlarının birlikte kullanıldığı bir NLP alt görevidir. Soru Yanıtlama Sistemleri (QAS), karmaşık doğal dil işleme tekniklerini, dilsel gösterimleri ve makine öğrenmesi yöntemlerini birleştirerek, yapılandırılmamış metin belgelerinde çeşitli doğal dil sorularına doğru cevaplar verebilen bir sistemdir (Strzalkowski and Harabagiu, 2006). QAS'in temel amacı, kullanıcıların sormuş olduğu sorulara beklenen cevap türüne göre hızlı ve kesin cevaplar sunabilmektir.

QAS, belgeden bilgi çıkarma, dil öğrenimi, çevrimiçi sınav sistemi, insan ve bilgisayar etkileşimi, belge yönetimi ve belgenin sınıflandırılması gibi cevap kaynağına dayanan birçok uygulamaya sahiptir (Pundge et al., 2016). Bu uygulamalarla ilgilenen araştırmacıların çoğu açık alan (open domain) ve sınırlı alan (restricted domain) soru yanıtlama sistemlerine odaklanmışlardır. Açık alan soru yanıtlamadaki sorular alandan bağımsız sorular olarak nitelendirilmektedir. Çok fazla belge içinde arama yapan bu sistemler, herhangi bir alan ile ilgili herhangi bir soruyu cevaplayabilmektedir (Lende and Raghuwanshi, 2016). Alana özgü sözlük ve anahtar kelime gerektirmemesi, soru havuzunun geniş olması, ve online gazete metinlerinin bilgi kaynağı olarak kullanılabilmesi bir avantaj oluştururken, alan belli olmadığı için üretilen cevapların doğruluğunun düşük olması bir dezavantaj oluşturmaktadır (Mishra and Jain, 2016). Sınırlı alan soru yanıtlamadaki sorular ise alana özgü sorular içermektedir (Molla and Vicedo, 2007). Cevaplar alana özgü belgeler içinde arandığı için ve soru modelleri sınırlı olduğu için üretilen cevapların doğruluğu yüksektir. Soru havuzunun sınırlı olmasından dolayı sınırlı sayıda soruya cevap verebilmektedir. Eğitim alanı soru yanıtlama sistemi, coğrafi alan soru yanıtlama sistemi, tıbbi alan soru yanıtlama sistemi, uzay bilimi soru

yanıtlama sistemi vb. sınırlı alan soru yanıtlama sistemlerine örnek olarak verilebilir. Farklı alanlardan olan birden fazla soru yanıtlama sistemi birleştirilerek açık alan soru yanıtlama sistemi elde edilebilmektedir (Lopez et al., 2011).

QAS kullanıcıların sordukları soruların türlerine göre 5'e ayrılmaktadır (Mishra and Jain, 2016):

1. ***Gerçek Tabanlı (Factoid) Sorular:*** Genellikle "Wh-" soru kelimesi ile başlayan ve cevabı tek kısa bir cümle parçası yada cümle olan sorulardır. İyi bir performans elde edilmesi sağlayan Gerçek tabanlı sorularda, doğal dil işlemlerini uygulamaya gerek yoktur.

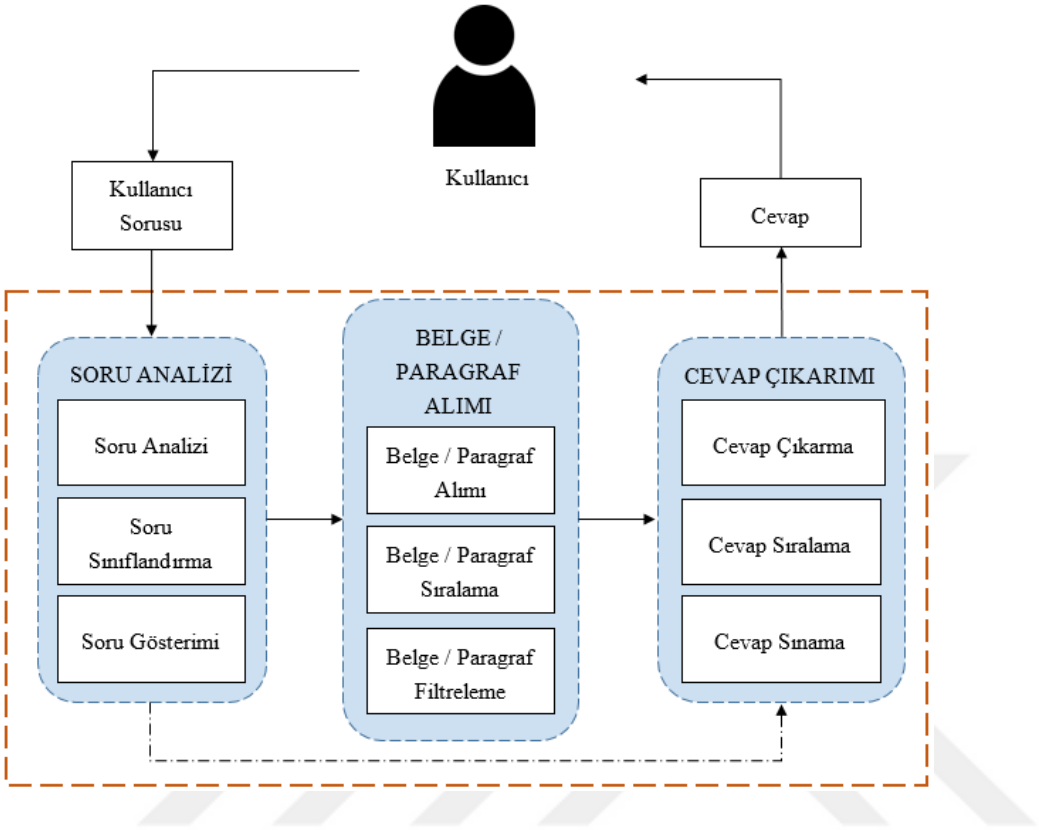
2. ***Liste Tipi Sorular:*** Cevabı bir olgu veya varlık listesi olan bu sorular, soru yanıtlama sistemlerinde art arda on kez sorulan gerçek tabanlı sorular olarak ele alınırlar (Indurkhya ve Damereau, 2010). Bu soru tipinde doğruluk oranı iyi olmasına rağmen, sorulan varlığın sayısı için eşik değerinin sabitlenmesinde problemler yaşanmaktadır.

3. ***Varsayımsal (Hypothetical) Sorular:*** Genellikle *"what would happen if"* ile başlayan (Kolomiyets and Moens, 2011) ve varsayımsal bir olayla ilgili bilgi isteyen sorulardır. Cevaplar, kullanıcılara ve içeriğe bağlı olduğu için soru yanıtlama sistemlerinin doğrulukları da düşüktür.

4. ***Onay (Confirmation) Soruları:*** Cevabı evet ya da hayır olan onay soru tipleri için çıkarım mekanizması, bilgi birikimi ve sağduyuya dayalı akıl yürütme gereklidir (Mishra and Jain, 2016).

5. ***Nedensel (Causal) Sorular:*** İleri doğal dil işleme tekniklerine ihtiyaç duyan nedensel sorular (Higashinaka and Isozaki, 2008), belirli bir olay veya varlık hakkında açıklama, neden ve detaylandırma gerektirirler. Bu tip sorular yoruma dayalı sorular olduğu için, ve çok anlamlı ve eş anlamlı kelimeler olmasından kaynaklı bilgi çıkarma sürecinde karşılaşılan problemlerden ötürü uygulanması zordur.

Tipik bir QAS üç ayrı alt görevden oluşmaktadır (Şekil 2.1): Soru analizi, belge/paragraf alımı ve cevap çıkarımı.



Şekil 2.1. Soru Yanıtlama Sistemleri'nin genel mimarisi.

2.1.1 Soru analizi

QAS'nin ilk aşaması olan soru analizi, sisteme girdi olarak verilen doğal dil sorusundan önemli olan bilgileri tespit eden ve çıktı olarak soru temsilini veren önemli bir safhadır. Bu bilgiler tespit edilirken, beklenen cevap doğrultusunda sorunun sınıflandırılması (kişi, yer, saat, tarih, miktar vb.) önem arz etmektedir (Cortes et al., 2020). Çünkü, cevabı içeren belgeler, bu bilgiler doğrultusunda aranmaktadır. Bu bilgilerin yanlış tespit edilmesi, beklenen cevabın türünü değiştireceğinden sistemin performansını olumsuz etkileyecektir. Sistemdeki hataların %36.4'ünün beklenen cevap türünün doğru sınıflandırılamamasından kaynaklandığı tespit edilmiştir (Moldovan et al., 2003). Aynı zamanda beklenen cevap türünün bilinmesi; bizi, veri kaynağındaki belgelerin veya paragrafların her birini incelemekten kurtaracaktır.

Soru analizi safhası, sorunun ayrıştırılması, sınıflandırılması ve temsil edilmesi işlemlerini içermektedir (Dwivedi at al., 2013; Saini et al., 2017; Ojokoh and Adebisi, 2019). Soru sınıflandırma veya cevap türü tanıma işlevi; sınıflandırılmış cevap türünü, adlandırılmış varlığı veya cevabı kategorize eden benzer bir sınıf belirtmektedir. Sorunun sınıflandırılmasında araştırmacılar; içerik tabanlı sınıflandırma (Li and Roth, 2002), Model Eşlemesi (Pattern Matching) ve makine öğrenmesi tekniklerini birleştirerek gerçekleştirilen işlev odaklı sınıflandırma (Bu et al., 2010), beklenen cevap türlerine göre sınıflandırma (Benamara, 2004) gibi birden çok yöntem kullanmaktadır. Zhang and Lee (2003), soruların sınıflandırılması için, Destek Vektör Makinesi (Support Vector Machine), Karar Ağaçları (Decision Trees), En Yakın Komşu (Nearest Neighbors) ve Naif Bayes (Naive Bayes) makine öğrenmesi yöntemlerini kullanmıştır.

Soru içindeki bilgilerin çıkarılması, anahtar kelime seçimi ve cevap modeli oluşturma yolu ile yapılmaktadır. Anahtar kelime seçimi, aday cevabı işaret eden soru terimlerinden seçilerek gerçekleştirilmektedir. Cevap modeli oluşturma ise her biri cevap olabilecek, soru terimlerinin kombinasyonlarından çeşitli sorgular elde edilerek gerçekleştirilmektedir. Literatürde bu iki yöntemi birleştirerek kullanan çalışmalar da bulunmaktadır.

2.1.2 Belge veya paragraf alımı

QAS'in ikinci safhası olan belge/paragraf alımı, soru analizi safhasından elde edilen bilgiler (soru/cevap türü, anahtar kelimeler vb.) doğrultusunda, IR ve Paragraf Alımı (Paragraph Retrieval - PR) sistemlerini kullanarak aday metinler elde etmektedir. Bu sayede, tüm belge tabanı yerine, gerekli cevabı içeren metin kümesi oluşturulmaktadır. Bir önceki aşamada elde edilen beklenen cevap türü ile eşleşmeyen metinler filtrelenmektedir.

Çoğu cevap çıkarma yöntemi paragraf gibi küçük metin parçalarını işlemeye uygun bir şekilde tasarlanmıştır (Jurafsky and Martin, 2021). Bu nedenle, bölüm, paragraf ve cümle gibi daha küçük metin parçaları elde edebilmek için, bu belgeler bir paragraf bölümleme algoritması ile daha küçük metin parçalarına ayrıştırılmaktadır. Klasik bir paragraf alma yöntemi, belgeyi paragraflara ayırarak,

her bir paragrafa bir skor değeri vermektedir. En yüksek skora sahip olan paragraflar döndürülerek bir filtreleme işlemi gerçekleştirilmektedir (Pundge et al., 2016). Paragraflar, NER etiketçisi kullanılarak veya cevap türünü içermeyen bölümler atılarak da filtrelenebilmektedir.

Filtrelenen paragrafları sıralamak için, aşağıda verilen öznitelikleri kullanarak Denetimli Öğrenme Algoritmalarını kullanmak da mümkündür (Jurafsky ve Martin, 2021).

- Paragraftaki soru anahtar kelimelerinin sayısı,
- Paragraftaki doğru Adlandırılmış Varlıkların sayısı,
- En uzun soru anahtar kelime dizisini içermek,
- Paragrafların çıkartıldığı belgenin sırası,
- Anahtar kelimelerin gerçek sorguda birbirlerine yakınlığı (Pasca 2003; Monz 2004),
- Paragraf ve soru arasında örtüşen n-gram sayısı (Brill at al., 2002),

Belge veya Paragraf Seçimi safhasının performansının iyi olması, döndürülen belgelerden veya paragraflardan nihai cevabın çıkarılması için kritik bir konudur. Önceki safhadan çıkarılan her bir bilgi parçası, döndürülen paragraflar içinde aranacağı için doğru cevabın tespit edilmesini doğrudan etkilemektedir.

2.1.3 Cevap Çıkarımı

QAS'in son safhası olan cevap çıkarımı, bir önceki safhadan elde edilen belgeler ile soru gösterimini eşleştirerek, aday cevapları doğruluk olasılığına göre sıralamaktadır. Daha basit bir yöntem ise adlandırılmış varlık etiketçisi (Jurafsky and Martin, 2021) kullanmaktır. Bir önceki aşamadan elde edilen belgeler üzerinde bu etiketçi çalıştırılarak, beklenen cevap türü ile eşleşmeyen varlıkların kaldırılması sağlanmaktadır. Cevap adaylarını sıralamak için kullanılan bazı yöntemler şunlardır (Clark et al., 2010; Jurafsky and Martin, 2021):

- ***Benzerlik:*** Soruyla çok benzer olan ve beklenen cevap türü ile uyumlu bir dize içeren cevap adaylarının sıralanması işlemidir. Bu işlem için, ortak

sözcük sayısı, sözdizimsel (syntactic) veya anlamsal (semantic) bilgiye dayalı benzerlik ölçümleri kullanılmaktadır.

- ***Popülerlik:*** Cevap adayları içinde en çok tekrarlanan cümlenin cevap olma ihtimalinin yüksek olduğu düşündeşine dayanan ve bu nedenle en çok tekrarlayan cevap adaylarını sıralayan yöntemdir.

- ***Modeller:*** Tam cevabı çıkarabilmek için soruya özgü modeller geliştirilerek, bu modeller ile eşleşen cevap adaylarının sıralanması işlemidir. Belirli sorulara cevap veren ifadeler incelenerek el ile oluşturulan bu modeller; karakter, noktalama işareti, bir kelime dizisi vb. olarak tanımlanmaktadır. Adaylar arasından seçim yapılabilmesi için oluşturulan modellere, soru ile eşleşmesine bağlı olarak bir kesinlik değeri (certainty value) atanmaktadır. Modellerin el ile hazırlanması ve alana bağımlı olması soru yanıtlama sistemleri için bir dezavantaj oluşturmaktadır.

- ***Cevap doğrulama:*** Beklenen cevap türü ile uyumlu adlandırılmış varlıkların bulunduğu cevap adaylarının sıralanması işlemidir.

2.2 Önceki Çalışmalar

Tezin bu bölümünde soru sınıflandırma problemi ve soru yanıtlama ile ilgili önceki çalışmalar kısaca tanıtılmaktadır.

Soru sınıflandırma veya cevap türü tanıma işlevi; sınıflandırma stratejisine bağlı olarak doğal dilde verilen bir soruya bir ve ya daha faza sınıf etiketi atamaktır. Örneğin; “Ege Üniversitesi nerededir? (Where is the Ege University?)” sorusunun cevabı, konum türünde bir varlık olduğu için “Konum (Location)” etiketi soruya atanmaktadır.

Soru sınıflandırması için yapılan araştırmalarda araştırmacılar; kural tabanlı yaklaşımlar, öğrenme tabanlı yaklaşımlar ve hibrit yaklaşımlar olmak üzere üç farklı yaklaşım kullanmışlardır (Hao et al., 2015; Razzaghnoori et al., 2018). Soru sınıflandırması için yapılmış ilk çalışmalar çoğunlukla kural tabanlı yaklaşımı

kullanmışlardır. Hatta öğrenme tabanlı soru sınıflandırma yaklaşımları, öznitelik sağlayıcı olarak kural tabanlı yaklaşımları kullanmaktadır.

Haris and Omar (2012) tarafından yapılan çalışmada, kurallar geliştirerek BloomTaksonomisi'ne göre, yazılı sınav sorularını analiz edilerek, soruları sınıflandırmış ve öğretim elemanlarının, öğrencilerin bilişsel düzeylerini değerlendirmelerini kolaylaştırmayı amaçlamıştır.

Öğrenme tabanlı soru sınıflandırma yaklaşımları, soruların bazı özniteliklerini çıkararak soruları sınıflandırmaya çalışan yaklaşımlardır. Sınıflandırıcıyı eğiterek, sorunun sınıf etiketini belirlemektedirler. Son zamanlarda yapılan çalışmaların büyük çoğunluğu denetimli öğrenme yaklaşımına dayanmaktadır.

Zhang and Lee (2003) tarafından yapılan çalışmada, sorunun sözdizimsel yapısına dayanılarak oluşturulmuş bir ağaç çekirdek (tree kernel) tanımlanmıştır. Verilen soru ilk olarak sözdizimsel ağaca ayrıştırılmış ardından soru, sözdizim ağacının alt ağaçları olan ağaç parçalarına dayalı olarak temsil edilmiştir. Çalışmada, öznitelik vektörünü, yüksek boyut uzayına eşleyen özel bir çekirdek fonksiyonu tanımlamışlardır. Aynı öznitelik uzayı üzerinde Destek Vektör Makinesi (SVM) ile En Yakın Komşu , Naif Bayes (NB), Karar Ağacı (DT) ve SNoW yaklaşımları karşılaştırılmıştır. TREC veri setindeki kaba taneli sınıflar için en iyi sonucu veren yöntem %90 başarı ile SVM olmuştur. Özellik uzayına ve diğer parametrelere bağlı olarak en iyi sınıflandırıcının değişebileceği, yapılan çalışmada vurgulanmıştır.

Metzler and Croft (2005) yaptığı çalışmada, SVM ile radyan temelli fonksiyonu kullanmıştır. Çalışmada ilk olarak soru kelimeleri tanımlanmıştır. Her soru kelimesine özgü, ayrı bir sınıflandırıcı eğitilmiştir. Sınıflandırmanın performansını artırmak için anlamsal öznitelikler olarak WordNet kullanılmıştır. TREC ve MadSci veri seti üzerinde test edilen çalışmada, TREC veri setindeki kaba taneli sınıflar için %90.2 doğruluk, ince taneli sınıflar için %83.6 doğruluk ve MadSci veri setinde ise %73.2 doğruluk elde edilmiştir.

Pan et al. (2008), anlamsal öznitelikleri kullanarak ağaç parçalarının anlamsal benzerliklerini ölçüp, anlamsal bir ağaç çekirdek elde etmişlerdir. TREC veri seti üzerinde %94 doğruluk elde edilmiştir.

Tomas ve Guliano (2009), etiketlenmemiş metin kullanılarak elde edilen soru sınıfları için anlamsal bir çekirdek tanımlamıştır. Gizli anlamsal bir çekirdek tanımlayarak öznitelik uzayını çok daha etkili bir uzaya indirgemek için Gizli Anlamsal İndeksleme (Latent Semantic Indexing) kullanılmıştır.

Huang et al. (2009), doğrusal çekirdekli SVM ile TREC veri setinin kaba taneli sınıfları üzerinde %93.4, ince taneli sınıfları üzerinde %89.2 doğruluk elde etmiştir.

Silva et al. (2011), SVM'i doğrusal çekirdek ile birlikte TREC veri seti üzerinde kullanıp, kaba taneli sınıflar için %95, ince taneli sınıflar için %90.8 doğruluk elde etmişlerdir. Bu doğruluk değeri o ana kadar elde edilen en yüksek doğruluk değeridir.

Zhang et al. (2010) tarafından yapılan çalışmada, Çince üzerinde sözcük türü etiketi (part-of-speech tag – POS) özniteliklerini, soru sınıflandırma ve anahtar kelime çıkarmak için SVM ile birlikte kullanmıştır.

Arapça QAS için iki aşamalı bir sistem öneren Al-Shenak et al. (2019), ilk aşamada, soru sınıflarını SVM yöntemi ile bulurken, ikinci aşamada cevabın geçtiği paragrafları bulabilmek için gizli anlamsal indeksleme kullanmıştır.

Pratiwi and Syukriyah (2019) tarafından yapılan çalışmada, e-öğrenme sistemleri için soruları konularına göre otomatik bir şekilde alt kategorilere ayrıştırabilen makine öğrenmesine dayalı bir yöntem önerilmiştir. Çalışmada NB, DT ve K-ortalamalar algoritmaları kullanılmış ve en yüksek başarımı %73.58 ile NB algoritması vermiştir.

Huang et al. (2008), makine öğrenmesi tabanlı bir sınıflandırıcıya anlamsal öznitelikler sağlamak için bir kural tabanlı soru sınıflandırıcısı kullanan hibrit bir model önermiştir. Çalışmada, önceden tanımlanmış bazı kurallar ile sorular

eşleştirilmiş, eşleşen kurallar öğrenme tabanlı sınıflandırıcıya öznitelik olarak verilmiştir.

Aniol et al. (2019), soru sınıflandırma problemi için dikkat mekanizmasına dayanan modelleri topluluk öğrenmesi ile birlikte kullanmıştır. Modeller, SQuAD veri seti üzerinde test edilmiş ve deneysel çalışmalar neticesinde en iyi performansı Mnemonic Reader modeli sergilemiştir.

Devlin et al., (2019) tarafından önerilen BERT, kullanıcıların arama sorgularının daha iyi analiz ederek, kullanıcılara en doğru cevabın döndürülmesini amaçlamaktadır. Transformer mekanizmasına dayanan BERT, soru yanıtlama görevi için SQuAD v1.1 veri seti üzerinde %91 F1-skor elde etmiştir.

Yang et al. (2019), BERT modelinden esinlenerek, BERT'in eğitim aşamasındaki dezavantajını ortadan kaldırmak amacıyla daha büyük girdi dizisine sahip XLNet'i önermiştir. Soru yanıtlama görevi için SQuAD v1.1 veri seti üzerinde BERT'ten daha iyi sonuçlar alarak %95 F1-skor elde etmiştir.

Liu et al. (2019) tarafından önerilen RoBERTa, BERT modelini iyileştirmek için BERT'in ön-eğitim safhası üzerinde iyileştirmelerde bulunmuştur. RoBERTa, SQuAD v2.0 veri seti üzerinde soru yanıtlama için %89 başarıya ulaşmıştır.

Lan et al. (2019), ALBERT ismini verdikleri çalışmada, diğer modellerden daha hızlı ve daha başarılı olmak amacıyla, diğer modellerin kullandığı parametre sayısını azaltan yeni bir model önermiştir. Model, SQuAD v2.0 veri seti üzerinde soru yanıtlama için %80 başarıya ulaşmıştır.

3. TEMEL YÖNTEMLER

Bu bölümde, tez kapsamında kullanılan öznitelik temsili ve yöntemleri, sınıflandırma algoritmaları ve topluluk öğrenme yöntemleri ayrıntılı bir şekilde sunulmaktadır.

3.1 Öznitelik Temsili

Makine öğrenmesi ve derin öğrenme gibi yöntemler, ham metin verileri üzerinde doğrudan işletilemezler. Yöntemlerin uygulanabilmesi için, metinlerin uygun bir biçimde temsil edilmesi gerekmektedir.

Metin verileri, karakter dizisi veya kelime dizisi olarak düşünülebilir. Metin verilerinde yer alan, yazma ve konuşmanın en küçük anlamlı parçası olan kelimeler, cümle içinde kullanımlarına göre farklı anlamlar kazanabilmektedir. Benzer bağlamlarda (*context*) geçen kelimeler, benzer anlamlara sahip olma eğilimindedirler (Jurafsky and Martin, 2021). Kelimelerin anlamsal bilgilerinin çıkarılması, metinlerin işlenmesindeki en kritik noktalardan biridir. Kelimelerin sayısal ifadelere dönüştürülerek işlenebilir hale gelmesi "kelime gömülmesi (*word embedding*)" olarak adlandırılmaktadır.

3.1.1 Frekans tabanlı kelime gömülmeleri

Frekans tabanlı kelime gömülmesi, belge içerisinde bulunan kelimelere ve bu kelimelerin frekanslarının tespit edilmesi işlemine dayanan bir gösterilim şeklidir. Kategorik ve ağırlıklı kelime gösterimi olarak iki çeşidi mevuttur (Naseem et al., 2021).

3.1.1.1 Kategorik kelime gösterimi

Metini temsil etmenin en basit yolu olan kategorik kelime gösteriminde kelimeler, ikili olarak ("1" ve "0" değerleri ile) temsil edilir. Bir-elemanı-bir kodlanmış (One-Hot Encoding - OHE) vektör ve Kelime Çantası (Bag of Words – BoW) yaklaşımı kategorik kelime gösterimine örnektir.

- **Bir-elemanı-bir kodlanmış vektör:** OHE gösterimi, kelimeleri tanımlamak için kullanılan, kelimelerin ikili olarak temsil edilmesini sağlayan en yaygın ve en basit yöntemdir. Vektörün boyutu, elimizdeki faklı kelimelerin sayısı kadardır. OHE gösterimi için öncelikle, her bir kelimeye benzersiz bir indeks atanır. Vektörde sadece ilgili kelimenin bulunduğu indeksin değeri 1, diğer tüm kelimelerin indekslerinin değeri 0 olarak verilir. Bunu bir örnek ile açıklayacak olursak; "*Daisy will learn to read soon"* ve *"Jack will learn to drive*" şeklinde iki cümlemiz olduğunu farz edelim. $V = \{\text{Daisy}, will,\ learn, to, read, soon, Jack, drive\}$, iki cümlede geçen benzersiz kelimelerden oluşan sözcük dağarcığıdır (vocabulary). OHE gösterimi için öncelikle her iki cümlede geçen kelimelere indeks atanması yapılır. Kelime indeksleri, *"Daisy": 1, "will":2, "learn": 3, "to": 4, "read":5, "soon":6, "Jack":7, "drive":8* şeklindedir. Burada her iki cümlede benzersiz sekiz kelime olduğu için, OHE gösterimi ile oluşturulan öznitelik vektörünün uzunluğu 8 olarak alınır. İki cümle için elde edilen öznitelik vektörleri aşağıda belirtildiği gibidir:

  1. cümle: [1 0 0 0 0 0 0 0], [0 1 0 0 0 0 0 0], [0 0 1 0 0 0 0 0], [0 0 0 1 0 0 0 0], [0 0 0 0 1 0 0 0], [0 0 0 0 0 1 0 0], [0 0 0 0 0 0 0 0], [0 0 0 0 0 0 0 0]

  2. cümle: [0 0 0 0 0 0 1 0], [0 1 0 0 0 0 0 0], [0 0 1 0 0 0 0 0], [0 0 0 1 0 0 0 0], [0 0 0 0 0 0 0 1], [0 0 0 0 0 0 0 0], [0 0 0 0 0 0 0 0], [0 0 0 0 0 0 0 0]

- **Kelime çantası yaklaşımı:** OHE gösteriminde, sözcük dağarcığının çok fazla olduğu durumlarda, vektörün uzunluğu artacağı için ve bu durum çok sayıda "0"dan oluşan seyrek bir matris ile sonuçlanacağından bir dezavantaj oluşturmaktadır. Bu nedenle, OHE'nin daha da geliştirilmesiyle BoW yaklaşımı ortaya atılmıştır. BoW yaklaşımı, bir dokümandaki tüm kelimeleri öznitelik olarak ele alan ve kelimelerin bu dokümandaki sayısını ifade eden bir yaklaşımdır. BoW ile öznitelik vektörü oluşturulurken, kelimelerin sırasına, yapısına ve kelimeler arası

ilişkilere önem verilmez. OHE'de olduğu gibi, dokümandaki benzersiz kelimelerden oluşan bir sözcük dağarcığı oluşturulur. Ardından, sözcük dağarcığındaki kelimelere indeks ataması yapılır. Her bir kelimenin dokümanda geçme sayısı vektör olarak ifade edilir. Birden fazla doküman olduğu durumlarda, doküman içerisindeki benzersiz kelime sayısı kadar sütun ve belge sayısı kadar satır içeren bir matris oluşturulur. d $=\{d_1, d_2, \dots, d_M\}$ doküman kümesi; M, derlemdeki (corpus) doküman sayısı; N, bu dokümanladaki benzersiz kelime sayısı; $w = \{w_1, w_2, \dots, w_N\}$, d dokümanlar kümesine ait kelime kümesi; f_i, d_i dokümanına ait öznitelik vektörü ve f_{ij}, w_j kelimesinin d_i dokümanında görülme sayısı olmak üzere, BoW öznitelik vektörü $f_i = [f_{i1}, f_{i2}, \dots, f_{iN}]$ şeklindedir. OHE'de verilen "*Daisy will learn to read soon"* ve *"Jack will learn to drive*" örneğini ele alalım. Sözcük dağarcığı, $V = \{\text{Daisy}, will, learn, to, read, soon, Jack, drive\}$ ve kelime indeksleri, *"Daisy": 1, "will":2, "learn": 3, "to": 4, "read":5, "soon":6, "Jack":7, "drive":8* olmak üzere, 8 uzunluklu öznitelik vektörü;

"Daisy will learn to read soon": [1 1 1 1 1 1 0 0]
"Jack will learn to drive" : [0 1 1 1 0 0 1 1]

şeklinde ifade edilir.

N-gram gösterimi, metinleri sayısal olarak ifade etmek için kullanılan frekans tabanlı başka bir yaklaşımdır. Bir cümleden çıkarılabilecek olan n veya daha az sayıda birbirini takip eden kelime grubunu ifade eder. En sık karşılaşılan N-gram modelleri, 1-gram (unigram), 2-gram (bigram/digram), 3-gram(trigram)'dır. BoW gösterimi aynı zamanda 1-gram olarak da adlandırılmaktadır. "*Daisy will learn to read soon"* cümlesinin en sık kullanılan N-gramlar ile gösterimi aşağıdaki gibidir:

- ✓ $1-gram = \{"Daisy", "will", "learn", "to", "read", "soon"\}$
- ✓ $2-gram =$
  $\{"Daisy\ will", "wiil\ learn", "learn\ to", "to\ read", "read\ soon"\}$
- ✓ $3-gram\ =$
  $\{"Daisy\ will\ learn", "will\ learn\ to", "learn\ to\ read",$
  $"to\ read\ soon"\}$

Bu gösterim şekli, kelime bazında yapılabildiği gibi, karakter bazında da yapılabilmektedir.

3.1.1.2 Ağırlıklı kelime gösterimi

Kategorik kelime gösterimine dayanan ağırlıklı kelime gösterimi, kelimelerin dokümandaki sayısına bakmak yerine, kelimenin frekansını referans alır. Ağırlıklı kelime gösteriminde sıklıkla kullanılan yöntemler, Terim Frekansı (Term Frequency - TF) ve Terim Frekansı-Ters Doküman Frekansı (Term Frequency-Inverse Document Frequency - TF-IDF)'dir.

- **Terim frekansı (TF):** Kelimeleri sayısal olarak ifade edebilmek için kullanılan en basit yöntemlerden biri olan TF, bir doküman içerisinde geçen bir kelimenin ağırlığını göstermektedir. Bir kelime, daha uzun dokümanlarda daha çok tekrar edeceği için TF değeri, ilgili kelimenin tekrar sayısının dokümandaki kelime sayısına bölünmesiyle hesaplanır. Yapılan bu işlem ile her bir kelimenin, ilgili dokümandaki önemi belirlenmiş olur. d dokümanında w_j kelimesinin tekrar sıklığı $TF(w_j, d)$ olmak üzere;

$$TF(w_j, d) = \frac{c_j(d)}{N} \tag{3.1}$$

şeklinde hesaplanır. Eşitlik 3.1'de, d dokümanındaki kelime sayısı N ile, d dokümanında w_j kelimesinin geçme sayısı $c_j(d)$ ile ifade edilir.

- **Terim frekansı – Ters doküman frekansı (TF-IDF):** Birden fazla doküman olduğu durumlarda TF gösterimi ile, dokümanlar içerisinde sıkça tekrar eden ve bu nedenle yüksek ağırlık değerine sahip olan ancak, belirleyici bir öneme sahip olmayan kelimeler (edatlar, bağlaçlar gibi etkisiz kelimeler) bulunmaktadır. IDF gösterimi, bu dezavantajı ortadan kaldırmak için geliştirilmiş bir gösterim yöntemdir. IDF, birden fazla dokümanda kelimenin tekrar sayısını bularak, bu kelimenin etkisiz bir kelime olup olmadığını tespit etmeye çalışır.

Doküman frekansı (Document Frequency - DF), TF ile oldukça benzer olup, ilgili kelimenin geçmiş olduğu doküman sayısını ifade eder. w_i kelimesinin

görüldüğü döküman sayısı, $DF(w_j)$ ile temsil edilir. d doküman kümesindeki herhangi bir w_j kelimesi için $IDF(w_j, d)$ Eşitlik 3.2'deki gibi hesaplanır.

$$IDF(w_j, d) = \log \frac{M}{DF(w_j)} \tag{3.2}$$

IDF gösterimi ile, dokümanda sıklıkla tekrar eden kelimelere veya en az tekrar eden kelimelere daha fazla ağırlık değeri verilir ve bu değerler $[0 - 1]$ aralığında değişmektedir. IDF gösterimi, TF gösterimi ile normalleştirilerek, sıklıkla tekrar eden kelimelerin etkisini azaltmak için birlikte kullanılmıştır. $TF - IDF(w_j, d_i, d)$ gösterimi;

$$\text{TF-}IDF(w_j, d_i, d) = TF(w_j, d) * IDF(w_j, d) \tag{3.3}$$

şeklinde ifade edilir. Eşitlik 3.3'te, d doküman kümesi; d_i, d doküman kümesindeki bir doküman ve w_j, d doküman kümesindeki herhangi bir kelime olarak temsil edilmiştir. Doküman kümesi içerisinde daha az tekrar eden ancak, belirli bir belgede daha çok tekrarlayan kelimelere TF-IDF gösterim şekli ile erişilebileceği için önemli olan kelimeler ayrıştırılabilmektedir.

Frekans tabanlı kelime gösterimlerini özetlemek gerekirse, anlaşılması, uygulanması basit olan gösterim şeklidir. Ancak, kullanılan gösterim şekline göre satırlarda veya sütunlarda benzersiz kelimeler bulunduran frekans tabanlı kelime gösterimi, kelime sayısı boyutunda matrisler oluşturur. Bu kelime gösterimi, elde edilen matristeki birçok değer 0 olacağı için seyrek matris (sparse matrix) olarak adlandırılan bir matris ortaya çıkarmaktadır. Bu bir dezavantajdır (Aydoğan ve Karcı, 2019). Derlem büyüdükçe, bu gösterim şeklinde gösterim büyüklüğü de artacağı için bellekte kısıtlamalara neden olabilmektedir. Ek olarak, frekans tabanlı kelime gösterimleri ile, tüm kelimeler birbirinden bağımsız olarak ele alındığı için kelimelerin bağlamı ve kelimeler arasındaki anlamsal yakınlıklar tespit edilememektedir (Şahin, 2017).

3.1.2 Öğrenme tabanlı kelime gömülmeleri

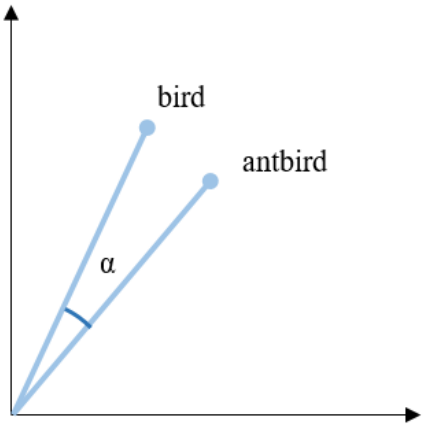
Bölüm 3.1.1'de bahsedilen frekans tabanlı kelime gömülmesinin dezavantajlarından (dokümanlardaki kelimelerin gözlemlenebilen özelliklerinin dikkate alınması, konum ve sırasının dikkate alınmaması ve bu nedenle anlam bütünlüğünün bozulması) kurtulabilmek için daha düşük boyutta ve daha yoğun temsil vektörleri kullanan yaklaşımlar önerilmiştir.

3.1.2.1 Kelime gömülmesi (Word embedding)

Temeli 2003 yılına dayanan (Bengio et al., 2003) kelime gömülmesi, vektör uzayında ifade edilebilen, aynı anlama gelen veya anlamca yakın olan kelimeleri vektör uzayında birbirine yakın şekilde temsil eden bir yaklaşımdır. Örneğin, "bird" kelimesini temsil ederken, "bird" kelimesine en yakın olan kelimeler "antbird, seabird, sparrow, finch, pelican, raptor" gibi kelimeler olacaktır. Bu yaklaşım türünde, birbirinden farklı iki kelime birlikte göründüğünde, kelime gömülmeleri birbirlerinden etkilenmektedir. Başka bir deyişle, frekans tabanlı kelime gömülmelerinin aksine, kelimeler birbirinden bağımsız olarak ele alınmamaktadır. Kısacası, kelime gömülmesi, kelimeler arasındaki ilişkileri de öğrenmektedir. Bir kelime hakkındaki bilgi, temsil edildiği kelime gömülmesi boyunca dağıtılmaktadır. Ek olarak, iki kelimenin kelime gömülmeleri arasındaki fark, benzer bir kelimenin kelime gömülmesini bulabilmek için başka bir kelimenin gömülmesine eklenebilir. vec(cow) – vec(calf) + vec(sheep) ≈ vec(lamb) örneğinde olduğu gibi, "cow" ve "calf" arasındaki ilişki kullanılarak, "sheep" ile benzer ilişkiye sahip olan kelimelerin gömülmeleri ele alındığında, "lamb" kelimesinin gömülmesi, diğer kelime gömülmelerine nazaran "sheep" kelime gömülmesine daha yakın olacaktır.

Frekans tabanlı kelime gömülmeleri, her kelime başına bir değer elde ederken; tahminleme tabanlı yaklaşımlar, her kelime için bir vektör elde etmektedir. Word2Vec, GloVe, FasText gibi yaklaşımlar, literatürde sıkça kullanılan kelime gömülme yaklaşımlarıdır.

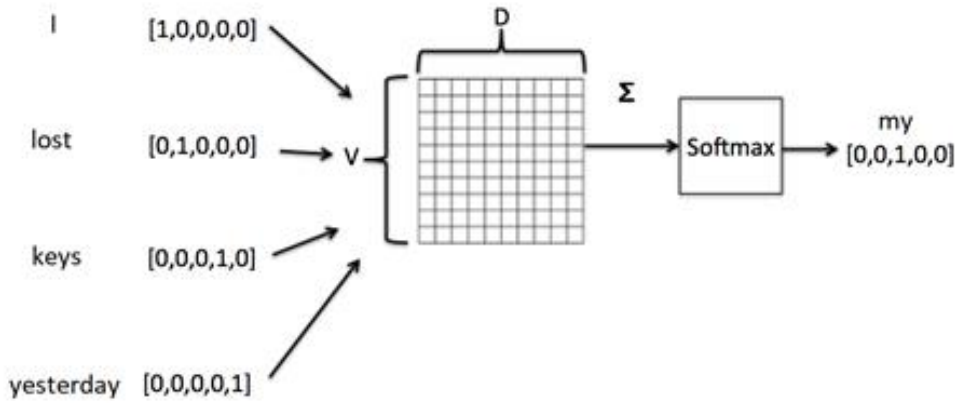
- **Word2Vec:** Daha düşük maliyetli, anlamsal ve sözdizimsel bilgileri içeren kelime temsilleri elde edebilmek için, Google araştırmacıları Mikolov ve arkadaşları (2013a) tarafından ortaya atılmış, girdi, çıktı ve gizli olmak üzere üç katmandan oluşan yapay sinir ağı temelli bir kelime gömülmesidir. Ağa girdi olarak bir derlem verildiğinde, bu eğitim verilerine bağlı olarak sözcük dağarcığı oluşturulur ve her bir kelime için, kelimeyi en iyi temsil eden benzersiz vektörler üretilir. Bu benzersiz vektörler, aynı anlama gelen kelimeler için birbirlerine yakın bir şekilde ifade edilirken, zıt anlamlı kelimeler için birbirlerinden çok farklı vektörler ile ifade edilmektedir. Şekil 3.1'de görüldüğü gibi, birbirlerine anlamca yakın olan kelimeler arasındaki α açısı sıfıra yakın olmalıdır.



Şekil 3.1. Vektör uzayında "bird" ve "antbird" kelime gömülmeleri arasındaki ilişki.

Word2Vec ile kelime gömülmeleri elde edilirken, pencere (window) olarak adlandırılan bir yapı kullanılır. Bu pencereye bir boyut verilerek derlem üzerinde kaydırılır ve hedef kelimelerin temsilleri çıkarılır. Bu kaydırma işlemi, derlem tamamlanana kadar devam eder. Pencere boyutu, hedef kelimenin sağında ve solunda olması gereken kelime sayısını belirtmektedir. Word2Vec, Sürekli kelime çantası (Continuous Bag-of-Words - CBOW) ve Skip-gram olmak üzere iki temel mimariye sahiptir. CBOW mimarisinde, hedef kelimenin etrafındaki kelimeler (pencere boyutu kadar) ağa OHE gösterimi ile kodlanarak girdi olarak verilir ve bu kelimeler ile görülme olasılığı en yüksek olan kelime tahmin edilmeye çalışılır. Şekil 3.2'de, "*I lost my keys yesterday*" derlemi için pencere boyutu "2" olan CBOW mimarisi

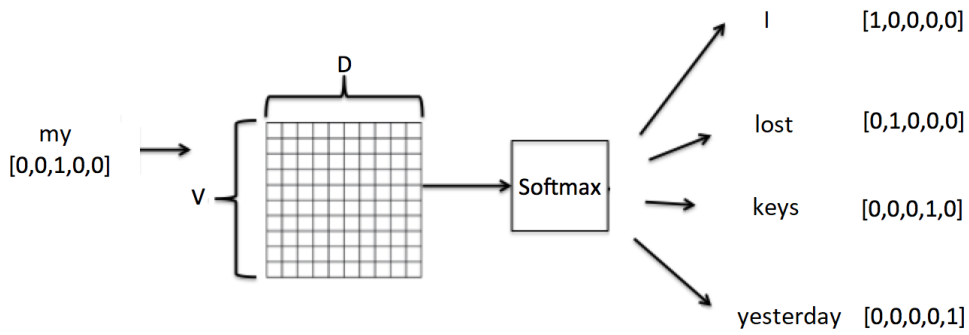
gösterilmiştir. CBOW mimarisi, verilen "*I lost keys yesterday*" bağlamından "*my*" kelimesini tahmin eder. Burada V, derlemdeki toplam kelime sayısını; D ise kelime gömülmelerinin boyutunu ifade etmektedir. ∑, gizli katman ağırlıkları ile çarpılan girdi kelime gömülmelerinin ortalamasını gösterirken; Softmax fonksiyonu, sözcük dağarcığındaki tüm kelimeler için olasılıksal dağılımı tahmin etmektedir.



Şekil 3.2. CBOW mimarisi.

Skip-gram mimarisi ise CBOW mimarisindeki girdilerin çıktı, çıktıların ise girdi olarak ifade edildiği durumdur. Diğer bir deyişle, bir kelime OHE gösterimi ile kodlanarak ağa girdi olarak verilir ve pencere boyutu kadar sayıda kelime vektörü elde edilir. Skip-gram mimarisine Şekil 3.3' te yer verilmiştir. Şekilde pencere boyutu "2" olan Skip-gram mimarisine "*my*" kelimesi girdi olarak verildiğinde, "*I"*, *"lost"*, *"keys"* ve *"yesterday*" kelimeleri çıktı olarak verilir. Burada, her kelime-bağlam çifti ayrı ayrı eğitilmektedir.

Her iki mimari de de pencere boyutu kadar bir derlem ile eğitim yapılır. CBOW, Skip-grama nazaran daha hızlıdır ve kelimelerin tekrarlandığı durumlarda daha iyi sonuçlar verir. Skip-gram ise eğitim setinin az olduğu ve kelimelerin daha seyrek olduğu durumlarda daha başarılı sonuçlar vermektedir (Naseem et al., 2021).



Şekil 3.3. Skip-gram mimarisi.

CBOW ve Skip-gram mimarileri için çıktı vektörlerini öğrenebilmek zor bir problemdir. Bu problemin üstesinden gelebilmek için Negatif Örnekleme (Negative Sampling) ve Hiyerarşik Softmax (Hierarchical Softmax) algoritmaları önerilmiştir (Mikolov et al., 2013a; 2013b). Negatif Örnekleme, gürültü dağılımına dayanarak, güncellenmesi gereken çıktı vektörlerinin sayısını sınırlandırırken; Huffman ağacına dayalı olan Hiyerarşik Softmax, kökten hedefe giden her adımı normalize eder. Bu iki algoritmayı karşılaştıracak olursak, Negatif Örnekleme, daha düşük boyutta gömülmeler kullanıldığı durumlarda ve yüksek frekanslı kelimelerin olduğu durumlarda daha iyi sonuçlar verirken; Hiyerarşik Softmax, kelimelerin seyrek olduğu durumlarda daha iyi sonuçlar vermektedir (Naili et al., 2017).

- **GloVe (Global Vectors for Word Representation):** Pennington ve arkadaşları tarafından önerilen "GloVe" modeli, kelime oluşum (word occurence) istatistiklerini kullanarak problemi çözmeyi amaçlayan denetimsiz bir öğrenme yöntemidir (Pennington vd., 2014). GloVe, kelimeler arasındaki uzaklığın, anlamsal benzerlikle ilişkili olduğu bir alana kelimeleri eşleyerek kelime gömülmelerini elde etmeye çalışır (Ollagnier and Williams, 2019). CBOW ve Skip-gram kelime gömülmesi, yerel bağlam pencere bilgisinden yararlanırken; GloVe, bir derlemdeki global kelime ve kelime oluşum istatistiklerini kullanır ve bunlar üzerinde eğitilir.

GloVe, iki temel adıma dayanmaktadır. İlk adım, X kelime-kelime birlikte görünme matrisinin (co-occurrence matrix) oluşturulmasıdır. İkinci adım ise vektörlerin elde edilebilmesi için, X kelime-kelime birlikte görünme

matrisinin çarpanlarına ayrılmasıdır. GloVe, Eşitlik 3.4'te verilen amaç fonksiyonunu optimize etmeye çalışır.

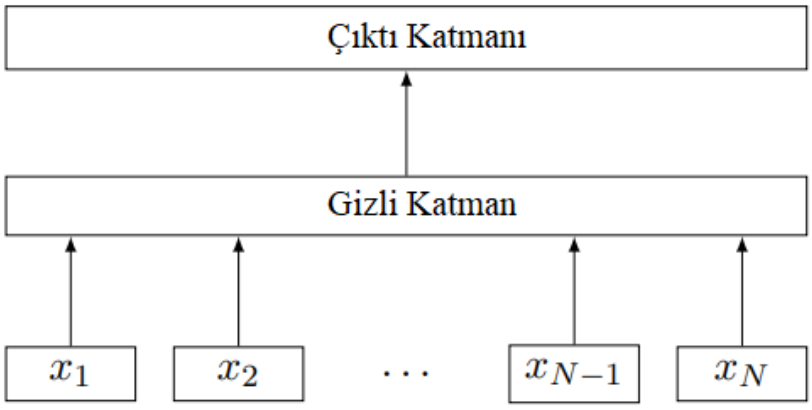
$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \quad (3.4)$$

Eşitlikte, X_{ij}, i kelimesi bağlamında j kelimesinin bulunma sayısını göstermektedir. V, derlemin büyüklüğü (sözcük dağarcığı sayısı) olmak üzere; $X_i = \sum_k^V X_{ik}$, derlem içerisindeki tüm kelimelerin i kelimesi bağlamında toplam görünme sayısını ifade etmektedir. $f(X_{ij})$, sık görülen birlikteliklere kısmen daha düşük ağırlıklar atayan bir ağırlıklandırma fonksiyonudur (Goldberg, 2017). w_i, d boyutlu uzayda i kelime gömülmesini; $\tilde{w}_j$, d boyutlu uzayda j kelimesinin bağlam kelime gömülmesini ve b, önyargı (bias) değerini göstermektedir. Kelimelerin birlikte görünme oranları ile optimize edilmeye çalışılan amaç fonksiyonu sayesinde, CBOW ve Skip-gram'da kullanılan pencere ile kelimelerin belirlenmesi işlemi de ortadan kaldırılmıştır.

GloVe, bir kelimenin görünme sayısından bağımsız olarak anlamsal temsilleri bulmaya çalışır. Bu durum, büyük bir derlemde az tekrarlayan kelimelerin gömülmelerinin bulunması için bir avantaj sağlamaktadır. GloVe yönteminde, istatistiki bilgilerin verimli bir şekilde kullanılması, elimizde küçük bir derlem olduğunda ve vektör boyutunun küçük olduğu durumlarda modelin doğru gömülmeler elde etmesine imkan sağlamaktadır.

GloVe yönteminin yukarıda bahsedilen avantajlarının yanında bazı dezavantajları da bulunmaktadır. GloVe yönteminde, kelime gömülmelerinin doğru bir şekilde oluşturulabilmesi için büyük derlemler üzerinde öğrenme işleminin yapılması gerekmektedir. Bu zaman açısından bir maliyet doğurmaktadır. Kelime-kelime birlikte görünme matrisini büyük bir derlemde oluşturmak, yer açısından da maliyetli olmaktadır. Ek olarak, sözcük dağarcığında ve eğitim derleminde olmayan kelimelerin (out-of-vocabulary - OOV) kelime gömülmelerini öğrenemezler (Naseem et al., 2021). Bu gibi dezavantajların üstesinden gelebilmek için yeni çözümler önerilmiştir.

- **FastText:** Word2Vec modelinin bir uzantısı olan FastText, 2016 yılında Facebook Yapay Zeka Araştırma Grubu (Facebook AI Research - FAIR) tarafından geliştirilmiş bir kelime temsil yöntemidir (Joulin vd., 2016a; 2016b; Bojanowski vd., 2017). Bu yöntemde, kelimeler tek tek yapay sinir ağına girdi olarak verilmek yerine, karakter bazlı olarak "n-gram" halinde parçalanıp yapay sinir ağına girdi olarak verilir. N-gram ifadesindeki n değeri, kelimenin ne kadarlık parçalara bölüneceğini ifade etmektedir. Bir kelimenin vektörü, tüm n-gram vektörlerinin toplamı alınarak elde edilir. Örneğin, "*Daisy*" kelimesi için 2-gram (bigram) değeri, $\{Da, ai, is, sy\}$ şeklinde gösterilmektedir ve "*Daisy*" kelimesinin vektörü elde edilirken "<D", "Da", "ai", "is", "sy" ve "y>" bigram vektörleri toplanır. Burada, kelimenin başlangıcını ve bitişini belirtmek için, kelimenin başlangıcına "<" karakteri, bitişine ise ">" karakteri eklenir. FastText mimarisine Şekil 3.4'te yer verilmiştir. Şekilde, modelin girdisi olan $x_1, x_2, \dots, x_{n-1}, x_n$ değerleri, w kelimesinin n-gramlarını göstermektedir.



Şekil 3.4. FastText model mimarisi (Joulin, 2016a).

FastWord2Vec ve GloVe modelleri, kelimelerin birlikte göründüğü kelimeleri baz alarak bir kelime gömülmesi elde ederken; Fasttext, kullandığı n-gram yapısı sayesinde kelimelerin biçimbilimsel (morphologic) özelliklerini hesaba katmaktadır. Bu sayede, Word2Vec ve GloVe modellerinin en önemli dezavantajlarından biri olan sözcük dağarcığında ve eğitim derleminde olmayan kelimelerin kelime gömülmelerini öğrenememe probleminin önüne geçmiş olur. Başka bir deyişle Fasttext, bazı kelimeler

eğitim veri setinde yer almamasına rağmen, kullandığı n-gramlar sayesinde farklı kelimeler arasında ortak olan alt kelimelerden (subword) yola çıkarak, kelimeler arası ilişkilendirme yapabilmektedir.

Derlemde düşük frekansa sahip olan kelimelerin n-gramlarının ortaya çıkma olasılığı, daha yüksek frekanslı kelimelere nazaran daha düşük olacağı için, n-gramlar sayesinde düşük frekanslı kelimeler, daha etkin bir şekilde temsil edilebilmektedir. Bu yöntemde, n-gram sayısı, kelime sayısından kat kat fazla olacağı için eğitim süresi uzamaktadır. Fastext, n-gramlar sayesinde, morfolojik olarak zengin diller için daha uygun bir gösterim şekli sunmaktadır.

Bahsi geçen kelime gömülme modelleri, her ne kadar bir belgedeki sözdizimsel ve anlamsal bilgileri kullansa da belge bağlamına özgü kelime gömülmelerinin nasıl oluşturulacağı ile ilgili sorunu çözememişlerdir. Bu da NLP'de birçok alt görev için problem teşkil etmektedir. Bu sorunun üstesinden gelebilmek için bağlama dayalı kelime gömülmeleri önerilmiştir.

3.1.2.2 Bağlam tabanlı gösterim (Context-based representation)

Doğal dilde, yazılışı aynı olan ancak anlamı kullanıldığı bağlam ile değişen bazı belirsiz (ambiguous) kelimeler bulunmaktadır (İng. bank, left, pen, tear, lead, conduct vb). Örneğin, aşağıda verilen iki cümleyi ele alalım:

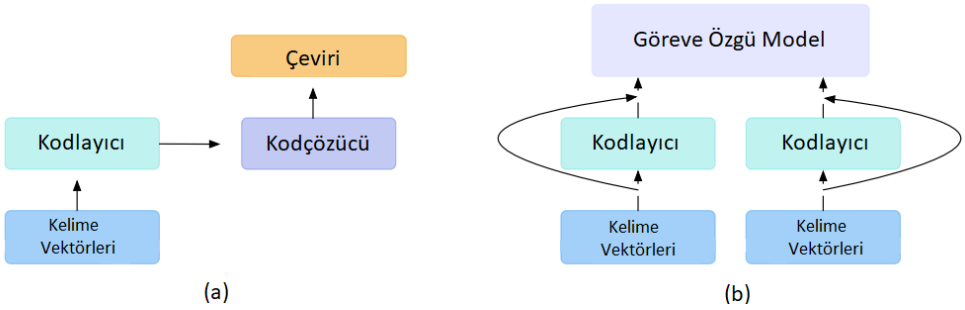
- "*The amount of the lead determines the quality of gasoline."*
- "*She was totally born to lead.*"

Burada, iki cümlede de yazılışı aynı olan "*lead*" kelimesinin anlamı, kullanıldığı bağlama göre değişmektedir. İlk cümlede geçen "*lead*" kelimesi, "*kurşun*" anlamında kullanılmaktayken, ikinci cümlede geçen "*lead*" kelimesi, "*liderlik/öncülük etmek*" anlamında kullanılmıştır. Belirsiz kelimelerin belirsizliğinin giderilememesi, bir çok NLP alt görevinde engel oluşturduğu gibi, kelime temsilleri için de engel teşkil etmektedir. Bu belirsizliğin, kelime temsili için

ortadan kaldırılmasını amaçlayan bağlam tabanlı yaklaşımlar önerilmiştir (Peters et al., 2017; McCann et al., 2017).

Bağlam tabanlı gösterim, bağlamdaki her bir kelime için ayrı ayrı bir vektör öğrenmek yerine; kelimelerin sırasını ve birlikte görünen kelime kümesini de dikkate alarak cümle için bir gömülme elde eder.

- **CoVe (Context Vectors) :** CoVe (McCann et al., 2017), bağlamsal kelime vektörlerini öğrenmek için CBoW / Skipgram Word2Vec ve GloVe kullanmak yerine Makine Çevirisi (Machine Translation - MT) mimarisinden faydalanmaktadır. İlk kelime vektörü olarak GloVe'u kullanan CoVe, MT görevi için eğitilen diziden diziye (sequence-to-sequence) dönüştürme yapan model ile bağlama dayalı şekilde bir çıktı oluşturur. Oluşturulan bu çıktı, GloVe kelime vektörleri ile art arda eklenir ve nihai kelime gömülmeleri elde edilir. Ardından, bu gömülmeler başka dillere çevirilir (Şekil 3.5). CoVe ile bağlam tabanlı gösterimler elde edebilmek için, iki katmanlı iki yönlü Uzun Kısa-Süreli Bellek (Bidirectional Long Short-Term Memory - BiLSTM) kullanılır.



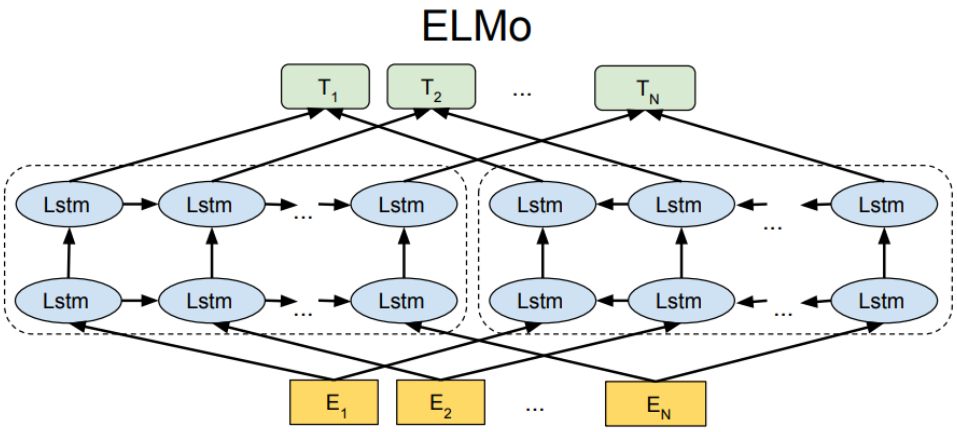
Şekil 3.5. (a) İki katmanlı BiLSTM'in kodlayıcı olarak eğitilmesi (b) NLP alt görevleri için kullanılması (McCann et al., 2017).

Şekil 3.5'te gösterildiği gibi MT veri kümeleri üzerinde eğitilen LSTM kodlayıcılar, diğer NLP alt görevleri - duygu analizi (sentiment analysis), soru sınıflandırma, metinsel gerektirim (textual entailment) ve soru yanıtlama için eğitilen modellerin performansını arttırmak için kullanılmıştır.

- **ELMo (Embeddings from Language Models):** Peters ve arkadaşları tarafından 2018 yılında ortaya atılan derin öğrenme tabanlı bir modeldir

(Peters et al., 2018). Çift yönlü Dil Modeli (Bidirecitonal Language Model - BiLM)'ne dayanan ELMo, bağlamsal gösterimleri çıkararak, geleneksel kelime gömülmelerini genelleştirir. Her bir kelime için sabit bir kelime gömülmesi kullanmak yerine, içinde geçtiği cümlenin tamamına bakan ELMo modelinin temelinde, belirli bir görev üzerinde eğitilmiş olan soldan-sağa ve sağdan-sola Uzun Kısa-Süreli Bellek (Long Short-Term Memory - LSTM)'lerin birleşimi bulunmaktadır. Soldan-sağa ve sağdan-sola LSTM kullanılmasındaki temel amaç, dil modellerinin her iki yönde de maksimum olasılığını almaktır (Wang et al., 2020). İki yönlü LSTM, belirli bir kelimenin sadece kendinden önceki kelimelere değil, sonrasındaki tüm kelimelere de bağlı olduğunu göstermeye yardımcı olmaktadır. ELMo modelinin çalışma prensibine Şekil 3.6'da yer verilmiştir.

Öznitelik tabanlı bir yaklaşım olan ELMo modeli, farklı cümlelerde yer alan aynı kelimeler için farklı kelime gömülmeleri elde etmektedir. Bunun için, son katmandaki kelime gömülmelerini kullanmak yerine, BiLM'nden öğrenilen kelime gömülmelerini doğrusal bir şekilde arda arda birleştirir.



Şekil 3.6. ELMo modeli çalışma prensibi (Devlin et al., 2019).

Mevcut olan modellere kolaylıkla adapte edilebilen ELMo, Soru Yanıtlama (Rajpurkar et al., 2016), Duygu Analizi (Socher et al., 2013), Adlandırılmış Varlık Tanıma (Named Entity Recognition -NER) (Tjong Kim Sang and De Meulder, 2003) gibi bir çok NLP alt görevinde oldukça başarılı sonuçlar elde etmiştir.

ELMo modelinin oluşturulmasıyla birlikte, her biri bir öncekinden daha iyi bir performans gösteren derin öğrenme modelleri ortaya çıkmıştır. Tez çalışması kapsamında kullanılan diğer derin öğrenme tabanlı öznitelik temsili modellerine Bölüm 3.2'de detaylı bir şekilde değinilmiştir.

Kısaca özetlemek gerekirse; öznitelik temsili, bir çok NLP alt görevinde performansı önemli ölçüde etkilediği için, kelimelerin mümkün olduğunca gerçek özniteliklerini bulabilmek, performansı arttırmaktadır. Bu doğrultuda önerilen frekans tabanlı kelime gösterimleri, uygulanması kolay olmasına karşın; kelimenin konum ve sıra bilgisini dikkate almadığı için kelimeler arasındaki ilişkileri gözden kaçırmaktadır. Ek olarak, derlem büyüklüğü arttıkça oluşturulan girdi vektörünün de büyüklüğü artacağı için bellekte ciddi sıkıntılara neden olabilmektedir. Öğrenme tabanlı kelime gömülmelerinden olan Word2vec, GloVe ve FastText modelleri, frekans tabanlı kelime gösterimi yöntemlerinin eksikliklerinin üstesinden gelmeyi başarmış ancak, bağlamı dikkate almadan farklı anlamlara gelen aynı kelimeler için aynı kelime vektörünü kullanmışlardır. Ek olarak, Word2Vec ve GloVe, eğitim derleminde olmayan kelimelerin kelime gömülmelerini öğrenemediği için bu kelimelere rastgele vektör atamaktadırlar. Bahsi geçen tüm öznitelik temsil modellerinin avantaj ve dezavantajlarına Tablo 3.1'de değinilmiştir. Tabloda yer alan bu dezavantajlar birçok NLP alt görevinde performans kaybına neden olduğu için derin öğrenmeye dayalı yeni kelime gösterim modelleri önerilmiştir.

3.2 Derin Öğrenme Modelleri

Günümüzde NLP'de makine öğreniminden bahsettiğimizde aklımıza ilk gelen yapı, 2017 yılında Vaswani ve arkadaşları tarafından önerilen bir Transformer yapısıdır. Transformer, evrişimleri ve yinelemeleri tamamen ortadan kaldıran "Dikkat (Attention)" mekanizmasına dayalı basit bir ağ mimarisidir (Vasvani et al., 2017). Transformer, kodlayıcı ve kodçözücü olmak üzere iki temel bileşenden oluşmaktadır. Yinelenen ağları kullanmadan kodlayıcı ve kodçözücü yardımıyla bir diziyi başka bir diziye dönüştürür.

Tablo 3.1. Kelime gösterim modellerinin karşılaştırılması.

Öznitelik Modeli	**Türü**	**Avantajları**	**Dezavantajları**
OHE ve BoW	Frekans tabanlı	▪ Kolaylıkla uygulanabilir. ▪ Daha önce karşılaşmadığı kelimeler ile çalışabilir. ▪ Kelimeleri çıkarmak için temel bir ölçüdür.	▪ Anlamsal ve sözdizimsel bilgileri yakalayamazlar. ▪ Ortak kelimeler sonuçları etkilemektedir. ▪ Kelimelerin duygularını yakalayamazlar.
TF ve TF-IDF	Frekans tabanlı	▪ Kolaylıkla uygulanabilir. ▪ Tanımlayıcı kelimeleri çıkarmak için temel bir ölçüdür. ▪ IDF'den dolayı ortak olan kelimeler sonuçları etkilemez.	▪ Anlamsal ve sözdizimsel bilgileri yakalayamazlar. ▪ Kelimelerin duygularını yakalayamazlar.
CBOW Word2Vec	Tahminleme tabanlı	▪ Büyük veri seti üzerinde önceden eğitilmiştir. ▪ Eğitimde, Skipgram'a göre daha hızlıdır ve sık görünen kelimelerle iyi çalışırlar. ▪ Anlamsal ve sözdizimsel bilgileri yakalayabilir.	▪ Küçük eğitim veri seti ve nadir görünen kelimelerle çalışamazlar. ▪ Bağlamsal bilgileri yakalayamazlar. ▪ OOV kelimelerinin gömülmelerini elde edemezler. ▪ Öğrenme işlemi için çok büyük bir derleme ihtiyaç duyarlar.
Skipgram Word2Vec	Taminleme tabanlı	▪ Büyük veri seti üzerinde önceden eğitilmiştir. ▪ Küçük eğitim veri seti ve nadir görünen kelimelerle iyi çalışırlar. ▪ Anlamsal ve sözdizimsel bilgileri yakalayabilir.	▪ Eğitim işlemi yavaştır ve sık kullanılan kelimeler için doğruluk değeri daha düşüktür. ▪ Bağlamsal bilgileri yakalayamazlar. ▪ OOV kelimelerinin gömülmelerini elde edemezler. ▪ Öğrenme işlemi için çok büyük bir derleme ihtiyaç duyarlar.

GloVe	Frekans tabanlı (Öğrenmeye dayalı)	▪ Alt-doğrusal (sub-linear) ilişkileri belirleyebilmek için vektör uzayını kullanır. ▪ Daha küçük ağırlıklar kullanmak, gereksiz kelimeler (stopwords) gibi sık tekrarlayan kelimeler için eğitim sürecini etkilemez. ▪ Elde edilen vektörler, diğer yöntemlerden elde edilen vektörlere göre daha yorumlanabilirdir.	▪ Öğrenme işlemi için çok büyük bir derleme ihtiyaç duyar. ▪ Eğitim esnasında, kelime-kelime birlikte görünme matrisini saklamak için büyük bellek gerektirir. ▪ Bağlamsal bilgileri yakalayamazlar. ▪ OOV kelimelerinin gömülmelerini elde edemez.
FastText	Tahminleme tabanlı	▪ Nadir görünen kelimelerle iyi çalışır. ▪ OOV kelimelerinin gömülmelerini bulabilir.	▪ Bağlamsal bilgileri yakalayamaz. ▪ Büyük bellek gerektirir. ▪ Word2Vec ve GloVe modelleri ile kıyaslandığında hesaplama açısından daha maliyetlidir.
CoVe	Tahminleme tabanlı	▪ Çokanlamlılık (polysemy) için bağlama dayalı gömülmeler elde edebilir. ▪ Elde edilen gömülmeler, NLP alt görevleri için kullanılabilir.	▪ OOV kelimelerinin gömülmelerini elde edemez. ▪ Zaman açısından maliyetlidir. ▪ Başka kelime gömülmeleri gerektirir.
ELMo	Tahminleme tabanlı	▪ Çok anlamlı kelimeler için bağlama dayalı gömülmeler elde edebilir. ▪ Gömülmeler, NLP alt görevleri için kullanılabilir. ▪ OOV kelimelerinin gömülmelerini bulabilir.	▪ Zaman açısından maliyetlidir. ▪ Başka kelime gömülmeleri gerektirir.

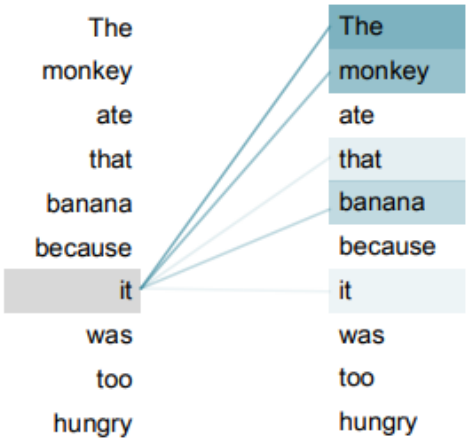
Transformer; RNN, LSTM ve GRU'daki Gradyan yok olması problemini, yinelemelerden dolayı hesaplamanın uzun sürmesi ve işlemlerin kelime kelime sıralı olarak gerçekleştirilmesi sorunlarını ortadan kaldırmıştır. Bu çalışmadan esinlenilerek BERT, RoBERTa, DistilBERT, ALBERT gibi birçok yeni derin öğrenme modeli ortaya atılmıştır.

3.2.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) Google Araştırma Grubu'nda yer alan araştırmacılar tarafından 2018 yılında geliştirilen, önceden eğitilmiş bir doğal dil işleme modelidir (Devlin et al., 2019). Google bu model ile kullanıcıların arama sorgularının daha iyi analiz edilmesini sağlayarak, kullanıcılara daha doğru sonuçlar döndürülmesini amaçlamaktadır. Kısacası temel amaç, modelin dili anlamasını sağlamaktır.

3.2.1.1 BERT model mimarisi

BERT, "Dikkat (Attention)" mekanizmasına dayalı çok katmanlı çift yönlü bir Transformer kodlayıcıdır. Öz-dikkat (self-attention), Transformer'ın işlemekte olduğu kelimeyi anlamak ve ilgili olan diğer kelimeye dönüştürmek için kullandığı yöntemdir. Transformer kodlayıcıdaki öz-dikkat mekanizmasına bir örnek Şekil 3.7'de verilmiştir.



Şekil 3.7. Transformer'daki öz-dikkat mekanizmasına bir örnek (Xie at al., 2021).

BERT, "base" ve "large" olmak üzere iki ayrı model olarak tanıtılmıştır. BERT-base 12 katman (Transformer bloğu), 768 gizli katman, 12 attention-head ve toplam 110 milyon parametre içermektedir. Bu modelin eğitimi için toplam 4 TPU'ya ihtiyaç duyulmuştur. BERT-large ise 24 katman (Transformer bloğu), 1024 gizli katman, 16 attention-head ve 6 toplam 340 milyon parametre içermektedir. Bu modelin eğitilmesi için ise 16 TPU'ya ihtiyaç duyulmuştur.

Model, 800 milyon BooksCorpus kelimesi (Zhu et al., 2015) ve 2.5 milyar Wikipedia kelimesi de dahil olmak üzere toplam 3.3 milyar kelimelik bir derlem üzerinde yaklaşık 40 epokta, Adam optimizasyon algoritması kullanılarak eğitilmiştir.

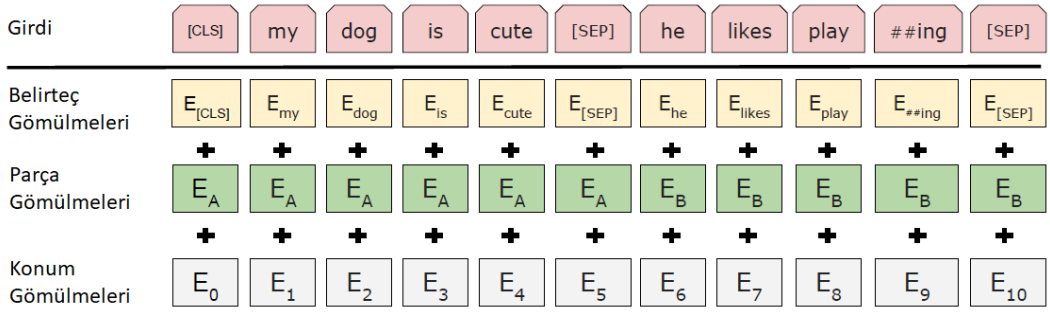
BERT'te girdi gösterimini oluşturmak için bazı özel belirteçler (token) kullanılmaktadır. Bu belirteçler;

- **[CLS]:** Her dizinin ilk belirteci olarak kullanılan sınıflandırma belirtecidir. Yapılacak olan görev ne olursa olsun, BERT bu belirteci beklemektedir.
- **[SEP]:** Bir cümle çifti girdi olarak verildiğinde iki cümleyi birbirinden ayırt etmek için kullanılan belirteçtir.
- **##:** Bir kelime, alt kelimelere ve karakterlere ayrıldığında; ayrılan belirteçlerin daha büyük bir kelimenin parçası olduğunu ve kendisinden önce başka bir kelime geldiğini ifade eden simgedir.

BERT, WordPiece Tokenizer (Wu et al., 2016) kullanarak girdi dizisini belirteçlerine ayırır. Belirteçlerin başlangıcına [CLS] belirteci ve ikinci cümlenin başına ve sonuna [SEP] belirteci ekler (girdi dizisi olarak cümle çifti mevcut ise). Aynı zamanda, girdi dizisinde bulunan kelimelerin alt kelimeleri var ise bu kelimelerin başına "##" simgesi ekleyerek kelimeleri alt kelimelerine ayırır. Wordpiece işleminden sonra 30000x768 boyutunda bir matrisin indekslenmesiyle "belirteç gömülmeleri (token embeddings)" elde edilir. Cümle çiftinin kullanılacağı görevler için hangi parçanın hangi cümleden geleceğini belirtmek gerekmektedir. Eğer parça ilk cümleden geliyorsa, 0; ikinci cümleden geliyorsa 1 olacak şekilde 768 uzunluğunda bir "parça gömülmeleri (segment embedding)" vektörü oluşturulur. Transformer mimarisinde olduğu gibi, cümle içerisindeki belirteçlerin hangi konumda olduğunu ifade etmek için "konum gömülmeleri (position embedding)"kullanılmaktadır. Şekil 3.8'de de görüldüğü gibi BERT'e verilecek olan girdi gömülmeleri, bu üç gömülme vektörünün toplamıdır.

BERT modelinde, ön-eğitim (pre-training) ve ince-ayar (fine-tuning) olmak üzere iki prosedür mevcuttur. Ön-eğitimde, farklı ön-eğitim görevleri için

etiketlenmemiş veriler üzerinde eğitim yapılırken; ince-ayar prosedüründe, BERT modeli önceden eğitilmiş parametrelerle başlatılır ve etiketlenmiş veriler kullanılarak ince-ayar yapılır.



Şekil 3.8. BERT girdi gösterimi (Devlin et al., 2019).

3.2.1.2 Ön-eğitim prosedürü

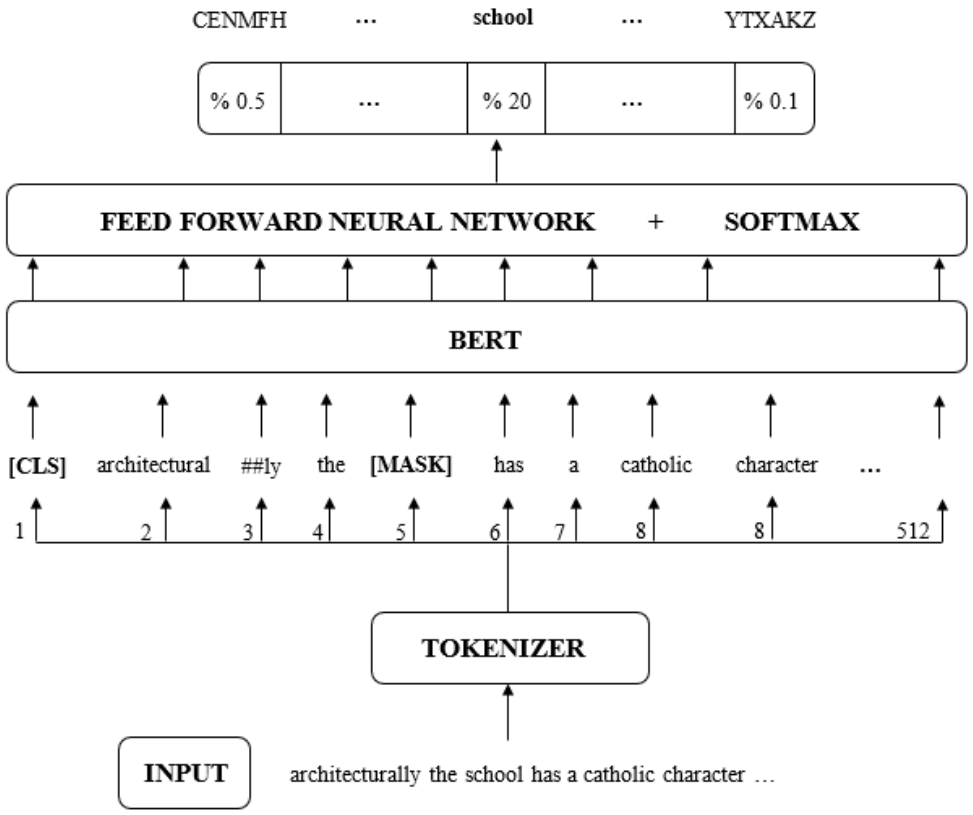
BERT modelini önceden eğitmek için, klasik sağdan-sola veya soldan-sağa dil modeli kullanmak yerine, denetimsiz "Maskelenmiş Dil Modeli (Masked Language Model - MLM)" ve "Sonraki Cümleyi Tahminleme (Next Sentence Prediction - NSP)" görevi kullanılmıştır.

- **Maskelenmiş Dil Modeli:** BERT modelinde, Taylor (1953)'ın önerdiği Cümle Tamamlama Görevi (Cloze Task)'nden esinlenilerek ortaya çıkarılan MLM kullanılmıştır. MLM, girdi metnindeki belirteçlerin %15'ini rastgele seçerek [MASK] etiketi ile maskeler ve maskelenen orijinal belirteci yalnızca metin bağlamına dayalı olarak tahmin etmeye çalışır. Bu işlem gerçekleştirilirken tüm girdiyi tahmin etmek yerine sadece maskelenmiş belirteç tahmin edilir. MLM'ne Şekil 3.9'da örnek verilmiştir. İnce-ayar sırasında, [MASK] etiketi ile maskelenen belirteç görünmediği için ön-eğitim ve ince-ayar arasında bir uyumsuzluk meydana gelmektedir. Bu uyumsuzluğu azaltmak için, tahmin edilmek üzere rastgele seçilen %15'lik belirteçlerden rastgele bir i belirteci seçilir. Bu i belirteci:

    - %80 oranında [MASK] belirteci ile yer değiştirir.

- %10 oranında rastgele bir belirteç ile yer değiştir.
- %10 oranında değiştirilmeden aynen bırakılır.

Örneğin; 400 uzunluğunda bir belirteç dizimiz mevcut ise, 60 belirtece (400 belirtecin %15'i) maskeleme işlemi yapılmalıdır. Bu 60 belirtecin 48 tanesi (60 belirtecin %80'i) hali hazırda maskelenmiş durumdadır. 60 belirtecin 6 tanesini (60 belirtecin %10'u) rastgele belirteçlerle yer değiştirmeli ve yine 6 belirteç değişmeden orijinal belirteç olarak kalmalıdır. Transformer kodlayıcı, hangi kelimelerin tahmin edileceğini veya hangilerinin rastgele kelimelerle değiştirildiğini bilmediği için, her girdi belirtecinin dağılımsal bağlamsal gösterimini elinde bulundurmak zorunda kalır.



Şekil 3.9. BERT Maskelenmiş Dil Modeli örneği.

- **Sonraki Cümleyi Tahminleme:** İki cümle arasındaki ilişkilerin anlaşılması ihtiyacı bulunan Doğal Dil Çıkarımı (Natural Language Inference - NLI) gibi görevler için NSP işlemine de gereksinim

duyulmaktadır. Her bir eğitim girdi dizisini oluşturmak için derlemden iki cümle çifti seçilir. İlk cümle, A gömülmesi ile ifade edilirken, ikinci cümle B gömülmesi ile ifade edilir. Seçim işlemi yapılırken, verilerin %50'sinde birinci cümleden sonra orijinalde olması gereken ikinci cümle gelirken, %50'sinde ise derlemden rastgele seçilen bir cümle ikinci cümle olarak gelmektedir. Bu görev ile ilgili örneğe aşağıda yer verilmiştir. Girdi için birleştirilmiş maksimum uzunluk 512 belirteçten daha küçük olmalıdır. [CLS] belirteci sonraki cümlenin tahminlenmesi için kullanılır.

Girdi: [CLS] architectural ##ly the [MASK] has a catholic character [SEP] atop the main building ' s gold dome is a [MASK] statue of the virgin mary [SEP]
Etiket: Sonraki

Girdi: [CLS] architectural ##ly the school has a [MASK] character [SEP] at the end of the [MASK] chopin returned to paris [SEP]
Etiket: Sonraki olmayan cümle

3.2.1.3 İnce-ayar prosedürü

Cümle çifti içeren uygulamalarda, iki-yönlü çapraz dikkat (bidirectional cross attention) mekanizması uygulanmadan önce cümle çiftleri bağımsız bir şekilde kodlanırken; BERT, bu iki işlemi birleştirerek metin çiftini öz-dikkat mekanizması ile kodlar.

Ön-eğitimde, farklı ön-eğitim görevleri için etiketlenmemiş veriler üzerinde eğitim yapılmaktadır. Burada ise, BERT modeli önceden eğitilmiş olan parametrelerle başlatılır ve etiketlenmiş veriler kullanılarak göreve özgü ince-ayar yapılır. Örneğin, soru yanıtlama gibi belirteç düzeyindeki görevler için çıktı katmanı belirteç gösterimleri iken, duygu analizi gibi sınıflandırma görevleri için çıktı katmanı [CLS] gösterimleridir.

İnce-ayar prosedüründe, yığın boyutu (batch-size) 16/32, öğrenme oranı 5e-5/3e-5/2e-5 ve epok sayısı 2/3/4 olarak alınmış; geri kalan tüm parametreler ön-eğitim prosedüründeki parametreler ile aynı olacak şekilde kullanılmıştır.

Dikkat mekanizmaları ve bunu temel alan BERT modeli, gerek diller arası çeviri gerekse sınıflandırma alanı gibi birçok NLP alanında umut vaat edici sonuçlar elde etmektedir. BERT tanıtılmasının akabinde, takip eden yıl (2019) içerisinde birçok yeni NLP modelinin ilham kaynağı olmuştur. Bu geliştirilmiş yeni yöntemlerde, BERT'in üzerinde yapılan çeşitli değişikliklerle bu modelin başarısının ve zaman/maliyet açısından verimliliğinin daha da artırılabileceği gösterilmiştir. Takip eden bölümlerde, geliştirilen bu yeni modellerin özellikle BERT ile farklılıkları ele alınarak incelenmiştir.

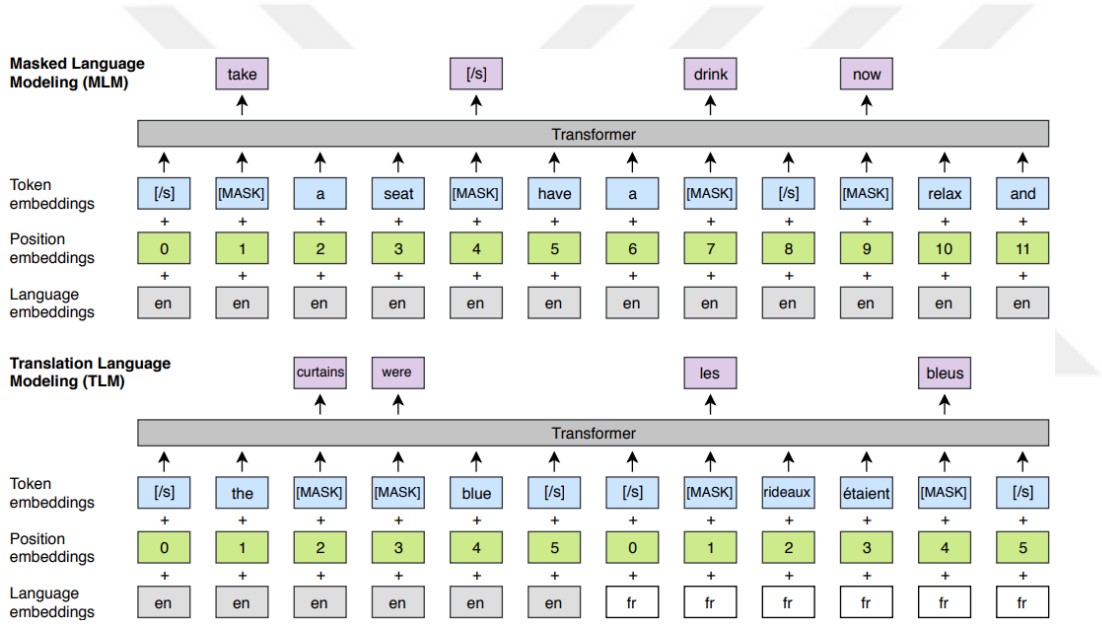
3.2.2 XLM

BERT mimarisini geliştirerek diller-arası (cross-lingual) çeviri konusunda başarı sağlamak için 2019 yılında Conneau and Lample (2019) tarafından öne sürülmüştür. XLM (Cross-lingual Language Model), bilinen önişleme tekniklerinden birini (2-gram kodlama/Byte Pair Encoding-BPE) ve yine bilinen bir çift-dil eğitim mekanizmasını BERT ile birleştirerek, farklı dillerde aynı anlama gelen kelimeler arasındaki ilişkinin öğrenilmesini sağlamaktadır. Bu çalışmanın literatüre olan temel katkısı, çoklu-dil sınıflandırma görevi için BERT ile çalışan yeni bir eğitim tekniği sunmasıdır. Şekil 3.10'da görüldüğü üzere önerilen çalışma, hem orijinal BERT 'deki standart eğitim modeli (MLM)'nin hem de yeni önerilen Çeviri Dil Modeli (Translation Language Modeling-TLM)'nin dönüşümlü olarak birlikte uygulanması ile elde edilmiştir. Ayrıca modelin, çoklu-dil sınıflandırma görevi (15 dilde cümle kurma) ve diller-arası çeviri görevi açısından, bilinen diğer modellere karşı daha başarılı olduğu önerilen çalışmada gözler önüne serilmiştir.

3.2.3 XLNet

Yang et al. (2019), temel olarak BERT'in MLM eğitim metodu üzerinde geliştirme yaparak, genelleştirilmiş otomatik-azalan ön-eğitim (generalized autoregressive pretraining) yöntemini önermiştir. Buna ek olarak, BERT'teki

standart Transformer yapısında girdi dizisinin maksimum uzunluğu önceden belirlenen bir boyut ile kısıtlanmaktadır. Bu durum maksimum boyuttan daha büyük bir girdi dizisinin gelmesi durumunda bilgi kaybı yaşanmasına neden olmaktadır. XLNet (Transformer-XL) ise, otomatik-azalan (Autoregressive-AR) eğitim modelinde BERT'teki standart Transformer'ları kullanmak yerine, Transformer-XL yapısı kullanarak eğitim esnasında ince-ayarın daha uzun cümleler ile yapılabilmesine olanak sağlamaktadır. Böylece, geliştirilen model sayesinde BERT'te mevcut olan eğitimsel eksiklikler ile mücadele edilmiş; soru yanıtlama, doğal dil çıkarsama, duygu analizi ve doküman sıralama gibi 20 farklı NLP görevinde BERT'ten daha iyi sonuçlar elde edildiği deneysel sonuçlar ile gösterilmiştir.



Şekil 3.10. XLM ön-eğitim modeli (Conneau and Lample, 2019).

3.2.4 RoBERTa

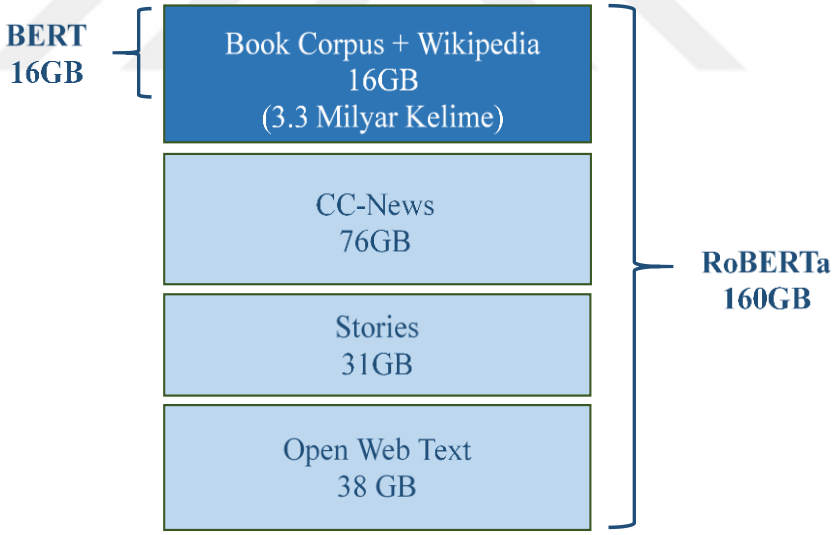
Liu et al. (2019), BERT'in yeteri kadar eğitilmediğini düşünerek, BERT'in ön-eğitim safhası üzerinde iyileştirmelerde bulunmuşlardır. Mimari olarak standart Transformer'ları kullanan RoBERTa (Robustly-optimized BERT approach), eğitim safhası için aşağıdaki dört değişikliği uygulamıştır:

- Eğitim veri setini Şekil 3.11'de özetlendiği üzere artırmıştır. RoBERTa modeli, BERT'in eğitim setinden 10 kat daha geniş bir İngilizce-dil derlemi ile eğitilmiştir.

- BERT'in aksine, maskeleme işlemini, veri ön-işleme esnasında bir kez hesaplamak yerine; dizinin modele her geçişinde maskenin hesaplandığı Dinamik Maskeleme Deseni (Dynamic Masking Pattern) yöntemini önermiştir.

- NSP kayıp fonksiyonunu eğitim işleminden kaldırarak, dört farklı NLP görevi üzerinde test etmiş ve daha başarılı sonuçlar verdiğini göstermiştir.

- Eğitim işlemini BERT'e göre daha geniş bir yığın boyutu ve daha az adımla gerçekleştirmiş ve böyle bir eğitim şeklinin daha başarılı sonuç verdiğini göstermiştir.



Şekil 3.11. RoBERTa eğitim veri seti (Godbole et al., 2020).

3.2.5 DistilBERT

BERT bilindiği üzere, zaman ve hesaplama açısından maliyetli bir modeldir. BERT'ten türetilen modeller ise genellikle daha ağır ve geniş modeller önererek

başarıyı artırma yoluna gitmektedir. Sanh et al. (2019), BERT'i sadeleştirme yoluna giderek de başarının artırılabileceğini yaptıkları çalışmada göstermişlerdir. DistiBERT (a Distilled version of BERT) ile, BERT modelinin boyutu %40 azaltılmasına karşılık, modelin dili anlama kapasitesi BERT kadar yüksek bir başarı gösterebilmiştir. Böylece, daha hızlı bir model ile neredeyse BERT ile aynı performans elde edilmiştir. Mimarinin boyutunun sadeleştirilmesi, bazı katmanların, belirteç-tipi gömülmelerin, NSP görevinin ve biriktirme (pooling) işlemelerinin kaldırılması ile gerçekleştirilmiştir. Ayrıca, RoBERT gibi daha geniş yığın boyutu ve daha az adımla eğitim stratejisini uygulamışlardır.

Eğitim kaybı için; arıtma kaybı (distillation loss), MLM kaybı ve kosinüs gömülme kaybı (cosine embedding loss) fonksiyonu kullanılmıştır. Bu kayıp fonksiyonları, modelin düzgün bir şekilde öğrenilmesini ve verimli bir bilgi aktarımı gerçekleşmesini sağlamaktadır.

3.2.6 ALBERT

ALBERT (A Little BERT), RoBERTa'nın amaçladığı gibi daha başarılı ve DistilBERT'in amaçladığı gibi daha hızlı olma özelliğini tek bir modelde birleştirmek için Lan et al. (2019) tarafından önerilmiştir. Bu işlemi gerçekleştirmek için, 110 milyon parametre içeren BERT'in parametre açısından verimsiz olduğu öne sürülmüş ve parametre sayısı düşürülerek yeni bir model önerilmiştir. ALBERT, 12 milyon parametre, 768 gizli katman ve 128 gömülme katmanından oluşmaktadır. Bu model temel olarak aşağıdaki iki yöntem ile gerçeklenmiştir:

- Çarpanlarına ayrılmış gömülme parametrelendirmesi (Factorized Embedding Parameterization) ile gömülme katmanı 728 katmandan 128 katmana düşürülmüş ve parametrelerin daha verimli ve az sayıda olmaları sağlanmıştır.
- İlk kodlayıcının parametresi öğrenilerek, katmanlar arası parametre paylaşımı (Cross-Layer Parameter Sharing) yapılmıştır.

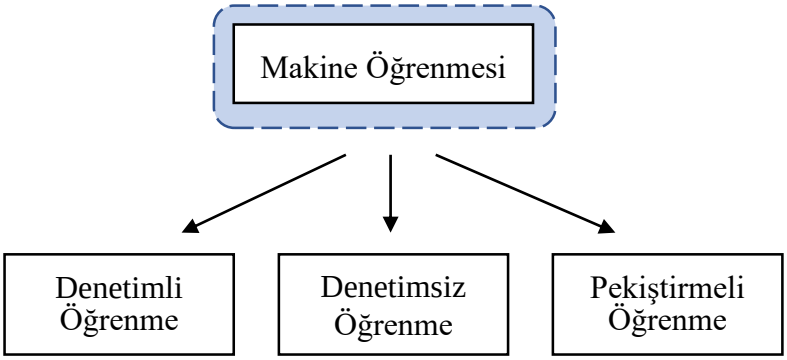
BERT'in kullandığı veri seti üzerinde, BERT ile aynı mimari (katman/öznitelik boyutu/öz-dikkat) kullanılmasına rağmen, ALBERT modeli ile daha düşük hafıza tüketimi ve daha hızlı eğitim başarısı elde edilmiştir. Ayrıca, parametre paylaşımı yapılarak parametre sayısının azaltılmasının doğurduğu başarı azalmasının, kazanılan verimliliğe göre kabul edilebilir seviyede olduğu ilgili çalışmadaki sonuçlar ile gösterilmiştir.

3.3 Makine Öğrenmesi Yöntemleri

Makine Öğrenmesi, Yapay Zekâ'nın ele aldığı temel konuların başında gelen alt çalışma alanıdır. Makine öğrenmesinin bilinen ilk tanımı Arthur Samuel (1967) tarafından 'bilgisayarları açıkça (direk) programlama yaparak işlevselleştirmek yerine, bilgisayarlara öğrenme yeteneği kazandıran çalışma alanı' olarak yapılmıştır. Makine öğrenmesinin daha modern ve günümüzde kabul gören tanımını ise Tom Mitchell (1997) tarafından şu şekilde yapılmıştır: 'bir bilgisayar programının, ölçülebilen bir görev üzerindeki performansı kazanmış olduğu deneyim ile gelişiyor ise, bu programın deneyim kazanarak öğrendiği söylenebilir.' Bu tanımlamalardan da anlaşılacağı üzere temel amaç, bilgisayarların insan müdahalesi olmadan kendi kendine öğrenmesini ve gelecekte daha iyi kararlar alabilmesini sağlamaktır.

Öğrenme türleri Şekil 3.12'de gösterildiği üzere üç ana başlık altında toplanmaktadır: Denetimli (supervised), denetimsiz (unsupervised) ve pekiştirmeli (reinforcement) öğrenme. Doğru cevapların öğrenme işlemi esnasında hazır olarak verildiği ML türüne denetimli öğrenme denmektedir. Diğer bir deyişle, bu tür makine öğrenmesinde, etiketlenmiş bir veri seti ML algoritmamıza girdi olarak verilmekte, böylece algoritma eğitim safhasında hangi örnekleri doğru sınıfta hangi örneklerin ise yanlış sınıfta olduğunu bilmektedir. Bu sayede öğrenme algoritması hipotezini, veri setindeki örnekler ve etiketleri arasındaki eşleştirmeler üzerine kurarak, daha önce görmediği benzer örnekleri sınıflandırma görevlerinde kullanılabilecek olan ML modelini oluşturmaktadır. Denetimli öğrenmeyi bir örnek vererek açıklayacak olursak: Elimizde erkek ya da kadın yazarlar tarafından yazıldığını bildiğimiz belli sayıda makale olduğunu farz edelim. Cinsiyet etiketleri her bir makale örneği için bilinen bu makale verisi üzerinde denetimli ML

algoritmalarından birini uygulayacak olursak, ilgili algoritmanın sağlanan makale ve etiketleri arasındaki eşleşmeden yararlanarak oluşturacağı ML modeli sayesinde daha önce hiç görülmemiş bir makalenin yazarı için erkek/kadın tahmininde bulunulabilecektir. Denetimsiz öğrenme türünü kısaca açıklayacak olursak, bu tür ML, doğru veya yanlış cevapların bilinmediği örnekler içeren veri setleri üzerinde uygulanmaktadır. Farklı bir deyiş ile bu tür makine öğrenmesinde, etiketlenmemiş bir veri seti ML algoritmamıza girdi olarak verilmekte, yani algoritmamız hangi örneklerin doğru sınıfta hangi örneklerin ise yanlış sınıfta olduğunu bilmemektedir. Dolayısı ile uygulayacağımız bir denetimsiz ML yöntemi, verilen görev doğrultusunda karar verebilmek için söz konusu olan etiketsiz veri setinin yapısını kendi kendine öğrenecektir. Bu öğrenme işlemini veri setindeki örnekler arasında benzerliğe bakarak ve benzer özelikteki örnekleri bir araya gelecek şekilde kümeleyerek gerçekleştirmektedir. Her ne kadar etiketsiz veriler üzerinde öğrenerek tahminleme yapmak daha zor görünse de, gerçek hayattaki verilerin genelde etiketsiz, gürültülü ve karmaşık olduğunu düşünürsek denetimsiz öğrenme yönteminin kısıtlı veriler üzerinde daha uygulanabilir olduğunu söyleyebiliriz. Denetimsiz öğrenmeyi de bir örnek ile açıklayacak olursak: internetteki çeşitli kaynaklardan elde edilen makalelerin ilgili oldukları konulara göre gruplandırılması görevi denetimsiz ML modeli ile geçekleştirilebilecektir. Buradaki gruplandırma işlemi temel olarak, örnekler arasındaki yakınlık/uzaklık analizi yapmamızı sağlayan en uygun benzerlik ölçüm kriterinin kullanılmasına dayanmaktadır. Sonuncu öğrenme türü olan pekiştirmeli öğrenme ise, tıpkı denetimsiz öğrenmede olduğu gibi doğru veya yanlış cevapların bilinmediği örnekler içeren veri setleri üzerinde uygulanmaktadır. Ancak denetimsiz öğrenmeden farklı olarak makine, doğru kararlar vermeyi ödül-ceza sistemi ile öğrenmektedir. Bu tip öğrenme esnasında ML algoritması her tahmininden sonra geri bildirimde bulunmak yerine, yalnızca verilen görevi başardığında geri bildirimde bulunarak öğrenme işlemini gerçekleştirmektedir. Dolayısı ile pekiştirmeli ML yöntemi genellikle robotik alanında ve akıllı bot uygulamalarında yaygın olarak kullanılmaktadır.



Şekil 3.12. Makine öğrenmesinin türleri.

Bu tez çalışması kapsamında, etiket bilgileri var olan veri setleri üzerinde etkin bir soru sınıflandırma yöntemi önerildiğinden, denetimli öğrenme türünün altında yer alan çeşitli sınıflandırma yöntemleri uygulanmıştır. Takip eden bölümde uygulanan bu sınıflandırma yöntemlerinden kısaca bahsedilmektedir.

3.3.1 Sınıflandırma algoritmaları

Bu tez kapsamında etkin bir soru sınıflandırma yöntemi geliştirmek, yani gözlemlenen her bir soruyu daha önceden sayısı belirlenmiş olan kategorilerden birine atamak yer almaktadır. Ayrıca sınıflandırma algoritmalarının NLP alanında başarıyla uygulandığı bilinmektedir. Bu sebeple, bu bölümde genellikle denetimli makine öğrenmesinin altında yer alan ve bu çalışmada uygulanan Lojistik Regresyon (Logistic Regression - LR), Destek Vektör Makineleri (Support Vector Machines - SVM), Karar Ağaçları (Decision Tree - DT) ve Naif Bayes (Naive Bayes) sınıflandırma algoritmalarının temel özellikleri incelenmektedir. Bu algoritmalara ek olarak K-En Yakın Komşu (K-Nearest Neighbour - KNN) sınıflandırma algoritmasına da değinilmiştir. Denetimli öğrenme türünün altında yer alan diğer bir kavram olan regresyon algoritmaları sınıflandırmadan ziyade hipotezde yer alan girdi değişkenleri arasındaki ilişkiden yola çıkarak, sürekli değerler alan bir başka değişkeni tahmin etmek için kullanıldığından bu tez çalışması kapsamında uygulanmamış ve dolayısı ile bu bölümde yer verilmemiştir.

3.3.1.1 Lojistik regresyon

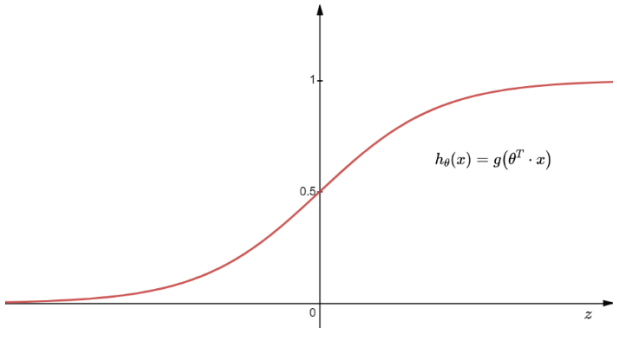
Lojistik Regresyon sınıflandırma yöntemi isim olarak (regresyon) her ne kadar kafa karıştırsa da, genellikle sınıflandırmada problemlerinin çözümünde sıklıkla kullanılan bir öğrenme algoritmasıdır. LR algoritmasının temelinde, gelen bir gözlem için 0-1 arasında bir tahmin üreterek, o gözlemin ait olduğu kategoriye atanması yer almaktadır. Diğer bir deyişle öğrenme hipotezimizin 0 ve 1 arasında değer alacak şekilde modellenmesidir. Hipotezimizin 0 ile 1 arasında değer alması için de lojistik (sigmoid) fonksiyon $g(z)$'den faydalanılmaktadır. Lojistik ya da sigmoid olarak adlandırılan bu fonksiyon matematiksel denklem olarak ifadesi Eşitlik 3.5'te verilmiştir.

$$g(z) = \frac{1}{1+ e^{-z}} \tag{3.5}$$

Daha sonra sigmoid fonksiyonumuzun hipotez fonksiyonu $h_\theta(x)$ olarak ifade edecek olur isek Eşitlik 3.6'daki denklemi elde etmiş oluruz.

$$h_\theta(x) = g(z) = g(\theta^T x) \tag{3.6}$$

Sigmoid fonksiyonun bir öğrenme modeli olarak nasıl çalıştığı, $h_\theta(x)$ fonksiyonun $z = \theta^T x$ değişkenine göre çizdirilmesi ile Şekil 3.13'te gözler önüne serilmiştir.



Şekil 3.13. Lojistik (Sigmoid) fonksiyon.

Eğer sigmoid fonksiyonumuz üzerinde gerekli yerine koyma işlemini, z yerine $\theta^T x$'i yerleştirerek gerçekleştirecek olursak, Eşitlik 3.5'teki denklemimiz Eşitlik 3.7'deki denkleme dönüşecektir.

$$g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \tag{3.7}$$

Lojistik Regresyon yöntemi için belirlenen hipotez $h_\theta(x)$, uygun bir maliyet fonksiyonu ve gradyan inişi (gradient descent) gibi bir en iyileme yöntemi ile eğitilerek, öğrenme işlemini gerçekleştirir. Böylece modelimizi en iyi şekilde ifade edecek olan θ parametreleri bulunmuş olunur. Lojistik Regresyon modelimize verilen gözlemin sınıf 1'e (hipotezimizin hedef sınıfı) ait olma olasılığını $P(y = 1|x; \theta) = h_\theta(x)$ ve diğer sınıfa ait olma olasılığı $P(y = 1|x; \theta) = 1 - h_\theta(x)$ olarak ifade edildiğinde; $h_\theta(x) \geq 0.5$ koşulunu sağlayan gözlemler 1'inci sınıfa atanmış olur. $h_\theta(x) \geq 0.5$ koşulunun sağlanmasında $z \geq 0$ koşulunun sağlanması anlamına geldiğinden (Bkz. Şekil 3.13), Lojistik Regresyon modelinin sınıflandırma için matematiksel koşul denklemi Eşitlik 3.8'deki gibi ifade edilmektedir.

$$g(z) = \begin{cases} 1 \; if \; z \geq 0 \\ 0 \; if \; z < 0 \end{cases} \tag{3.8}$$

Lojistik Regresyon sınıflandırıcısının yapısı gereğince ikili (binary) sınıflandırma problemlerinde kullanılmasının yanı sıra , 1'e-karşı-hepsi mantığı ile uygulanarak çok sınıflı (multiclass) sınıflandırma problemlerinde de yüksek başarıma sahip sınıflandırma modelleri oluşturması beklenmektedir.

3.3.1.2 Naif Bayes algoritması

Naif Bayes algoritması, makine öğrenmesine istatiksel bir yaklaşımda bulunan ve Bayes teoremine dayalı olan bir sınıflandırıcıdır. Karar verme işlemi için verilen bir problemi olasılıksal olarak ele alan bu yöntemi uygulayabilmek için Bayes teoreminde yer alan bütün olasılıksal dağılımları bilmemiz gerekmektedir. Bundan dolayı NB algoritması, gerekli hesaplamaların daha basit şekilde gerçekleştirilebilmesi adına, sınıf koşul bağımsızlığını kabul eder. Sınıf koşul

bağımsızlığı ise bir özniteliğin varlığının başka bir özniteliğin varlığını ya da değerini etkilemeyeceğini ifade eder. Bayes teoremi Eşitlik 3.9'da verildiği üzere hesaplanmakta, bu olasılıksal hesaplama sayesinde ortalama sınıflandırma hatası en düşük seviyede tutacak olan karar verme kuralı türetilmiş olur.

$$P(C|X) = \frac{P(X|C)\,P(C)}{P(X)} \tag{3.9}$$

Burada X, n adet öznitelikten oluşan bir öznitelik vektörünü temsil ederken, C ise özniteliğimizin ait olup olmadığına karar vermeye çalıştığımız sınıfı temsil etmektedir. Hesaplamaya çalıştığımız $P(C|X)$ ise verilen bir X örneğinin C sınıfına ait olup olmama olasılığını temsil etmektedir.

Örneğin, elimizde farklı renklerde meyvelerle dolu olduğu bilinen bir torbadan rastgele bir X meyve seçildiğinde, kırmızı olduğu bilinen bu X meyvesinin elma (C_i) sınıfına ait olup olmadığına karar verilmesi NB algoritması ile gerçekleştirilebilmektedir. NB sınıflandırma algoritmasının çalışması bu örnek üzerinden aşağıdaki adımlar ile açıklanabilir:

- X'in renk özelliği belli olduğu var sayılmak üzere, X, bu renk özelliklerini belirleyen n tane öznitelikten oluşan bir vektör X= $(x_1, x_2, x_3, \ldots, x_n)$ ile temsil edilmektedir.
- Çeşitli meyve türlerinin C_1, C_2, C_3, . . . , C_m, olarak ifade edildiği m adet sınıfımızın olduğunu varsayalım. Rastgele seçilen bir kırmızı meyvenin hangi sınıfına ait olduğu bütün sınıflar (C_i) için ardıl olasılıkların Eşitlik 3.10'a göre hesaplanması ile bulunur:

$$P(C_i|X) = \frac{P(X|C_i)\,P(C_i)}{P(X)} \tag{3.10}$$

- Bu hesaplamada, paydada yer alan $P(X)$ değeri bütün sınıflar için aynı değeri alacağından göz ardı edilir ve sadece eşitliğin payında yer alan $P(X|C_i)\,P(C_i)$ çarpımı dikkate alınır. Burada $P(C_i)$ torbadaki elmaların görünme olasılığı, yani C_i sınıfındaki meyve sayısının bütün

meyvelere olan oranıdır. $P(X|C_i)$ ise elma sınıfı içinde kırmızı elmaların olasılıksal dağılımına tekabül etmektedir.

- Sonuç olarak $P(X|C_i)\,P(C_i)$ ifadesi için en büyük değeri veren C_i sınıfı, kırmızı olduğunu bildiğimiz X gözlemimizin sınıfı olarak belirlenir. Örneğin, $P(C_{armut}|X) > P(C_{elma}|X)$ sonucuna erişmemiz demek, sadece kırmızı olduğunu bildiğimiz X meyvesini armut olarak sınıflandırmamız demektir.

NB sınıflandırıcısı, anlaşılır yapısı ve hesaplama kolaylığına rağmen yüksek doğrulukta sınıflandırma başarısından dolayı başta NLP olmak üzere makine öğrenmesini içeren birçok çalışma alında sıklıkla tercih edilmektedir. Ayrıca lojistik regresyonun aksine çok sınıflı sınıflandırma problemlerinde kullanıma uygundur. NB sınıflandırıcısın diğer sınıflandırıcı yöntemleri ile baş edebilir sonuçlar verdiği birçok çalışmada (Han and Kamber, 2006; Shmueli et al., 2010) gözler önüne serilmiştir.

3.3.1.3 Karar ağaçları

En eski ML algoritmalarından biri olan Karar Ağaçları, uygulanması kolay olmasına karşın bir o kadar kuvvetli bir sınıflandırma yöntemdir. Karar verme işlemi için sağlamış olduğu ağaç yapıları sayesinde çok sınıflı sınıflandırma problemlerinde etkin bir şekilde kullanılmaktadır. Karar ağacı yapısı, girdi almayan bir ana kök düğüm ve her biri girdi alan alt/iç düğümlerden meydana gelmektedir. Çıktısı başka bir düğüme girdi olarak girmeyen iç düğümler ise yaprak düğümler olarak adlandırılmaktadır. Bu yönlü ağaç yapısı sayesinde, doğrusal bir şekilde ayrıştırılamayan bir örnek uzayı, girdi öznitelik değerlerinin bir fonksiyona tabi tutulması ile iki veya daha fazla parçaya ayrılabilmektedir (Rokach and Maimon, 2010).

Karar Ağaçları kullanarak tahminde bulunma ve sınıflandırma işlemi kabaca iki adımda gerçekleştirilmektedir:

1) İlk olarak, karar ağacı modeli, gerekli kural düğümleri ve eşik değerleri belirlenerek eğitim veri seti üzerinde oluşturulmaktadır.

2) Daha sonra, elde edilen modelin başarısı sınama veri seti üzerinde uygun ölçütler kullanılarak değerlendirilir ve elde edilen öğrenme modeli ileride gözlemlenecek değerlerin sınıflandırılmasında kullanılır.

Gehrke (2003) tarafından önerilen çalışmada DT algoritmasının kolay anlaşılır ve hızlı biçimde oluşturulabilir bir sınıflandırma yöntemi olduğu vurgulanmıştır. Buna karşın, özelikle öznitelik sayısının çok olduğu uzayların sınıflandırılmasında kural düğümleri olarak hangi özniteliklerin seçileceğinin dikkatli bir şekilde belirlenmesi gerektiği, aksi durumda sınıflandırıcı modelin aşırı uyumluluk (overfitting) problemi ile karşı karşıya kalacağı Zhao and Zhang (2008) tarafından vurgulanmıştır. Bu problem ile baş edebilmek adına kural düğümlerinin seçimine yönelik birçok başarılı yöntem çeşitli çalışmalarda önerilmiştir (Han and Kamber, 2006; Niuniu and Yuxun, 2010).

3.3.1.4 K-en yakın komşu algoritması

KNN algoritması, denetimli öğrenme yöntemlerinde olduğu gibi eğitim veri seti üzerinde matematiksel bir model oluşturmamaktadır. KNN yeni gelen bir örnek ile elimizdeki veri setini eğitim/test diye ayırmaksızın tamamı ile kıyaslayarak, yakınlık ilişkisine bakmakta ve bu yeni örneği en yakın olduğu örneğin sınıfına atamaktadır. Temelinde yakınlık ölçümü yer alan KNN algoritması sınıflandırmanın yanı sıra kümeleme ve regresyon problemlerinde de sıklıkla kullanılmaktadır.

Veri setimizdeki x_i, x_j örneklerinin, *n* elemanlı veri setimizdeki herhangi iki nokta olduğu ve *N* boyutlu sayısal nitelikler ile ifade edildiğini kabul edersek veri setimiz *X*, aşağıda verilen eşitlikteki gibi ifade edilmektedir:

$$X = \{x_1, x_2, \dots, x_n\} \quad x_n \in \mathbb{R}^N \tag{3.11}$$

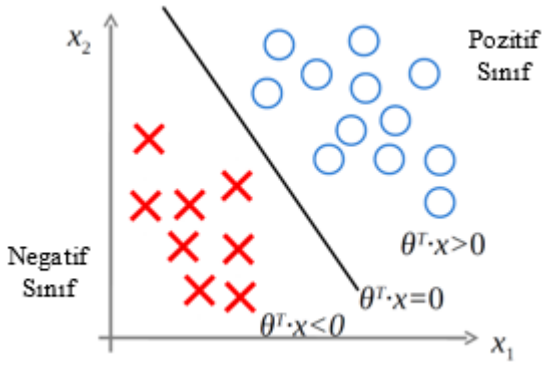
KNN algoritması kullanarak i'inci örneği sınıflandırırken, i'inci örneğin diğer bütün örneklere olan uzaklığı genellikle Öklid uzaklığı (Bkz. Bölüm 3.5) ya da başka bir uzaklık ölçütü ile hesaplanmaktadır. Ardından en küçük uzaklık $dist(x_i, x_j)$ belirlenerek i'inci örneğe en yakın olan j' inci örnek döndürülmüş

olunur. Sonuç olarak, i'inci örnek döndürülen j'inci örneğin ait olduğu sınıfa atanır. Yakınlık hesaplamasının sağlık bir şekilde yapılabilmesi için, özniteliklerin aldığı farklı aralıklardaki değerler sebebi ile bir birine karşı baskın olmasının önüne geçilmelidir. Bunun için de yakınlık hesaplaması yapılmadan önce, tüm özniteliklerin 0-1 aralığında değer alacak şekilde normalize edilmesi (Han and Kamber, 2006) unutulmamalıdır.

KNN algoritmasının arkasındaki temel felsefe, aynı sınıfa ait olan örneklerin benzer özellikler taşımasıdır. Az parametre gerektirmesi ve basit yapısı sayesinde metin sınıflandırmada sıklıkla tercih edilmektedir. Sahip olduğu en büyük problem ise, büyük veri setleri üzerinde başarılı sonuç vermesine rağmen, uygulanması için gerekli olan yüksek maliyetidir.

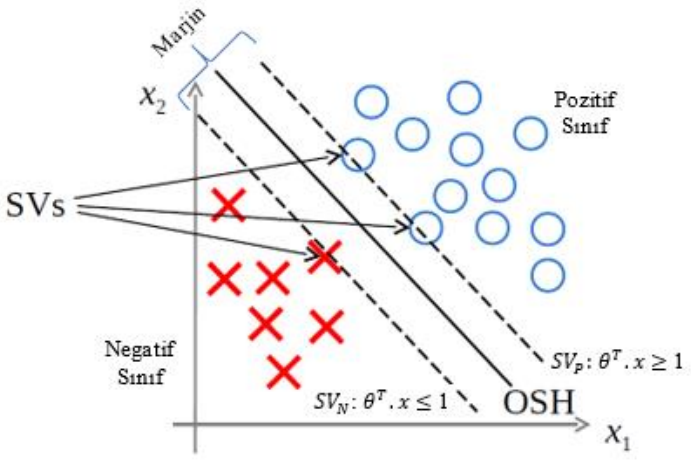
3.3.1.5 Destek vektör makineleri

Destek vektör makineleri metin sınıflandırma problemlerinde sıklıkla kullanılan bir denetimli öğrenme algoritmasıdır. SVM önerildiği (Cortes et al., 1995; Vapnik, 2006) günden itibaren sahip olduğu popülaritesini, hem doğrusal hem de doğrusal olmayan veriler üzerinde başarılı bir şeklide uygulanabilmesine borçludur. SVM, tıpkı lojistik regresyon ve diğer sınıflandırma algoritmalarında olduğu gibi sınıflar arasında karar sınırı (decision boundry) çizmektedir. Bu karar sınırının çizimi de, eğitim veri seti içindeki her iki sınıfa ait 'en ayırt edici' örnekleri belirleyip, bu örnekler üzerinden belirlenecek ve iki sınıf arasındaki ayrışımı maksimum düzeye çıkaracak olan destek vektörlerin belirlenmesi ile gerçekleştirilir. Karar sınırını belirleyen fonksiyonumuzu hipotez fonksiyonu $h_\theta(x) = \theta^T.x$ olarak ifade edecek olur isek, sıradan bir sınıflandırıcı modelinin iki öznitelikli (x_1, x_2) eğitim seti üzerinde yapacağı sınıflandırma Şekil 3.14'te gösterilmiştir.



Şekil 3.14. Sıradan bir sınıflandırıcının çizeceği karar sınırı: $h_\theta(x) = \theta^T.x$.

Şekil 3.14'te ayrıştırılmaya çalışılan iki farklı sınıfa ait olan örnekler (Negatif sınıf: kırmızı çarpılar, Pozitif sınıf: mavi daireler) görülmektedir. Bu örnekler, belirlenen karar sınırı hipotezinde yerine koyulduğunda 0'dan büyük ($\theta^T.x > 0$) yada küçük ($\theta^T.x < 0$) olmasına göre Pozitif sınıf veya Negatif sınıfa atanmaktadır. SVM sıradan bir sınıflandırıcıdan ufak bir fark ile, bu örnekleri karar sınırı ile keskin bir şekilde ayırmak yerine karar sınırı ile destek vektörleri arasına (Pozitif destek vektörü SV_P: $\theta^T.x \geq 1$, negatif destek vektörü SV_N: $\theta^T.x \leq -1$) belirli bir mesafe koyacak şekilde ayırmaktadır. SVM ile elde edilen karar sınırı farklı olarak, optimum ayırt edici düzlem (Optimal Separating Hyper-Plane (OSH)) olarak isimlendirilmiştir. Şekil 3.14'te ele alınan aynı eğitim seti üzerinde SVM algoritmasının eğitilmesi sonucunda elde edilen OSH ve destek vektörleri (SVs) Şekil 3.15 ile gösterilmiştir.



Şekil 3.15. Destek vektörleri SV_N, SV_P ve optimum ayırt edici düzlem OSH.

Burada verilen SVs' ler eğitim setimizin içinde yer alıp, OSH' a en yakın olan örnekleri temsil etmektedir. OSH ile SVs' ler arasındaki mesafe de marjin genişliği olarak tanımlanmaktadır. Daha öncede vurgulandığı gibi, SVM' in temel amacı bu marjin genişliğini maksimum yapan OSH' yi bulmaktır. Şekil 3.15'te görüldüğü üzere veriler iki değişken (x_1, x_2) ile ifade ediliyorsa, SVM üreteceği doğrusal çizgi olan OSH ile verileri ayrılabiliyorken; eğer veri setimiz üç değişken ile ifade edilebiliyorsa OSH bir düzlem; eğer veri setimiz üçten fazla değişken ile ifade ediliyorsa OSH bir üst düzlem olacak şekilde verileri sınıflarına ayırmaktadır. Bu sayede SVM, çoklu sınıflandırma problemlerinde de başarılı şekilde uygulanmaktadır.

SVM yüksek genelleştirme yeteneği sayesinde doğrusal olarak ayrılamayan yüksek boyutlu veriler üzerinde etkin sınıflandırma sonuçları vermektedir. Üstelik bunu yaparken girdi olarak sadece *C* tolerans parametresine ihtiyaç duymakta, bu parametre sayesinde öğrenme işlemini sınıflandırma hatası ve marjin genişliği arasındaki dengeyi kurarak gerçekleştirmektedir. SVM yönteminin uygulanmasında karşılaşılan en büyük sorun, özellikle büyük veri setleri üzerinde sergilemiş olduğu yüksek eğitim süresidir. Bütün bu bilgiler ışığında SVM, örüntü tanıma (pattern recognition), yüz tespiti (face detection) ve sahte mail filtreleme (spam filtering) gibi alanlarda yaygın olarak kullanılmasının yanı sıra, metin sınıflandırma için de sıklıkla kullanılmaktadır. SVM' in metin sınıflandırma için ideal bir sınıflandırıcı olacağı sebepleri ile birlikte Joachims et al. (1998) tarafından önerilen çalışmada vurgulanmıştır.

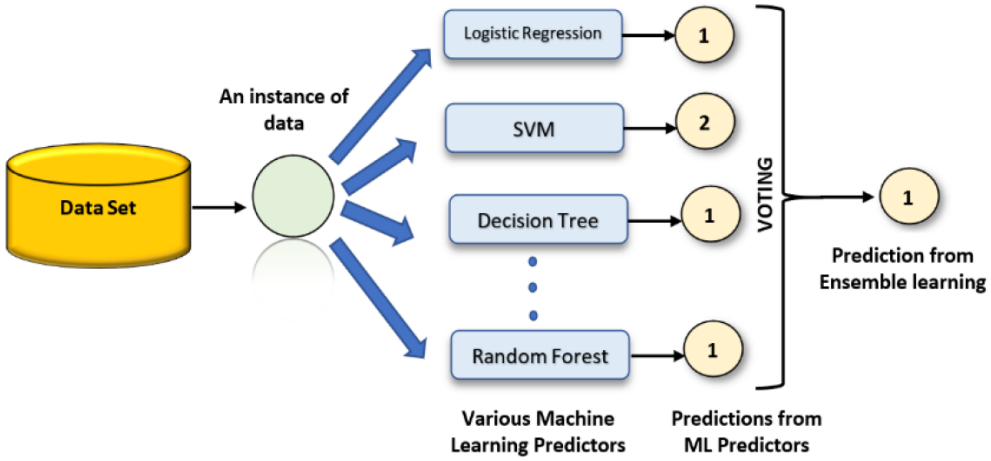
3.4 Topluluk Öğrenmesi Yöntemleri

Topluluk öğrenmesi yöntemi, tek bir sınıflandırıcının karar verme noktasındaki yetersizliklerini gidermek için, birçok öğrenme algoritmasının farklı safhalarda birleştirilmesi ile elde edilen makine öğrenmesi yöntemidir. Bazı öğrenme algoritmalarının başarısının, önceden belirlenmesi gereken parametrelere bağlı olarak değişmesi, bazılarının ise yerel minimuma takılabilmesi gibi birçok eksik yönü bulunmaktadır. Topluluk öğrenmesi ile temel olarak yapılmak istenen, bu eksik yönlerden kaynaklı yanlış karar verme riskini en aza indirmektir. Topluluk öğrenme yönteminde uygulanacak olan birleştirme işleminin hangi seviyede

gerçekleştirileceği seçimi, önerilen yöntemin başarısını doğrudan etkilemektedir. Bu birleştirme işlemi Kuncheva (2014)'nın çalışmasında da belirttiği üzere; sırası ile farklı özniteliklerin, farklı verilerin, farklı sınıflandırıcıların ve farklı birleştirme kurallarının kullanılması ile öznitelik seviyesinde, veri seviyesinde, sınıflandırıcı seviyesinde ve sınıflandırıcı birleştirme seviyesinde gerçekleştirilebilmektedir. Takip eden alt bölümlerde bu sınıflandırıcı topluluğu birleştirme kuralları ve temel topluluk sınıflandırıcıları incelenmektedir.

3.4.1 Oylama (Voting) yöntemi

Topluluk öğrenmesinin en önemli adımlarından birisi, bir araya getirilmek istenen sınıflandırıcıların nasıl birleştirileceğine karar verilmesidir. Gerçek bir sınıflandırıcı metodu olmayan Oylama yöntemi, tam bu noktada karşımıza çıkarak farklı bir grup sınıflandırıcı için bir topluluk çözümü sunmaktadır. Şekil 3.16'da gösterildiği üzere, aynı eğitim seti üzerinde ayrı ayrı eğitilmiş olan, LR, SVM, DT vb. sınıflandırıcı metotlarının tahminlerinin sınıflandırıcı birleştirme seviyesinde birleştirilmesi ile elde edilmektedir. Birleştirme işlemi ise, çoğunluk sınıflandırıcı kararının nihai karar olarak atanmasına dayanmaktadır. Böyle bir model, sınıflandırıcıların bireysel zayıflıklarını dengelemek için eşit derecede iyi performans gösteren bir dizi sınıflandırıcı için yararlı olabilir.



Şekil 3.16. Topluluk oylama modeli (Sumathı, 2021).

Bir tahmin için Çoğunluk / Sert (Majority / Hard) ve Yumuşak (Soft) oylama olmak üzere iki farklı stratejiye göre nihai karara varılır (Bonaccorso, 2017). Bu Oylama çeşitlerinin işleyişi aşağıda ayrıntılı bir şekilde açıklanmıştır.

- **Çoğunluk / Sert oylama:** Belirli bir örnek için tahmin edilen sınıf etiketi, her bir sınıflandırıcı tarafından tahmin edilen sınıf etiketlerinin çoğunluğunu temsil eden sınıf etiketidir. Yani oyların büyük bir kısmını alan sınıf, sınıf etiketi olarak seçilir. Tahminlenen sınıf etiketi $\tilde{y}$, sınıflandırıcı sayısı m ve her bir sınıflandırıcının çoğunluk oyu C_j olmak üzere Eşitlik 3.12'de gösterildiği şekilde bulunur.

$$\tilde{y} = mode\{C_1(x), C_2(x), \dots, C_m(x)\} \tag{3.12}$$

Örneğin, verilen bir x örneği için birinci sınıflandırıcı $C_1(x)$ bu örneği 1 sınıfına, ikinci sınıflandırıcı $C_2(x)$ bu örneği 1 sınıfına ve üçüncü sınıflandırıcı $C_3(x)$ bu örneği 2 sınıfına atamış olsun. Bu durumda çoğunluk oylaması Eşitlik 3.13'te gösterildiği gibi, verilen *x* örneğinin "sınıf 1" olarak sınıflandırılması ile sonuçlanır. Oy eşitliğinin olması durumunda, artan sıralama düzenine göre bir sınıf etiketi seçilecektir.

$$\tilde{y} = mode\{1, 1, 2\} \tag{3.13}$$

Çoğunluk oylama stratejisini, bir j sınıfına atanan w_j ağırlığı ile ilişkilendirerek "Ağırlıklı Çoğunluk Oylama (Weighted Majority Voting)" yöntemi olarak da uygulanabilir. Yukarıdaki örneği baz alarak, ağırlıkları 1., 2., ve 3. sınıflandırıcı için sırası ile {0.1, 0.1, 0.8} olarak kabul edersek nihai sınıf etiketi Eşitlik 3.14'deki gibi 'sınıf 2' olarak belirlenir. Burada *i* değerleri her bir sınıf etiketine (1 veya 2) denk gelmektedir.

$$arg \max_i [0.1 * i_1 + 0.1 * i_1 + 0.8 * i_2] = 2 \tag{3.14}$$

- **Yumuşak oylama:** Sınıf etiketleri sınıflandırıcılar tarafından örnekler için öngörülmüş olan p olasılıklarına göre tahmin edilir. Bu yaklaşım sadece sınıflandırıcılar iyi ayarlanmışsa önerilir. Tahmin edilen her sınıf için olasılık vektörleri (tüm sınıflandırıcılar için) toplanır ve ortalaması alınır. Kazanan sınıf, Eşitlik 3.15'te gösterildiği gibi $\tilde{y}$ için en yüksek değeri veren i' ye tekabül eder.

$$\tilde{y} = \arg\max_i \sum_{j=1}^{m} w_j p_{ij} \tag{3.15}$$

Bu eşitlikte w_j, j sınıflandırıcısına atanan ağırlık değerleridir. Yukarıda yumuşak oylama için verilen aynı örnek üzerinden açıklayacak olursak, her sınıflandırıcının tahminleme olasılığı yine {1,1,2} yani olasılıksak olarak p değerleri $C_1(x) = [0.9, 0.1]$, $C_2(x) = [0.8, 0.2]$, $C_3(x) = [0.4, 0.6]$' dür. Sınıf ağırlıklarını da yine $w_1,\ w_2, w_3$ için sırası ile {0.1, 0.1, 0.8} olarak kabul edersek, her bir sınıf için (i_1, i_2) ortalama olasılıklar Eşitlik 3.16' deki gibi hesaplanır.

$$p(i_1|x) = (0.9 * 0.1 + 0.8 * 0.1 + 0.4 * 0.8)/3 = 0.49$$
$$p(i_2|x) = (0.1 * 0.1 + 0.2 * 0.1 + 0.6 * 0.8)/3 = 0.51 \tag{3.16}$$

Olasılıkların ortalamasını Eşitlik 3.15 te yerine koyar isek nihai sınıf etiketimiz $\arg\max_i[p(i_1|x), p(i_2|x)] = 2$ olacaktır.

3.4.2 Bagging algoritması

Bagging (*Bootstrap aggregating*), Breiman (1996) tarafından önerilen bağımsız sınıflandırıcı topluluk yöntemidir. Aynı sınıflandırıcının bir birinden bağımsız olarak, veri setinden rastgele oluşturulacak farklı alt kümeler (örneklem) üzerinde çalıştırılması ile elde edilir. Bagging algoritmasının temelinde yatan özgünlük, oluşturulacak örneklemelerin tamamen birbirinden farklı hatta birbiriyle zıt özelliklere sahip örneklerden oluşmasıdır. Bu sayede aynı sınıflandırıcının, veri setinin farklı kısımlarında eğitilmesi ile farklı tahminlemeler üreten modeller elde edilmiş olunur. Bagging yönteminde kullanılacak olan sınıflandırıcının seçmi , farklı model çeşitliliğinin elde edilebilmesi için dikkat edilmesi gereken önemli bir

husus olarak karşımıza çıkmaktadır. Farklı veri setleri üzerinde farklı sonuçlar veren, yani tutarsız çalışan sınıflandırıcıların seçilmesi elzemdir. Daha sonra bu sınıflandırıcılara ait çıktılar, sanki farklı sınıflandırıcı modellerinden geliyormuş gibi kabul edilir ve genellikle ağırlıklı çoğunluk oylama yöntemi kullanılarak birleştirilir. Bagging algoritmasının genel çalışması, eğitim ve işleyiş safhaları ile birlikte Şekil 3.17'de gözler önüne serilmiştir.

Eğitim: $Z=\{z_1, \ldots, z_N\}$ Etiketli Veri Seti

1. Topluluk boyutu (L) ve temel sınıflandırma modelini belirle.
2. Z veri setinden L adet önyüklemeli örneklem oluştur ve her bir $D=\{D_1, \ldots, D_L\}$ sınıflandırıcısını, oluşturulan örneklemlerden birini kullanarak eğit.

İşleyiş: Her bir x nesnesi için:

1. Nesneyi, $D=\{D_1, \ldots, D_L\}$ sınıflandırıcılarını kullanarak sınıflandır.
2. Herhangi bir D_i sınıflandırıcı tarafından nesneye atanan etiketi, ilgili sınıf olarak kullanılan bir oy olarak kabul ederek, x nesnesini toplamda sınıflandırıcılar tarafından en çok oylanan sınıfa ata.

Şekil 3.17. Bagging algoritması (Kuncheva, 2014; Onan, 2016).

3.4.3 Rastgele orman algoritması

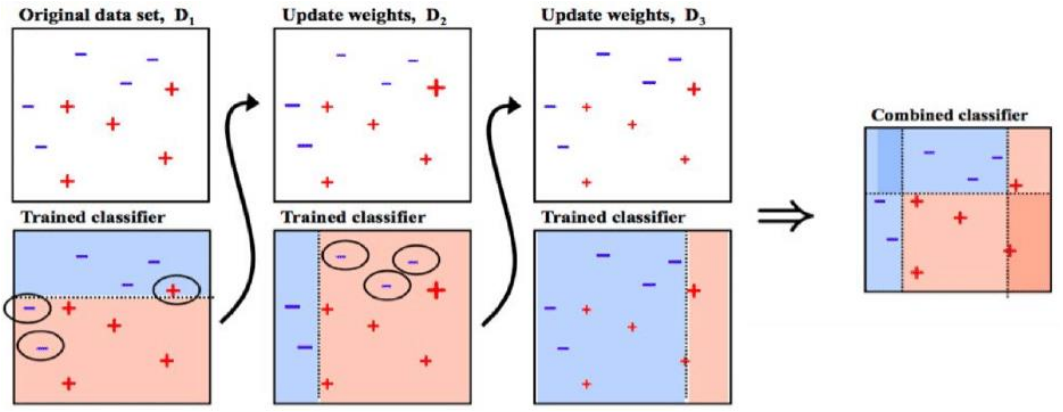
Rastgele orman algoritması, karar ağaçları yöntemindeki aşırı uyumluluk sorunu ile mücadele etmek için geliştirilen bir topluluk öğrenmesi yöntemidir. En basit tanımı ile, Bagging yönteminin karar ağaçları üzerinde uygulanmış halidir. Breiman (2001) tarafından önerilen RF yöntemi, yine kendisinin geliştirmiş olduğu Bagging yöntemi (1996) ve Ho (1998) tarafından geliştirilen rastgele altkümeler yönteminin birleştirilmesi ile oluşturulmuştur. Sınıflandırmanın daha başarılı bir şekilde gerçekleştirilmesi için, karar ağaçlarının oylama mekanizması ile bir araya getirilerek aşağıdaki adımlarla gerçekleştirilmektedir:

- İlk olarak eğitim veri seti rastgele alt kümelere ayrılır, daha sonra oluşturulan her bir alt küme üzerinde karar ağacı yapısı oluşturulur. Oluşturulan bu karar ağaçları topluluğuna karar ormanı denir.
- Karar ormanını oluşturan karar ağaçları bir oylama mekanizması ile bir araya getirilir. RF algoritması gelecekte görmediği veriler üzerinde tahminde bulunurken, daha yüksek ağırlığa sahip karar ağacının sınıflandırmasına güvenerek karar verir.

RF algoritması girdi olarak sadece iki parametre almaktadır. Birinci parametre en iyi kararları verebilmek için her bir düğümde kullanılan özniteliklerin sayısı m, diğeri ise karar ormanını oluşturulacak olan ağaç sayısının N'dir. Basit yapısına rağmen pek çok veri seti üzerinde, SVM gibi diğer etkili sınıflandırıcılar kadar iyi sonuçlar verdiği gözlemlenmektedir.

3.4.4 AdaBoost algoritması

Buraya kadar incelenmiş olan topluluk öğrenmesi yöntemlerinin hepsi paralel bir şekilde, yani birbirinden bağımsız sınıflandırıcıların öğrenmesi ile elde edilen sonuçlarının birleştirilmesi ile oluşturulan yöntemlerdi. Artırımlı (Boosting) öğrenme algoritmaları ise bağımlı sınıflandırıcı öğrenmesi ile adım adım gerçekleşen öğrenme yöntemleridir. Yani her bir adımdaki sınıflandırıcı, bir önceki sınıflandırıcının sergilediği sonuçlara göre eğitilmekte ve son adımda elde edilen sınıflandırıcı nihai model olarak kabul edilmektedir. Adaboost (adaptif artırım) algoritmaları artırımlı öğrenme algoritmalarının en bilineni olarak Freund and Schapire (1996) tarafından geliştirilmiştir. Adaboost algoritması tıpkı Bagging yönteminde olduğu gibi aynı sınıflandırıcının, fakat rastgele değil bilinçli olarak seçilmiş veri seti alt kümeleri üzerinde eğitilmesi ile elde edilir. Şekil 3.18'de gösterildiği üzere, veri setindeki her örnek önce eşit ağırlıklar ile başlayıp, her adımda bir önceki eğitimde doğru sınıflandırılamayan örneklerin ağırlıkları artırılarak eğitime devam edilmektedir. Aynı zamanda başarılı bir şekilde sınıflandırılan örneklerin ağırlıkları da azaltılmaktadır. Böylece Adaboost algoritması ile, her eğitim adımında bir önceki adımdaki eksiklikler üzerinde durularak, önceki sınıflandırıcının hatalarının giderilmesi öğrenilmektedir. Adaboost algoritmasının daha etkin sınıflandırma yapması için, her adımda farklı bir temel sınıflandırıcı yöntemi kullanılarak da uygulanabilmektedir. Bu durumda her adımda örneklerin ağırlıklandırılmasına ek olarak, sergiledikleri performansa göre her bir sınıflandırıcı da ağırlıklandırılmaktadır. Yani daha doğru sınıflandırma yapan baz sınıflandırıcı yöntemlerine daha yüksek ağırlıklar verilmektedir. Bagging yöntemine göre daha yavaş ama etkili olan Adaboost yöntemi, metin sınıflandırma problemlerinde sıklıkla tercih edilen bir öğrenme modelidir.



Şekil 3.18. Adaboost sınıflandırıcısının eğitilmesi (Marsh, 2016).

3.5 Metin Benzerlik Ölçütleri

Soru yanıtlama problemi, soru ve cevap arasında bir benzerlik bulunmasını gerektirmektedir (Achananuparp et al., 2008). Metin benzerlik ölçütleri, soru yanıtlamada dahil olmak üzere, metin özetleme, metin sınıflandırma, belge kümeleme, konu tespiti, makine çevirisi, bilgi alma ve belge eşleştirme gibi birçok NLP görevlerinde önemli bir rol oynamaktadır. İki cümle birbiri ile iki şekilde benzer olabilir: Sözdizimsel benzerlik veya anlamsal benzerlik. İki cümlenin sözdizimsel olarak benzerliği incelenirken, iki cümlede de yer alan kelime sayısı (ortak kelime/kelime grubu sayısı) baz alınmaktadır. İki cümlenin anlamsal olarak benzerliği incelenirken ise kelimelerin anlamları (Pradhan et al., 2015) ile birlikte sözdizimsel bilgi kullanılmaktadır.

Metinler veya cümleler arasındaki benzerliği tespit edebilmek amacıyla çeşitli metin benzerlik ölçütleri geliştirilmiştir. Bu benzerlik ölçütlerinden tez kapsamında kullanılan yaklaşımlar aşağıda kısaca açıklanmıştır.

- **Kosinüs Benzerliği (Cosine Similarity):** Sayısal vektörler ile temsil edilen iki cümlenin Kosinüs benzerliği, iki vektör arasındaki açının kosinüs değerine bakılarak ölçülmektedir. S_1 ve S_2 iki cümle vektörü ve θ bu iki cümle vektörü arasındaki açı olmak üzere, S_1 ve S_2 arasındaki kosinüs benzerliği Eşitlik 3.17'ye göre hesaplanmaktadır:

$$CosSim(S_1, S_2) = cos\theta = \frac{S_1.S_2}{\|S_1\|.\|S_2\|} \quad (3.17)$$

Eşitlikte yer alan $S_1.S_2$, iki cümle vektörünün skaler çarpımını (dot product) ve $\|S_1\|.\|S_2\|$ iki cümle vektörünün büyüklüklerinin (magnitude) çarpımını ifade etmektedir. Kosinüs benzerliği, [0,1] aralığında değişmektedir. Kosinüs benzerliği 1'e yakın ise iki cümle birbirine benzer, 0'a yakın ise iki cümle birbirleri ile ilişkili değildir, anlamına gelmektedir. İki cümle vektörü arasındaki θ açısı ne kadar küçük ise, iki cümlenin birbirine olan benzerliği de o kadar yüksektir.

- **Öklid Uzaklığı (Euclidean Distance):** Sayısal vektörler ile temsil edilen iki cümlenin Öklid uzaklığı, vektörlerin birbirlerine olan geometrik uzaklığıdır. L_2 normu olarak da bilinen Öklid uzaklığı, Eşitlik 3.18'de belirtilen formüle göre bulunmaktadır.

$$EucDis(S_1, S_2) = \sqrt{\sum_{i=1}^{n}(S_{1i} - S_{2i})^2} \quad (3.18)$$

Burada S_1 ve S_2, iki cümle vektörünü; n ise S_1 ve S_2 vektörünün uzunluğunu temsil etmektedir. Öklid uzaklığının 0 olması, iki cümlenin birbiri ile aynı olduğunu göstermektedir.

- **Jaccard Benzerliği (Jaccard Similarity):** Belirteçlerine ayrılan iki metin veya iki cümle arasında hangi belirteçlerin farklı, hangi belirteçlerin aynı olduğunu tespit ederek, [0,1] aralığında bir değer elde eden benzerlik yöntemidir. A ve B belirteçlerine ayrılmış iki cümle olmak üzere, A ve B arasındaki Jaccard benzerliği Eşitlik 3.19 yardımıyla hesaplanmaktadır.

$$JaccSim(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (3.19)$$

Eşitlikteki pay, iki cümledeki ortak belirteç sayısını temsil ederken payda, iki cümlenin birleşim kümesindeki eleman sayısını temsil

etmektedir. Jaccard benzerlik değerinin 1'e yakın olması, iki cümlenin birbirine benzer olduğunu göstermektedir.

- **Jensen-Shannon Uyuşmazlığı (Jensen-Shannon Divergence - JSD):** Bilgi çapı (information radius -IRAD) veya ortalamaya olan toplam uzaklık (Manning and Schütze, 1999) olarak da bilinen JSD, iki olasılık dağılımı arasındaki benzerliği veya mesafeyi ölçmek için kullanılan bir yöntemdir. Kullback-Leibler uyuşmazlığındaki (KLD) simetrik olmama problemini ortadan kaldırmak için önerilen JSD, Eşitlik 3.20 ve Eşitlik 3.21 yardımı ile hesaplanmaktadır.

$$KLD(P||Q) = \sum_{i=1}^{n} P_i \log \frac{P_i}{Q_i} \tag{3.20}$$

$$JSD(P_1||P_2) = \frac{1}{2}\left[KLD(P_1||\frac{P_1+P_2}{2}) + KLD(P_2||\frac{P_1+P_2}{2})\right] \tag{3.21}$$

Eşitlik 3.20, P ve Q olasılık dağılımları arasındaki KLD değerini vermektedir. Buradaki n değeri, P ve Q olasılık dağılımlarının uzunluğunu temsil etmektedir. Eşitlik 3.21'deki P_1 ve P_2, JSD değeri bulunmak istenilen iki cümlenin olasılık dağılımlarını temsil etmektedir. JSD değerinin 0 olması, iki cümlenin birbiri ile aynı olduğunu göstermektedir.

- **Sørensen-Dice Katsayısı (Sørensen-Dice Coefficient):** Dice katsayısı olarak da bilinen yöntem, iki küme arasındaki benzerliği bulmak için kullanılmaktadır. A ve B belirteçlerine ayrılmış iki cümle kümesi olmak üzere, A ve B kümesi arasındaki Dice katsayısı Eşitlik 3.22 yardımıyla hesaplanmaktadır (Dice, 1945).

$$Dice(A,B) = \frac{|A \cap B|}{\propto|A|+(1-\propto)|B|} \tag{3.22}$$

Burada, $\propto \in [0,1]$ aralığında olmak üzere, genellikle $\propto = \frac{1}{2}$ olarak alınmaktadır (Mukhopadhyay, 2018). Eşitlikteki pay, iki cümledeki ortak belirteç sayısını gösterirken, paydada yer alan $|A|$ ve $|B|$

sırasıyla, A ve B cümlelerinin belirteç sayılarını göstermektedir. Dice katsayısının 1'e yakın olması iki cümlenin birbiri ile yakından ilişkili olduğunu göstermektedir.

- **Overlap Katsayısı (Overlap Coefficient):** Szymkiewicz-Simpson katsayısı olarak da adlandırılan Overlap katsayısı, Dice katsayısı gibi iki küme arasındaki benzerliği bulmak için kullanılan bir metin benzerlik ölçütüdür. Dice katsayısı ile aralarındaki temel fark, iki cümlenin birbiri ile aynı kabul edilebilmesi için, bir cümlenin diğer cümlenin alt kümesi olması gerekliliğidir (Gomaa and Fahmy, 2013). A ve B belirteçlerine ayrılmış iki cümle kümesi olmak üzere, A ve B kümesi arasındaki Overlap katsayısı Eşitlik 3.23'teki gibi hesaplanmaktadır.

$$Overlap(A,B) = \frac{|A \cap B|}{min(|A|,|B|)} \tag{3.23}$$

Burada pay, A ve B belirteç kümelerinin kesişimini ifade ederken payda, A ve B belirteç kümelerinden en az elemana sahip olan kümeyi ifade etmektedir. Overlap katsayısının 1 olması, iki kümenin birbiri ile aynı olduğunu göstermektedir.

- **Korelasyon Uzaklığı (Correlation Distance):** Verilen A ve B vektörü arasındaki korelasyon uzaklığı Eşitlik 3.24'e göre hesaplamaktadır.

$$Corr(A,B) = \left| \frac{(A-\bar{A}).(B-\bar{B})}{\|(A-\bar{A})\| \, \|(B-\bar{B})\|} \right| \tag{3.24}$$

Eşitlikte $\bar{A}$ ve $\bar{B}$ sırasıyla, A ve B vektörlerinin ortalamalarını; $\|x\|$, x vektörünün büyüklüğünü; $x.y$, x ve y vektörlerinin skaler çarpımını; $|z|$, z'nin mutlak değerini ifade etmektedir. Birbirine benzer vektörlerin korelasyon uzaklığı 0'a yakın, birbirleri ile ilişkili olmayan vektörlerin arasındaki korelasyon uzaklığı 1'e yakındır.

- **Kulsinski Uzaklığı (Kulczynski Distance):** İki boolean vektör olan A ve B arasındaki farklılığı bulmamızı sağlayan Kulsinski uzaklığı, Eşitlik 3.25 yardımı ile hesaplamaktadır.

$$Kul(A,B) = \left(\frac{NTF+NFT-NTT+n}{NTF+NFT+n}\right) \tag{3.25}$$

Burada n, A veya B vektörünün uzunluğunu; NTF, A vektöründe doğru, B vektöründe yanlış olanların sayısını ve NFT, A vektöründe yanlış, B vektöründe doğru olanların sayısını ifade etmektedir. NTT ise her iki vektörde de doğru olanların sayısını göstermektedir. Bu değerin küçük olması iki vektörün benzerliğinin yüksek olduğunu göstermektedir.

4. GELİŞTİRİLEN YÖNTEMLER

Bu bölümde, soru sınıflandırma için tez kapsamında önerilen öznitelik seçimi, soru sınıflandırma algoritmalarının hibritlenmesi için önerilen mimari ve Soru Yanıtlama Sistemleri için önerilen hibrit soru yanıtlama mimarisi sunulmaktadır.

4.1 Soru Sınıflandırma İçin Önerilen Yöntemler

Tezin bu bölümünde, soru analizinin bir parçası olan soru sınıflandırma aşaması için önerilen öznitelik seçimi ve sınıflandırma algoritmalarının birleştirilmesi için geliştirilen hibrit yönteme değinilmiştir.

4.1.1 Soru sınıflandırma için önerilen öznitelik seçimi

Bu bölümde, sınıflandırma algoritmalarının soru sınıflandırmadaki başarımını arttırmak amacıyla önerilen öznitelik seçimi yöntemine değinilmiştir. Yüksek boyutlu verilerin metin temsilinde kullanılması, ilk bakışta metin hakkında daha fazla bilgi veriyor gibi görünse de bu veriler hem eğitim süresini uzatmakta hem de aşırı uyumluluk riskini arttırmaktadır. Ayrıca, gereksiz öznitelikler, metinin etkili bir şekilde temsil edilmesini engellemektedir. Diğer bir deyişle, etkili bir şekilde metin temsili yapabilmek, dikkatli bir şekilde seçilecek sınırlı sayıdaki öznitelik ile de mümkündür. Bu sorun ile mücadele etmek için sıklıkla kullanılan yöntem öznitelik seçimi (feature selection) yöntemidir.

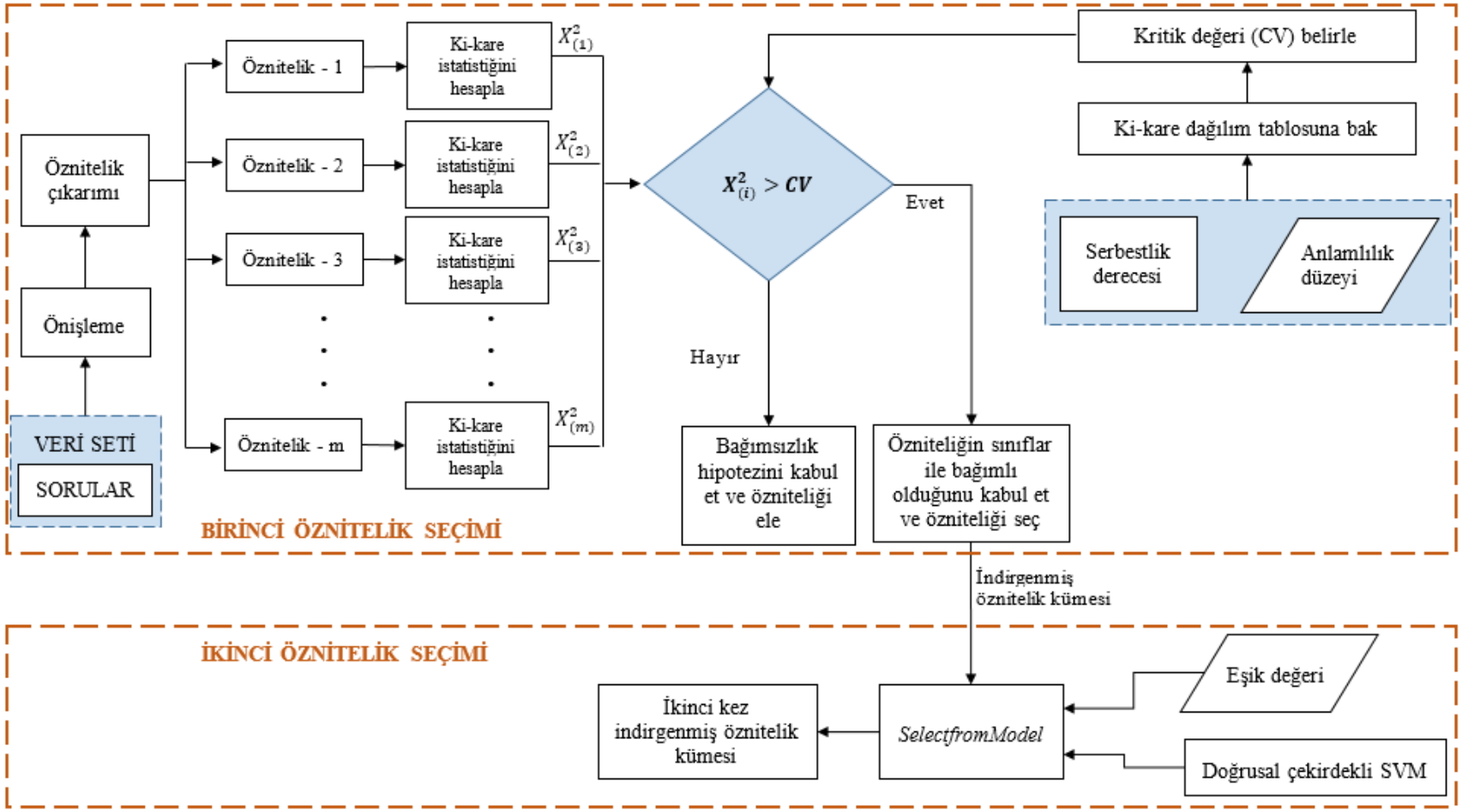
Öznitelik seçimi, her bir sınıftaki en önemli kelimeleri hassas bir şekilde ayrıştırarak, veri kümesini en iyi şekilde temsil etmeyi sağlayan ve veri kümesindeki öznitelik sayısının azaltılmasını amaçlayan bir işlemdir. Ki-kare (Chi-square) testi de seyrek matrisler üzerinde sıkça kullanılan öznitelik seçimi yöntemlerinden biridir. Öznitelik seçimi için kullanılan Ki-kare yöntemi (Liu and Setiono, 1995), temel olarak sınıf etiketleri ile öznitelikler arasındaki istatistiksel ilişkiden yola çıkılarak geliştirilmiştir. Yöntemde ilk olarak, özniteliklerin sınıflara göre Ki-kare istatistikleri Eşitlik 4.1 yardımıyla hesaplanır. Ardından, serbestlik derecesi (degree of freedom) ve önceden belirlenen anlamlılık düzeyine

(significance level) göre Ki-kare dağılım tablosuna bakılarak kritik değer bulunur. Daha sonra, ilk adımda elde edilen her bir özniteliğe ait olan Ki-kare istatistikleri, bulunan kritik değer ile karşılaştırılarak özniteliklerin ayrıştırılması gerçekleştirilir. Yüksek bir Ki-kare değerine sahip olan öznitelik, veri kümesi için daha anlamlı olacağından (Yazıcı vd., 2015), kritik değerin üzerinde kalan ki-kare istatistiğine sahip olan öznitelikler saklanırken, kritik değerin altında kalan öznitelikler elenir.

$$X^2 = \sum \frac{(Gözlenen\ değer - Beklenen\ değer)^2}{Beklenen\ değer} \tag{4.1}$$

SelectfromModel ise her bir özniteliğe farklı yöntemlerle belirlenen eşik değerine göre önemlilik değeri atayan başka bir öznitelik seçim yöntemidir. Özniteliklerin önemlilik değeri, belirlenen eşik değerinin altında ise, öznitelikler önemsiz olarak kabul edilir ve bu öznitelikler elenir. Eşik değeri sayısal olarak doğrudan verilebileceği gibi, ortalama (mean), ortanca (median) gibi yerleşik sezgisel yöntemlerin kullanılmasıyla da elde edilebilir. Diğer taraftan, eşik değerinin belirlenmesine alternatif olarak; seçilmek istenilen maksimum öznitelik sayısı verilerek de istenilen sayıda önemli öznitelik seçimi gerçekleştirilebilmektedir.

Soru analizinin önemli bir parçası olan soru sınıflandırma için sınıflandırma algoritmalarının başarımını arttırabilmek amacı ile önerilen öznitelik seçim yöntemi, yukarıda bahsi geçen öznitelik seçimi yöntemlerinin hibritlenmesi ile elde edilmiştir. Önerilen öznitelik seçiminin temel mimarisine Şekil 4.1'de yer verilmiştir. İlk olarak, veri seti üzerinde Ki-kare yöntemi ile öznitelik seçim işlemi gerçekleştirilmiş, veri seti bir kez indirgenmiştir. Ardından, doğrusal çekirdekli SVM ve bir eşik değeri kullanılarak ikinci kez öznitelik seçimi gerçekleştirilmiştir.



Şekil 4.1. Soru sınıflandırma için önerilen öznitelik seçimi mimarisi.

4.1.2 Sınıflandırma yöntemlerinin birleştirilmesi (Hibrit yöntem)

Bu bölümde, sınıflandırma algoritmalarının soru sınıflandırmadaki başarımını arttırmak amacıyla önerilen birleştirme yöntemine (hibrit yönteme) değinilmiştir. Sınıflandırma algoritmalarını birleştirmedeki motivasyonumuz, her yöntemin yüksek doğrulukla sınıflandırdığı sınıfları bularak, yöntemlerin güçlü yanlarından faydalanabilmektir. Bu amaç doğrultusunda karmaşıklık matrisinden (confusion matrix) aşağıda anlatıldığı gibi yararlanılmıştır.

n yöntem sayısı ve s sınıf olmak üzere, m. yöntemin karmaşıklık matrisi C_m;

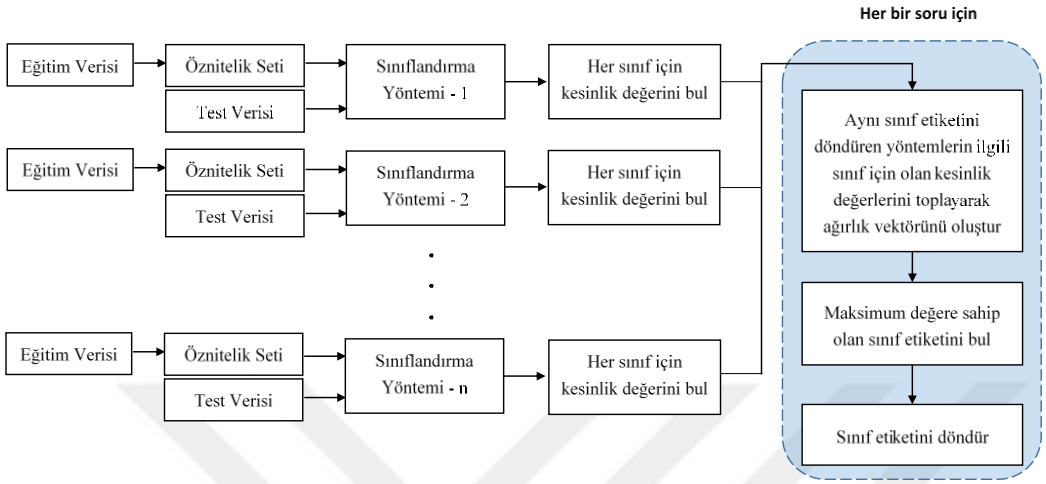
$$C_m = \begin{matrix} Predicted\ Class \\ \begin{bmatrix} C_m(1,1) & C_m(1,2) & \cdots & C_m(1,s) \\ C_m(2,1) & C_m(2,2) & \cdots & C_m(2,s) \\ \vdots & \vdots & \ddots & \vdots \\ C_m(s,1) & C_m(2,2) & \cdots & C_m(s,s) \end{bmatrix} \end{matrix} True\ Class \quad (4.2)$$

şeklindedir. Eşitlik 4.2'de $C_m(i,j)$, gerçekte i sınıfına ait olan ancak, sınıflandırıcı tarafından j olarak tahminlenen eleman sayısını göstermektedir. m. yöntemin i sınıfı için kesinlik değeri Eşitlik 4.3'te gösterildiği gibi hesaplanmıştır.

$$P_{m,i} = \frac{C_m(i,i)}{\sum_{j=1}^{s} C_m(j,i)} \quad (4.3)$$

Bu şekilde her bir yöntemin s sayıda sınıfı için kesinlik değeri hesaplanmış ve n adet $P_m = [P_{m,1}, P_{m,2}, \dots, P_{m,s}]$ şeklinde bir vektör elde edilmiştir. Ardından, her bir soru için sınıflandırma yöntemlerinin bulduğu sınıf etiketlerine bakılarak bir ağırlık vektörü hesaplanmıştır. Elde edilen ağırlık vektöründe, en yüksek ağırlığa sahip olan sınıf etiketi, nihai sınıf etiketi olarak döndürülmüştür. Önerilen hibrit yöntemin temel mimarisi Şekil 4.2'de sunulmuştur. Hesaplama işlemini, 3 yöntem ve 4 sınıf sayısının olduğu bir örnek ile açıklayacak olursak: Herhangi bir a sorusuna x yöntemi 1 sınıf etiketini, y yöntemi 3 sınıf etiketini ve z yöntemi 1 sınıf etiketini atıyorsa, ağırlık vektörü $W_a = \left[(P_{x,1} + P_{z,1}), 0,\ P_{y,3}, 0\right]$ şeklinde olur. Elde edilen ağırlık vektöründeki en büyük değer hangi sınıfa ait ise ilgili soruya o sınıf

etiketi atanmıştır. Yukarıdaki örnek için, $(P_{x,1} + P_{z,1}) > P_{y,3}$ ve $(P_{x,1} + P_{z,1}) > 0$ olduğunu farz edersek, en büyük ağırlık değeri 1 etiketli sınıfa ait olduğu için a sorusuna 1. sınıf etiketi atanmıştır.



Şekil 4.2. Önerilen hibrit yöntem mimarisi.

4.2 Soru Yanıtlama İçin Önerilen Yöntemler

Bu bölümde, soru yanıtlama sistemleri için önerilen hibrit soru yanıtlama yöntemlerinin genel yapısı sunulmaktadır. Önerilen yaklaşımlar, soru yanıtlama problemi için literatürde sıklıkla kullanılan derin öğrenme modelleri, topluluk öğrenmesi yöntemlerinden faydalanılarak birleştirilmiştir. Birleştirme işlemini gerçekleştirmedeki temel motivasyon, bir modelin kaçırdığı cevap cümlesini başka bir modelin yakaladığını deneysel çalışmalar sonucunda gözlemlemiş olmaktır.

Aniol et al. (2019)'un çalışmasından esinlenilerek önerilen yaklaşımlar için soru yanıtlama problemi, bir sınıflandırma problemi olarak ele alınmıştır. Önerilen yaklaşımlar için, kullanılan modellerden elde edilen sonuçların (cevap cümlelerinin) karmaşıklık matrisinden faydalanılmıştır. Her bir modelin her bir sınıf için elde ettiği kesinlik (precision), duyarlılık (recall), F1-skor (F1-score) ve doğruluk (accuracy - ACC) değerleri hesaplanmıştır.

M yöntem kümesi, $M_i, M_j \in M$ ve k sınıf sayısı olmak üzere $C = \{C_1, C_2, C_3, \dots, C_k\}$ sınıf etiketi kümesi ve $C_n, C_m \in C$ iken;

- P_{iC_n} ve P_{jC_n} sırası ile M_i ve M_j yöntemlerinin C_n sınıfı için ve, P_{iC_m} ve P_{jC_m} sırası ile M_i ve M_j yöntemlerinin C_m sınıfı için kesinlik değerleri,

- R_{iC_n} ve R_{jC_n} sırası ile M_i ve M_j yöntemlerinin C_n sınıfı için ve, R_{iC_m} ve R_{jC_m} sırası ile M_i ve M_j yöntemlerinin C_m sınıfı için duyarlılık değerleri,

- F_{iC_n} ve F_{jC_n} sırası ile M_i ve M_j yöntemlerinin C_n sınıfı için ve, F_{iC_m} ve F_{jC_m} sırası ile M_i ve M_j yöntemlerinin C_m sınıfı için F1-Skorları,

- ACC_{iC_n} ve ACC_{jC_n} sırası ile M_i ve M_j yöntemlerinin C_n sınıfı için ve, ACC_{iC_m} ve ACC_{jC_m} sırası ile M_i ve M_j yöntemlerinin C_m sınıfı için doğruluk değerleri

olarak ifade edilmektedir. Bu doğrultuda;

➢ Eğer M_i ve M_j yöntemleri cevap cümlesi olarak aynı C_n sınıfını döndürüyorsa;

Yaklaşım 1: Skor = $F_{iC_n} + F_{jC_n}$

Yaklaşım 2: Skor = $F_{iC_n} * P_{iC_n} + F_{jC_n} * P_{jC_n}$

Yaklaşım 3: Skor = $ACC_{iC_n} * P_{iC_n} + ACC_{jC_n} * P_{jC_n}$

➢ Eğer M_i yöntemi cevap cümlesi olarak C_n sınıfını, M_j yöntemi cevap cümlesi olarak C_m sınıfını döndürüyor ise;

Yaklaşım 1: Skor = $\begin{cases} F_{iC_n} > F_{jC_m}\ ise, & F_{iC_n} \\ Aksi\ taktirde, & F_{jC_m} \end{cases}$

Yaklaşım 2: $\text{Skor} = \begin{cases} F_{iC_n} > F_{jC_m}\ ise, & F_{iC_n} * P_{iC_n} \\ Aksi\ taktirde, & F_{jC_m} * P_{jC_m} \end{cases}$

Yaklaşım 3: $\text{Skor} = \begin{cases} F_{iC_n} > F_{jC_m}\ ise, & ACC_{iC_n} * P_{iC_n} \\ Aksi\ taktirde, & ACC_{jC_m} * P_{jC_m} \end{cases}$

şeklinde hesaplanmaktadır. Maksimum skor değerine sahip olan sınıf etiketi çözüm olarak döndürülmektedir.

Aynı cevap cümlesini seçen modellerin ağırlık değerlerini toplamak, çoğunluğun önerdiği değerleri seçmeye yönelttiği için, toplama işlemi yerine aynı cevap cümlesini seçen modellerin ağırlık değerlerinin aritmetik ve geometrik ortalamaları kullanılmıştır. Bu doğrultuda önerilen yaklaşımlar;

Yaklaşım 4:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort(F_{iC_n}, F_{jC_n}) \\ Aksi\ taktirde, & \text{F1} - \text{Skor} \end{cases}$$

Yaklaşım 5:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort(F_{iC_n} * P_{iC_n}, F_{jC_n} * P_{jC_n}) \\ Aksi\ taktirde, & \text{F1} - \text{Skor} * \text{Kesinlik değeri} \end{cases}$$

Yaklaşım 6:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort(ACC_{iC_n} * P_{iC_n}, ACC_{jC_n} * P_{jC_n}) \\ Aksi\ taktirde, & \text{Doğruluk} * \text{Kesinlik değeri} \end{cases}$$

Yaklaşım 7:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & geo\_ort(F_{iC_n}, F_{jC_n}) \\ Aksi\ taktirde, & \text{F1} - \text{Skor} \end{cases}$$

Yaklaşım 8:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa,} & geo\_ort(F_{iC_n} * P_{iC_n}, F_{jC_n} * P_{jC_n}) \\ Aksi\ taktirde, & \text{F1} - \text{Skor} * \text{Kesinlik değeri} \end{cases}$$

Yaklaşım 9:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa,} & geo\_ort(ACC_{iC_n} * P_{iC_n}, ACC_{jC_n} * P_{jC_n}) \\ Aksi\ taktirde, & \text{Doğruluk} * \text{Kesinlik değeri} \end{cases}$$

şeklindedir. Burada en büyük skor değerine sahip olan aday cümle id'si, cevap cümle id'si olarak kabul edilmektedir.

Kullanılan modeller içerisinden çoğunluğun seçtiği cevap cümlesini, daha az seçilen cevap cümlelerine göre daha fazla ödüllendirecek başka yaklaşımlar da önerilmiştir. l, model sayısı ve z, aynı cevap cümlesini döndüren model sayısı olmak üzere, modellerin ödül ağırlıkları (AW) Eşitlik 4.4 yardımıyla hesaplanmaktadır.

$$AW = \frac{z}{l} \tag{4.4}$$

Ödüllendirme işlemini Tablo 4.1'de verilen ve cevabı içeren cümle id'si "2" olan bir örnek ile açıklayacak olursak; tablodaki örnekte, bir model (RoBERTa) 1. cümleyi cevap cümlesi olarak, iki model (Albert ve DistilBERT) 3. cümleyi cevap cümlesi olarak, üç model (BERT, XLM ve XLNet) 5. cümleyi cevap cümlesi olarak döndürmektedir. Örnekte toplam altı model kullanıldığı için aynı sınıf etiketini döndüren model sayısı, toplam model sayısına oranlanarak her bir model için ödül ağırlık değeri oluşturulmuştur.

Tablo 4.1. SQuAD eğitim veri setinde bir soru için elde edilen örnek bir çözüm kümesi ve ona ait olan ağırlık değerleri.

	Albert	**Bert**	**Distilbert**	**Roberta**	**XLM**	**XLNet**
Tahmin	2	4	2	0	4	4
Ağırlık	$\frac{2}{6}$	$\frac{3}{6}$	$\frac{2}{6}$	$\frac{1}{6}$	$\frac{3}{6}$	$\frac{3}{6}$

Anlatılanlar ışığında, önerilen ödüllendirme yaklaşımları;

Yaklaşım 10:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort\left(F_{iC_n}, F_{jC_n}\right) * AW \\ Aksi\ taktirde, & \text{F1} - \text{Skor} * AW \end{cases}$$

Yaklaşım 11:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort(F_{iC_n} * P_{iC_n}, F_{jC_n} * P_{jC_n}) * AW \\ Aksi\ taktirde, & \text{F1} - \text{Skor} * \text{Kesinlik değeri} * AW \end{cases}$$

Yaklaşım 12:

$$Skor = \begin{cases} M_i \text{ ve } M_j \text{ yöntemleri aynı } C_n \text{sınıfını döndürüyorsa}, & ort(ACC_{iC_n} * P_{iC_n}, ACC_{jC_n} * P_{jC_n}) * AW \\ Aksi\ taktirde, & \text{Doğruluk} * \text{Kesinlik değeri} * AW \end{cases}$$

şeklindedir. Burada en büyük skor değerine sahip olan sınıf etiketi, cevap cümle id'si olarak kabul edilmektedir.

Aday cevap cümleleri elde edilirken, paragraf cümleleri ve soru cümlesi arasındaki Kosinüs benzerliğine bakıldığından önceki bölümlerde bahsedilmişti. Paragraf cümleleri, Kosinüs benzerliğine göre sıralandığı için cevap cümlesinin seçiminde önemli bir rol oynamaktadır. Bu nedenle modelleri birleştirirken Kosinüs benzerliği kullanan yaklaşım:

Yaklaşım 13:

$$Skor = \begin{cases} M_i, M_j \text{ yöntemleri } C_n \text{sınıfını veriyorsa,} & ort(ACC_{iC_n} * P_{iC_n} * F_{iC_n}, ACC_{jC_n} * P_{jC_n} * F_{jC_n}) * CosSim \\ Aksi\ taktirde, & \text{Doğruluk} * \text{Kesinlik değeri} * \text{F1} - \text{Skor} * CosSim \end{cases}$$

şeklindedir. Cevap cümlesi, en büyük skor değerine sahip olan sınıf etiketi olarak alınmaktadır.

5. DENEYSEL SONUÇLAR

Bu bölümde, tez çalışması kapsamında kullanılan veri setlerine, veri setlerine uygulanan önişlemlere, yöntemlerin başarımlarının ölçümlenmesi için kullanılan değerlendirme ölçütlerine ve geliştirilen yöntemlerden elde edilen deneysel sonuçlara yer verilmiştir.

5.1 Veri Setleri

Tez çalışması kapsamında geliştirilen soru sınıflandırma yöntemleri için TREC[1] soru sınıflandırma veri seti, soru yanıtlama yöntemleri için ise SQuAD (Stanford Question Answering Dataset)[2] veri seti kullanılmıştır.

5.1.1 TREC veri seti

Soru ve cevap sınıfları olarak düşünülen önceden tanımlanmış kategorilere "soru taksonomisi (question taxonomy)" veya "cevap türü taksonomisi (answer type taxonomy)" adı verilir (Loni, 2011). Tablo 5.1'de görülen Li ve Roth (2002) tarafından önerilen iki katmanlı taksonomi soru sınıflandırma için sıklıkla kullanılmaktadır (Mohasseb vd., 2018). Bu taksonomi altı kaba-taneli (coarse-grained) sınıftan oluşmaktadır. Bu kaba-taneli sınıflar: *abbreviation*, *entity*, *description*, *human*, *location*, ve *numeric*. Veri seti içerisinde, *abbreviation* sınıfının altında iki, *entity* sınıfının altında yirmi iki, *description* sınıfının altında dört, *human* sınıfının altında dört, *location* sınıfının altında beş ve *numeric* sınıfının altında on üç olmak üzere toplamda elli adet ince-taneli sınıf bulunmaktadır. Makine öğrenmesinde sıkça kullanılan bu taksonomi, 5452 tanesi eğitim verisi ve 500 tanesi test verisi olmak üzere toplamda 5952 adet etiketli soru içermektedir.

5.1.2 SQuAD (Stanford Question Answering Dataset)

Stanford Üniversitesi'nde doktora eğitimine devam etmekte olan Rajpurkar ve arkadaşları (2016) tarafından önerilen SQuAD, Wikipedia'daki 500'den fazla

[1] https://cogcomp.seas.upenn.edu/Data/QA/QC/
[2] https://rajpurkar.github.io/SQuAD-explorer/

makaleden oluşturulmuş bir soru yanıtlama veri setidir. Bu veri setinde, her bir sorunun cevabı kendisine bağlı olan paragrafın bir parçasıdır. Buradaki amaç, bağlı olunan paragraftan cevap parçasını bulup çıkarmaktır.

Tablo 5.1. Li vand Roth (2002) taksonomisi (Sundblad, 2007).

Sınıf	**Tanımı**	**Sınıf**	**Tanımı**
ABBREVIATION	abbreviation	HUMAN	human beings
abb	abbreviation	group	a group or organization of persons
exp	expression abbreviated	ind	an individual
ENTITY	entities	title	title of a person
animal	animals	description	description of a person
body	organs of body	LOCATION	locations
color	colors	city	cities
creative	inventions, books and other creative pieces	counrty	countries
currency	currency names	mountain	mountains
dis.mend.	disease and medicine	other	other locations
event	events	state	states
food	food	NUMERIC	numeric values
instrument	musical instrument	code	postcodes or other codes
lang	languages	count	number of
letter	letters like a-z	date	
other	other entities	distance	
plant	plants	money	
product	products	order	
religion	religions	other	
sport	sports	period	
substance	elements and substances	percent	
symbol	symbols and signs	speed	
technique	techniques and methods	temp	
term	equivalent terms	size	
vehicle	vehicles	weight	
word	words with a special property		
DESCRIPTION	description and abstract concepts		
definition	definition of sth.		
description	description of sth.		
manner	manner of an action		
reason	reasons		

SQuAD veri setinin, SQuAD v1.1 ve v2.0 olmak üzere iki versiyonu bulunmaktadır. SQuAD v1.1, 107785 soru-cevap çifti içermektedir. SQuAD v1.1 veri setinden örnek bir paragraf ve ona bağlı olan beş soru-cevap çifti Şekil 5.1'de gösterilmiştir. SQuAD v2.0, mevcut SQuAD v1.1 veri setine çözümsüz 53775 soru eklenmesi ile elde edilmiştir. SQuAD v2.0 veri setinde başarılı olmak için, sadece

soruları cevaplamakla kalınmamalı, aynı zamanda paragrafta hiçbir cevabın olmadığı belirlenmeli ve soruya bir cevap verilmemelidir.

Article: University of Notre Dame

Paragraph: *"...The library system of the university is divided between the main library and each of the colleges and schools. The main building is the **14**-story **Theodore M. Hesburgh Library**, completed in **1963**, which is the third building to house the main collection of books. The front of the library is adorned with the Word of Life mural designed by artist **Millard Sheets**. This mural is popularly known as "**Touchdown Jesus**" because of its proximity to Notre Dame Stadium and Jesus' arms appearing to make the signal for a touchdown."*

Question 1: *"How many stories tall is the main library at Notre Dame?"*
Answer: **14**

Question 2: *"What is the name of the main library at Notre Dame?"*
Answer: **Theodore M. Hesburgh Library**

Question 3: *"In what year was the Theodore M. Hesburgh Library at Notre Dame finished?"*
Answer: **1963**

Question 4: *"Which artist created the mural on the Theodore M. Hesburgh Library?"*
Answer: **Millard Sheets**

Question 5: *"What is a common name to reference the mural created by Millard Sheets at Notre Dame?"*
Answer: **Touchdown Jesus**

Şekil 5.1. SQuAD v1.1 veri setinden örnek bir paragraf ve soru-cevap çiftleri.

Tez kapsamında yapılan deneysel çalışmalar, SQuAD v1.1 veri seti üzerinde gerçekleştirilmiştir.

5.2 Veri Önişleme

Tezin bu bölümünde, soru sınıflandırma ve soru yanıtlama işlemini doğru bir şekilde yapabilmek için veri setleri üzerinde uygulanan ön işlemlerden bahsedilmiştir.

k5.2.1 TREC veri seti üzerine uygulanan önişlemler

İlk olarak, TREC veri setinde yer alan altı kaba-taneli ve elli ince-taneli sınıftan oluşan sorular, sınıf etiketlerine göre kategorize edilmiştir. Ardından eğitim seti kullanıma uygun bir formata getirilmiştir. Bunun için eğitim setinde yer alan sorular ve soruların sınıf etiketleri ayrıştırılmıştır. Ayrıştırma işlemi yapılırken soru cümlelerinin sonundaki boşluklar ve cümle içinde yer alan nokta, virgül, ünlem gibi

özel karakterler kaldırılmış ve metin sınıflandırmasında yaygın olarak kullanılan ön işlemlerden olan belirteçlerine ayırma (tokenization), gereksiz kelimelerin kaldırılması ve stemming/lemmatization (kök bulma/ kök çözümleme) işlemleri (Uysal and Günal, 2014) uygulanmıştır. Önişlem uygulanmış bu soru listesine derlem denir.

Belirteçlere ayırma, bir metinin sözcüklere, sözcük öbeklerine veya diğer anlamlı parçalara bölünmesi işlemidir. Başka bir deyişle, bir metin bölümleme biçimidir. Bu işlem, alfa numerik olmayan karakterler (noktalama işaretleri, boşluklar vb.) tarafından sınırlandırılarak sadece alfabetik veya alfa numerik karakterler dikkate alınarak gerçekleştirilmektedir. Bağlaçlar, edatlar gibi belirli bir bağlama bağlı olmayan kelimeler metinlerde sıkça karşılaşılan kelimelerdir. Bu kelimeler belgenin içeriği hakkında bilgi vermemektedir. Sıkça tekrarlayan bu kelimelerin frekanslarını doğrudan sınıflandırıcıya vermek, daha az tekrarlayan ve daha faydalı olan kelimelerin gözden kaçmasına neden olmaktadır. Bu nedenle, bu sözcüklerinin genellikle metin sınıflandırma çalışmalarında ilgisiz olduğu varsayılarak, sınıflandırma işlemi yapılmadan önce kaldırılmaktadır. Bir örnek verecek olursak, aşağıda verilen soru cümlesindeki “did, in, and, then” kelimeleri sorudan kaldırılmıştır.

Soru: *How **did** serfdom develop **in and then** leave Russia ?*

Önişlemden geçmiş soru: *How serfdom develop leave Russia ?*

Hem stemming hem de lemmatization işleminin amacı, çekimlenmiş ve türetilmiş olan kelimeleri ortak bir biçime (köke) indirgemektir (Manning et al., 2009). Örneğin, "token, tokenizer, tokenization" kelimeleri aynı köke sahip olan, anlamları birbirine yakın ancak birbirinden farklı olan kelimelerdir. Stemming, genellikle türetilmiş kelimelerin sonlarını gelişigüzel kesen sezgisel bir süreç iken; lemmatization, kelimelerin morfolojik analizi ile yapılan bir işlemdir. Lemmatization, kelimelerde sadece çekimli kısımları kaldırmayı ve bir kelimeyi köküne döndürmeyi amaçlamaktadır. Örneğin, "caught" kelimesi stemming işlemine tabi tutulduğunda, kelimelerin sonları gelişigüzel kesildiği için "caug" kelimesi döndürülebilir. Aynı kelime lemmatization işlemine tabi tutulduğunda ise

morfolojik analiz yapıldığı için kelimenin isim veya fiil olmasına bağlı olarak "caught" veya "catch" döndürebilmektedir.

5.2.2 SQuAD veri seti üzerine uygulanan önişlemler

Json türünde olan SQuAD veri setinin kullanıma uygun bir hale getirilmesi için, veri setinde yer alan veriler sırasıyla, paragraflar, bu paragraflara bağlı olan sorular ve cevaplar olarak ayrıştırılmış ve bir csv dosyası oluşturulmuştur. Paragrafları işleyebilmek için, paragraflar da cümlelere bölünmüştür. Ardından, paragraf cümleleri ve soru cümleleri öznitelik vektörleri ile temsil edilmiştir.

5.3 Değerlendirme Ölçütleri

Sınıflandırma algoritmalarının etkili bir sınıflandırma yapıp yapmadığını belirleyebilmek için bazı değerlendirme ölçütleri kullanılmaktadır. Farklı değerlendirme ölçütleri, sınıflandırıcıların farklı özelliklerini elde aldığı için değerlendirme ölçütü seçimi önem taşımaktadır (Hossin and Sulaiman, 2015). Tez çalışması kapsamında yapılan deneysel çalışmaların değerlendirilmesinde, literatürde sınıflandırma algoritmaları için sıklıkla tercih edilen "doğruluk" ve "F1-skor" ölçütü kullanılmıştır. Doğruluk ölçütü, sınıflandırıcının doğru tahmin ettiği örneklerin sayısının, veri setinde yer alan toplam örnek sayısına bölünmesiyle elde edilmektedir. Doğruluk ölçütü,

$$doğruluk\ (acc) = \frac{TP+TN}{TP+FP+TN+FN} \tag{5.1}$$

şeklinde hesaplanmaktadır. Eşitlik 5.1'de yer alan;

- **Doğru Pozitif (True Positive – TP):** Pozitif olarak sınıflandırılan örneklerin gerçekte de pozitif olduğu durumlar
- **Yanlış Pozitif (False Positive – FP):** Pozitif olarak sınıflandırılan örneklerin gerçekte negatif olduğu durumlar
- **Yanlış Negatif (False Negative – FN):** Negatif olarak sınıflandırılan örneklerin gerçekte pozitif olduğu durumlar

- **Doğru Negatif (True Negative –TN):** Negatif olarak sınıflandırılan örneklerin gerçekte de negatif olduğu durumlar

olarak ifade edilebilir. Doğruluk ölçütünün değeri [0,1] aralığında değişmektedir. Bu değerin 1'e yakın olması sınıflandırıcının başarımının iyi olduğunun, 0'a yakın olması ise sınıflandırıcının başarımının kötü olduğunun bir göstergesidir.

Kesinlik ölçütü, modelin pozitif olarak tahmin ettiği örneklerin, gerçekte kaç tanesinin pozitif olduğunu göstermektedir. Kesinlik ölçütü Eşitlik 5.2'ye göre hesaplanmaktadır.

$$kesinlik\ (P) = \frac{TP}{TP+FP} \tag{5.2}$$

Duyarlılık ölçütü, gerçekte pozitif olarak tahmin edilmesi gereken örneklerin ne kadarını pozitif olarak tahmin ettiğini göstermektedir. Duyarlılık ölçütü Eşitlik 5.3 yardımıyla hesaplanmaktadır.

$$duyarlılık\ (R) = \frac{TP}{TP+FN} \tag{5.3}$$

F1-skor, kesinlik ve duyarlılık ölçütlerinin harmonik ortalamasının alınmasıyla elde edilmektedir. F1-skor ölçütü Eşitlik 5.4 kullanılarak hesaplanmaktadır.

$$F1-skor\ = 2 * \frac{P*R}{P+R} \tag{5.4}$$

Veri setinde dengesiz (eşit olmayan) bir sınıf dağılımı söz konusu olduğunda, F1-skor ölçütünü kullanmak uygundur. Çok sınıflı sınıflandırma problemleri için F1-skor ölçütü, Mikro ortalama (Micro-average), Makro ortalama (Macro-average) ve Ağırlıklı ortalama (Weighted-average) olmak üzere üç şekilde hesaplanmaktadır. Mikro F1-skor, bütün sınıflar aynı anda ele alınarak hesaplanan genel kesinlik değerini ifade etmektedir ve Eşitlik 5.5 yardımı ile hesaplanır.

$$Mikro\ F1-skor = \frac{TP_{tüm}}{TP_{tüm}+FP_{tüm}} \tag{5.5}$$

Makro F1-skor, her bir sınıf için ayrı ayrı hesaplanan kesinlik değerlerinin ortalamasıdır. c sınıf sayısı ve P_c, c. sınıfın kesinlik değeri olmak üzere, Makro F1-skor Eşitlik 5.6 kullanılarak elde edilmektedir.

$$Makro\ F1 - skor = \frac{P_1 + P_2 + \cdots + P_c}{c} \tag{5.6}$$

Ağırlıklı F1-skor ise, her bir sınıfın kesinlik değerinin, ilgili sınıfa ait örnek sayısı ile ağırlıklandırılarak ortalamasının alınması ile elde edilmektedir. c sınıf sayısı; P_c, c. sınıfın kesinlik değeri ve N_i, i. sınıfın örnek sayısı olmak üzere Ağırlıklı F1-skor Eşitlik 5.7 kullanılarak hesaplanmaktadır.

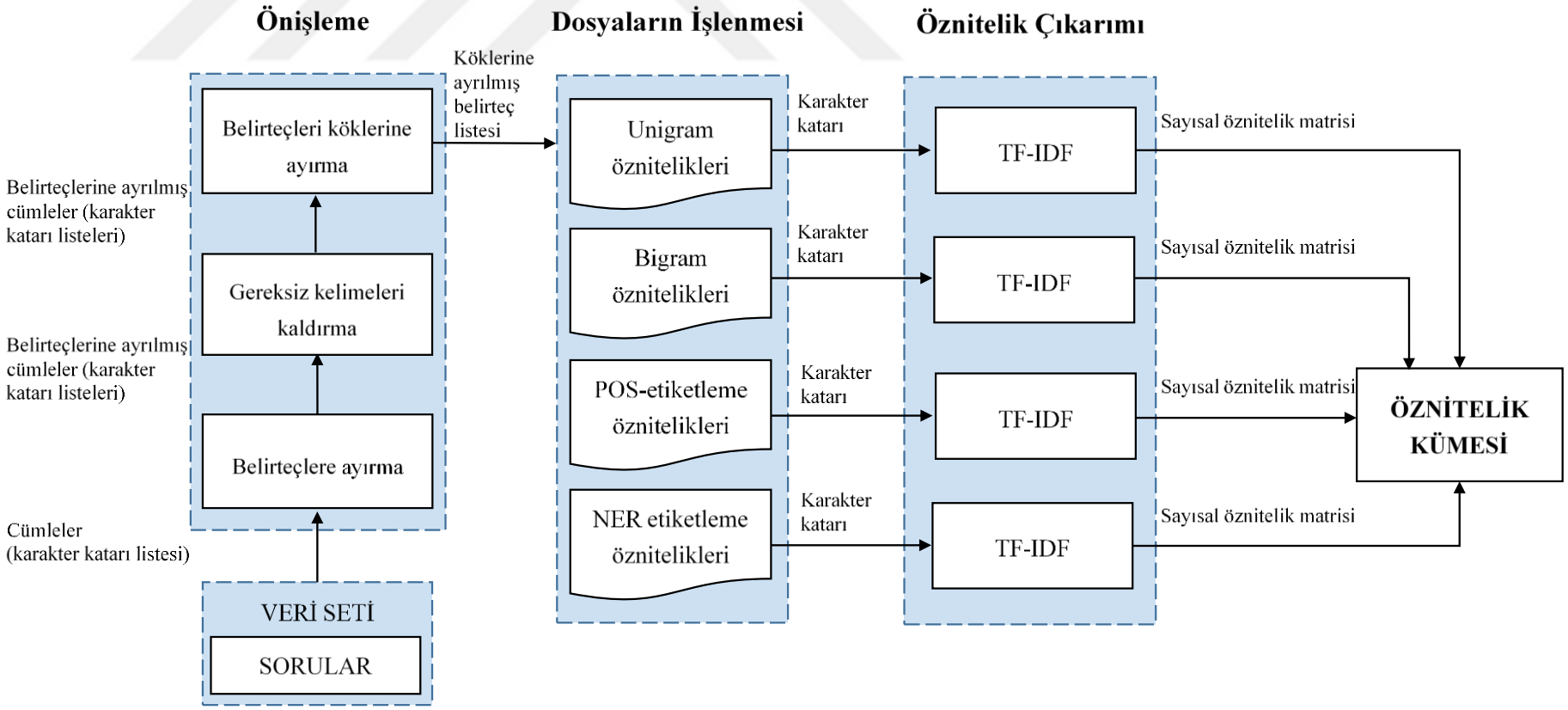
$$Ağırlıklı\ F1 - Skor = \frac{P_1 * N_1 + P_2 * N_2 + \cdots + P_c * N_c}{(N_1 + N_2 + \cdots + N_c)} \tag{5.7}$$

5.4 Soru Sınıflandırma Deneysel Sonuçları

Tezin bu bölümünde, geliştirilen soru sınıflandırma mimarisinde kullanılan öznitelik temsiline, soru sınıflandırma için izlenilen deneysel sürece ve kullanılan değerlendirme ölçütleri ile elde edilen deney sonuçlarına yer verilmiştir.

5.4.1 Soru sınıflandırma için öznitelik temsilinin uygulanması

Öğrenme tabanlı yaklaşımları kullanırken dikkat edilmesi gereken en önemli hususlardan birisi de verileri temsil eden özniteliklerin seçilmesidir. Soru sınıflandırma aşaması için soruları temsil edecek öznitelik vektörünün oluşturulması için unigram (BoW), POS etiketleyici, NER etiketçisi ve bölümleme (chunking) kullanılmıştır. Uygulanan öznitelik temsilinin genel yapısına Şekil 5.2'de yer verilmiştir.



Şekil 5.2. Eğitim ve test veri setinin hazırlanması.

Önişlemden geçirilen derlem (Bkz. Bölüm 5.2.1) üzerinde, ilk olarak her bir soru cümlesinin unigram ve bigram öznitelikleri elde edilmiştir. Bu işlem için Bölüm 3.1.1.1'de sunulan bilgilerden faydalanılmıştır. İkinci olarak, aynı derlem üzerinde, POS etiketleme işlemi gerçekleştirilmiştir. Bu amaç doğrultusunda, derlemdeki cümleler için ayrı ayrı oluşturulan belirteç listesi, NLTK kütüphanesinde yer alan sözcük türü etiketleyicisi yardımı ile isim, fiil, sıfat gibi etiketlerle etiketlenmiştir. Örneğin, "When was Ozzy Osbourne born?" sorusu üzerinde sözcük türü etiketleyici uygulandığında "WRB VBD NNP NNP VBN ." çıktısı elde edilmiştir. Her soru cümlesi için elde edilen POS etiketleri bir sonraki aşamada kullanılmak üzere saklanmıştır.

Üçüncü olarak, önişlemden geçirilen derlem üzerinde NER etiketçisi uygulanmıştır. Bu işlem için, Stanford Üniversitesi Doğal Dil İşleme Grubu tarafından geliştirilen Java tabanlı StanfordNERTagger yazılımı kullanılmıştır. NER etiketçisi, metin içerisinde yer alan kişi ve şirket adları, konum bilgisi, zaman bilgisi gibi kelimeleri etiketlemektedir (Finkel et al., 2005). Derlemde yer alan her soru cümlesi için elde edilen NER etiketleri sonraki aşamada kullanılmak üzere saklanmıştır.

Son olarak, derlem üzerinde bölümleme işlemi uygulanmıştır. Bu işlem "yüzeysel parçalama (shallow parsing)" olarak da adlandırılmaktadır. Bölümleme, yapılandırılmamış metinden öbek (phrase) çıkarma işlemidir (Bird et al., 2009). POS etiketleme, kelimenin isim, fiil, sıfat vb. olup olmadığını gösterirken; cümledeki tamlamalar, öbekler veya cümlenin yapısı hakkında bilgi vermemektedir. Bölümleme etiketleri;

- **S:** Başında bulunduğu kelimenin tek kelime olduğunu gösterir.
- **B:** Etiket, başında bulunduğu kelime ile başlar.
- **I:** Başında bulunduğu kelime ilgili etikete dahildir.
- **E:** Etiket, başında bulunduğu kelime ile biter.
- **O:** Diğer etiketler.

şeklindedir. Örneğin, "What country has the largest sheep population ?" sorusu üzerinde bölümleme işlemi yapıldığında, elde edilen çıktı; "B-NP E-NP S-VP B-

NP I-NP I-NP E-NP O" şeklindedir. Burada, "What country" isim öbeği (noun phrase -NP), "has" fiil öbeği (verb phrase - VP), "the largest sheep population" isim öbeği ve "?" diğer etiket olarak ifade edilmektedir. Bölümleme işlemi için, Stanford Dependency Extractor ve SENNA üzerinde geliştirilen practNLPTools adı verilen bir python kütüphanesinden faydalanılmıştır. Her bir soru cümlesi için elde edilen bölümleme etiketleri bir sonraki aşamada kullanılmak üzere saklanmıştır.

Sıkça tekrarlayan kelimelerin frekanslarını doğrudan sınıflandırıcıya vermek, daha az tekrarlayan ve daha faydalı olan kelimelerin gözden kaçmasına neden olmaktadır. Bu nedenle bu değerleri sınıflandırıcıya vermeye uygun hale getirmek için, bu değerler yeniden ağırlıklandırılmıştır. Bu ağırlıklandırma işlemi için literatürde de sıkça kullanılan TF-IDF (Bkz. Bölüm 3.1.1.2) yönteminden yararlanılmıştır. Önceki adımlardan elde edilmiş olan unigram, POS, NER ve Chunk öznitelikleri TF-IDF yardımı ile sayısal forma dönüştürülerek TF-IDF terim-doküman matrisi oluşturulmuştur. Buradan elde edilen öznitelikler birleştirilerek tek bir öznitelik matrisi elde edilmiştir. Bahsedilen işlemlerin tamamı, hem eğitim hem de test veri seti için ayrı ayrı uygulanmıştır.

5.4.2 Soru sınıflandırma deneysel süreci ve sonuçları

Bu bölümde, TREC veri seti üzerinde kaba-taneli sınıflar için önerilen öznitelik seçiminin başarımının değerlendirilmesinde izlenen deneysel süreç ve uygulanan bireysel sınıflandırma algoritmalarının ve topluluk öğrenmesi yöntemlerinin deneysel sonuçları sunulmuştur.

Kullanılan bireysel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemleri Python programlama dili ile geçekleştirilmiştir. Deney sonuçları, i7 1.90 GHz işlemcili, 6GB RAM'e ve Linux işletim sistemine sahip bilgisayar üzerinde çalıştırılarak elde edilmiştir.

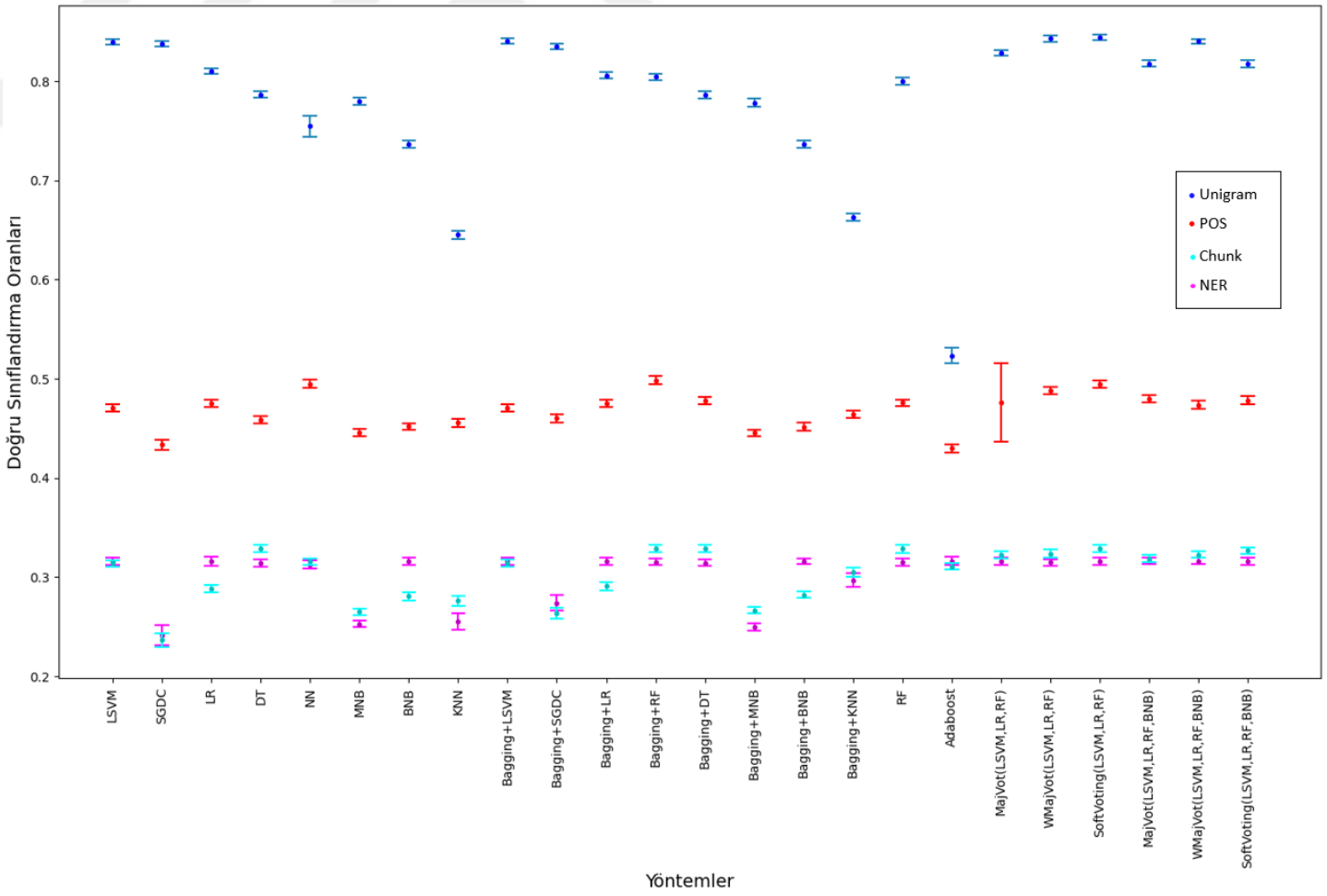
Soru sınıflandırma için kullanılan makine öğrenmesi yöntemlerinin başarımlarının incelenmesinde, 10-kat çapraz doğrulama (10-fold cross validation) yapılmıştır. 10-kat çapraz doğrulama yöntemi, verisetini rastgele on eşit parçaya bölmektedir. Elde edilen on parçadan dokuz tanesi eğitim için, geri kalan bir parça ise test için kullanılmaktadır. Her bir yinelemede test için kullanılan parça, bir adım

kaydırılarak aynı parçanın eğitim amaci ile kullanılması da sağlanmaktadır. On parçanın her biri test için kullanılıncaya dek bu süreç tekrarlanmaktadır. Sonuçlar tablosunda yer alan değerler, 10-kat çapraz doğrulama işleminin 10 defa tekrarlanması sonucunda elde edilen doğruluk değerlerinin ortalamasını göstermektedir.

Öznitelik vektörünün elde edilmesinin makine öğrenmesi temelli bir sınıflandırma modeli için en önemli işlem adımı olduğundan daha önce bahsedilmişti. Bu nedenle kullanılan özniteliklerin sınıflandırma için ne kadar etkili olduğunu öğrenmek adına bu öznitelikler öncelikle tek tek ele alınmıştır. Veri setindeki soruları, unigram ile temsil ettiğimizde 8680 öznitelik, POS ile temsil ettiğimizde 30 öznitelik, NER ile temsil ettiğimizde 3 öznitelik, CHUNK ile temsil ettiğimizde ise 7 öznitelik elde edilmiştir. Bu öznitelik vektörlerini ayrı ayrı sınıflandırıcılara tabi tuttuğumuzda elde edilen doğruluk sonuçlarına Tablo 5.2'de yer verilmiştir.

Kullanılan öznitelik vektörlerine bakıldığında, unigram öznitelik vektörü ile elde en iyi sonucun Yumuşak Oylama yöntemine ait olduğu gözlemlenmektedir. POS öznitelik vektörü kullanıldığında en iyi sonuç, Bagging algoritması ile topluluk öğrenme algoritmalarından olan Rastgele Orman algoritmasının birlikte kullanılmasıyla elde edilmektedir. NER öznitelik vektörü kullanıldığında ise başarılı olan sınıflandırma algoritması, yine bir topluluk öğrenme algoritması olan Adaboost'tur. Çoğunluk Oylama ve Ağırlıklı Çoğunluk Oylama algoritmaları da Adaboost'a oldukça yakın sonuçlar vermiştir. Son olarak, CHUNK öznitelik vektörüyle sınıflandırma işlemi yapıldığında ise DT algoritmasının başarılı olduğu gözlemlenmektedir. Genel olarak Tablo 5.2 incelendiğinde, en başarılı sınıflandırma algoritmasının % 84.4 ile unigram öznitelik vektörü üzerinde işletilen Yumuşak Oylama yöntemi olduğu görülmektedir.

Şekil 5.3'te unigram, POS, NER ve CHUNK öznitelikleri üzerinde karşılaştırılan temel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemlerine ilişkin %95 güven aralığı için güven aralığı grafiği sunulmaktadır. Grafikten gözlemlendiği üzere, unigram temsili üzerinde edilen sınıflandırma oranları istatistiksel olarak anlamlıdır.



Şekil 5.3. Unigram, POS, NER ve CHUNK öznitelikleri için tüm yöntemlere ilişkin güven aralığı grafiği.

Elde edilen sonuçlardan hareketle, unigram öznitelik türü ile POS, NER ve CHUNK öznitelik türleri ayrı ayrı birleştirilerek, bu öznitelik birleşimleri üzerinde soru sınıflandırma işlemi tekrar gerçekleştirilmiştir.

Sınıflandırma işleminden elde edilen sonuçlar Tablo 5.3'te sunulmuştur. Burada sınıflandırma algoritmalarının performansları öznitelik vektörü bazında incelendiğinde, unigram ve POS etiketlerinin birleştirilmesiyle elde edilen öznitelik vektörleri üzerinde doğrusal çekirdekli SVM algoritmasının en iyi performansı gösterdiği gözlemlenmiştir. Sorular, unigram ve NER öznitelikleri ile birlikte ifade edildiğinde de doğrusal çekirdekli SVM algoritması iyi sonuçlar vermiştir. Ayrıca, Unigram ile CHUNK öznitelikleri birleştirilerek sorular sınıflandırılmaya çalışıldığında ise yine doğrusal çekirdekli SVM algoritması başarılı olmuştur.

Tablo 5.3'teki sonuçlar genel olarak değerlendirildiğinde, unigram ve POS etiketlerinin birleştirilmesiyle elde edilen öznitelik vektörünün üzerinde doğrusal çekirdekli SVM algoritmasının %84.65 doğruluk oranı ile en yüksek sınıflandırma doğruluğunu verdiği gözlemlenmiştir.

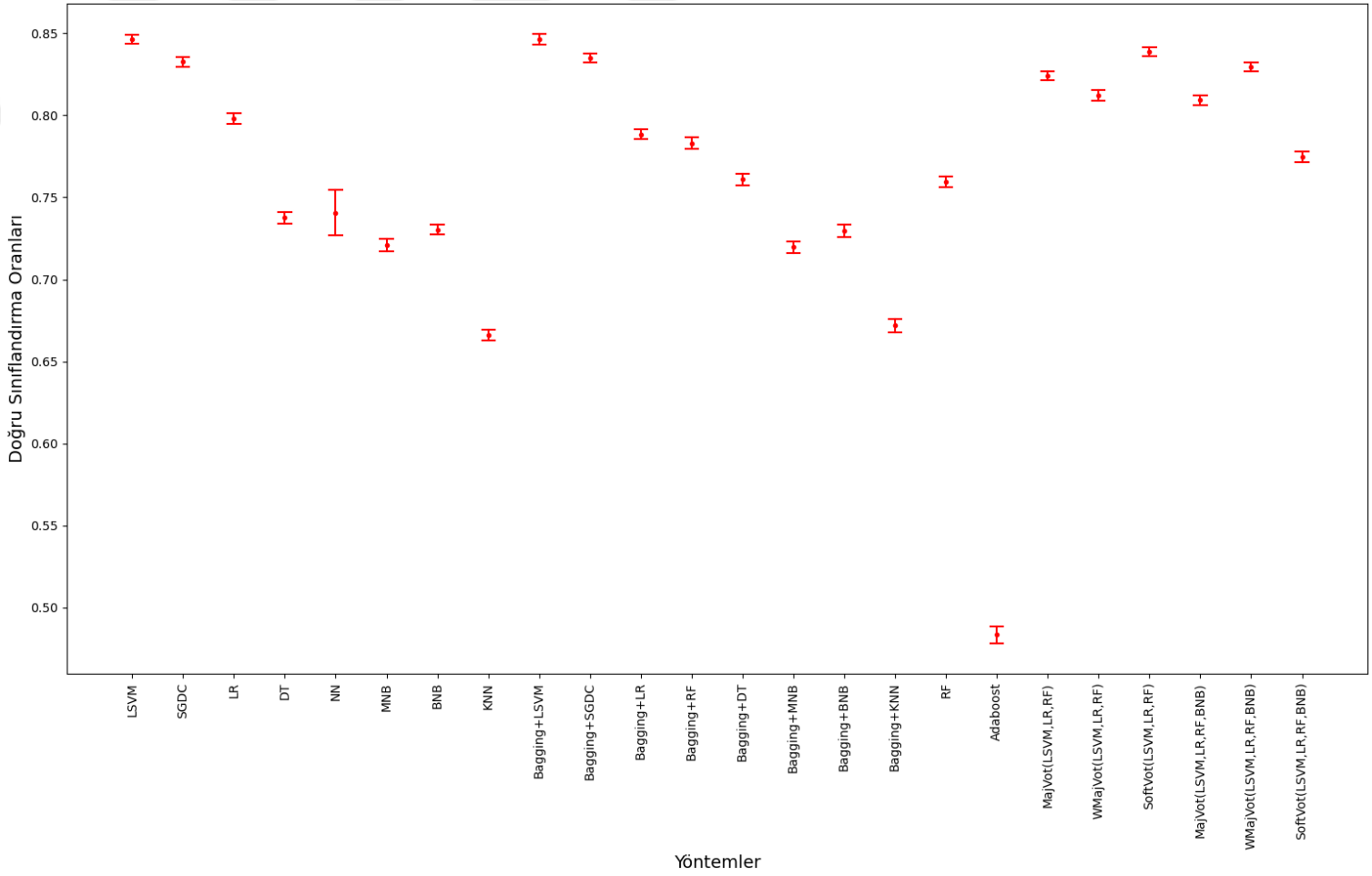
Şekil 5.4’te unigram + POS öznitelikleri, Şekil 5.5’te unigram + NER öznitelikleri ve Şekil 5.6’da unigram + CHUNK öznitelikleri üzerinde karşılaştırılan temel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemlerine ilişkin %95 güven aralığı için güven aralığı grafikleri sunulmaktadır. Şekil 5.4’te, doğrusal çekirdekli SVM sınıflandırıcısı ve Bagging algoritmasının doğrusal çekirdekli SVM ile oluşturduğu sınıflandırıcı topluluğunun diğer sınıflandırıcılara kıyasla istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği görülmektedir. Şekil 5.5’te, doğrusal çekirdekli SVM sınıflandırıcısı, Bagging algoritmasının doğrusal çekirdekli SVM ile oluşturduğu sınıflandırıcı ve Yumuşak Oylama yönteminin doğrusal çekirdekli SVM, LR, RF ve BNB ile oluşturduğu sınıflandırıcı topluluğunun diğer sınıflandırıcılara kıyasla istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği görülmektedir. Şekil 5.6’da ise, unigram + POS özniteliğinde olduğu gibi, doğrusal çekirdekli SVM sınıflandırıcısı ve Bagging algoritmasının doğrusal çekirdekli SVM ile oluşturduğu sınıflandırıcı topluluğunun diğer sınıflandırıcılara kıyasla istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği gözlemlenmektedir.

Tablo 5.2. Unigram, POS, NER ve CHUNK özniteliklerinin ayrı ayrı sınıflandırma algoritmalarına ve topluluk öğrenmesi yöntemlerine tabi tutulması ile elde edilen doğruluk sonuçları.

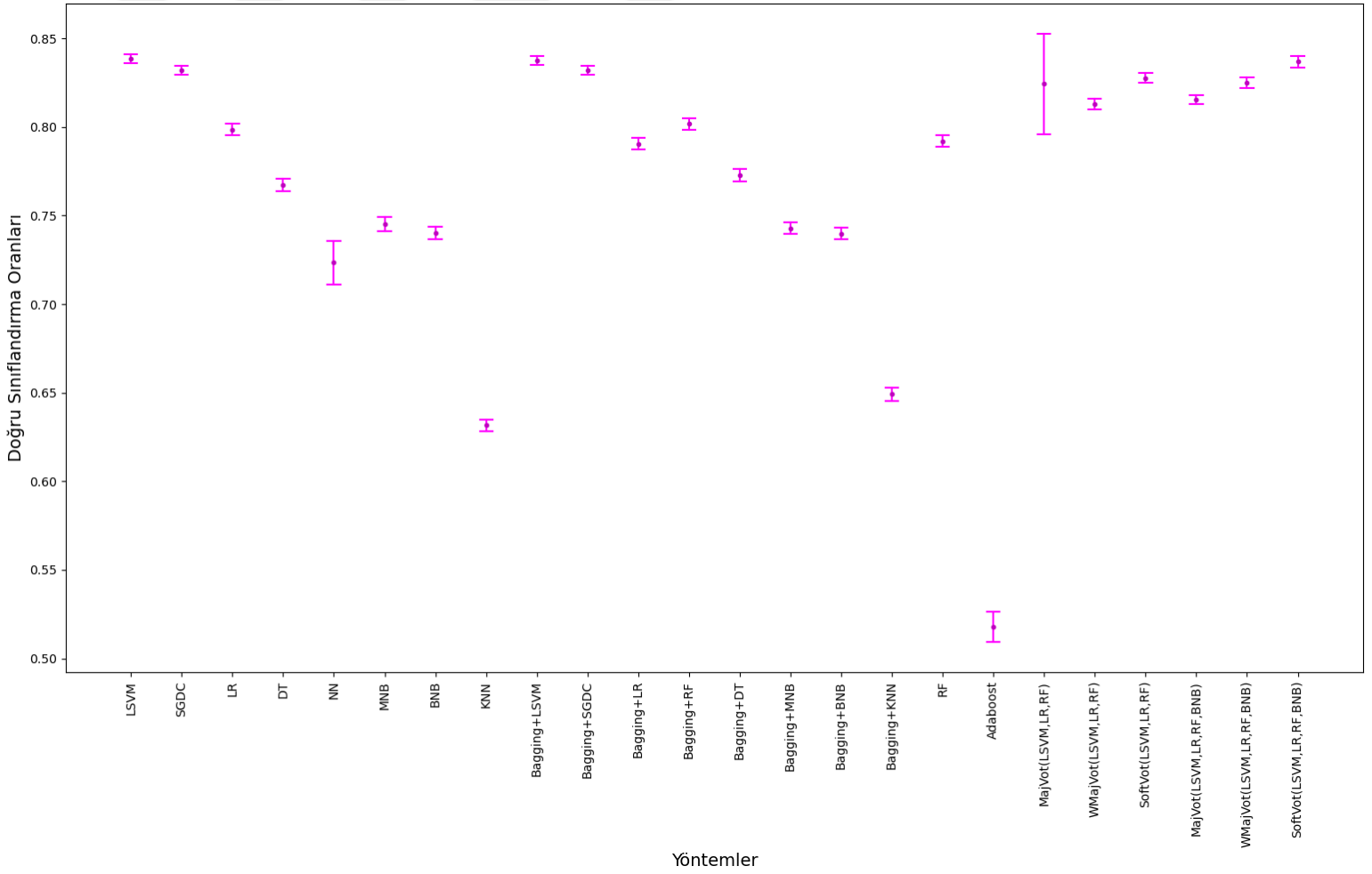
Sınıflandırıcı	Unigram		POS		NER		CHUNK	
	Ortalama	Standart Sapma	Ortalama	Standart Sapma	Ortalama	Standart Sapma	Ortalama	Standart Sapma
Linear SVM	0.8399	0.0147	0.4702	0.0187	0.3160	0.0183	0.3137	0.0181
SGDC	0.8381	0.0140	0.4334	0.0251	0.2414	0.0508	0.2367	0.0352
LR	0.8100	0.0154	0.4751	0.0196	0.3160	0.0219	0.2884	0.0209
DT	0.7865	0.0164	0.4589	0.0195	0.3143	0.0181	**0.3292**	0.0193
NN	0.7547	0.0529	0.4947	0.0211	0.3127	0.0231	0.3158	0.0169
Multinomial NB	0.7798	0.0198	0.4458	0.0178	0.2529	0.0173	0.2650	0.0177
Bernoulli NB	0.7365	0.0201	0.4521	0.0166	0.3160	0.0173	0.2807	0.0209
KNN	0.6450	0.0196	0.4557	0.0211	0.2554	0.0434	0.2760	0.0239
Bagging + LSVM	0.8407	0.0153	0.4706	0.0208	0.3160	0.0193	0.3141	0.0171
Bagging + SGDC	0.8349	0.0151	0.4602	0.0205	0.2738	0.0393	0.2634	0.0283
Bagging + LR	0.8060	0.0151	0.4754	0.0185	0.3160	0.0208	0.2910	0.0211
Bagging + RF	0.8042	0.0166	**0.4986**	0.0204	0.3151	0.0165	0.3287	0.0184
Bagging + DT	0.7861	0.0190	0.4779	0.0173	0.3146	0.0184	0.3285	0.0184
Bagging+ Multinomial NB	0.7782	0.0210	0.4454	0.0177	0.2497	0.0195	0.2667	0.0159
Bagging+ Bernoulli NB	0.7366	0.0178	0.4513	0.0210	0.3163	0.0143	0.2821	0.0170
Bagging+ KNN	0.6629	0.0182	0.4640	0.0196	0.2970	0.0362	0.3050	0.0237
RF	0.7999	0.0187	0.4757	0.0183	0.3149	0.0179	0.3286	0.0203
Adaboost	0.5235	0.0377	0.4298	0.0222	**0.3164**	0.0203	0.3107	0.0166
Majority Voting (LSVM, LR, RF)	0.8288	0.0141	0.4763	0.201	0.3160	0.0170	0.3228	0.0190
Weighted Majority Voting (LSVM, LR, RF)	0.8431	0.0165	0.4881	0.0201	0.3149	0.0171	0.3237	0.0201
Soft Voting (LSVM, LR, RF)	**0.8440**	0.0132	0.4945	0.0171	0.3157	0.0180	0.3287	0.0200
Majority Voting (LSVM, LR, RF, BNB)	0.8177	0.0156	0.4798	0.0192	0.3163	0.0167	0.3187	0.0201
Weighted Majority Voting (LSVM, LR, RF, BNB)	0.8401	0.0131	0.4736	0.0203	0.3163	0.0178	0.3226	0.0168
Soft Voting (LSVM, LR, RF, BNB)	0.8174	0.0173	0.4784	0.0196	0.3161	0.0177	0.3266	0.0176

Tablo 5.3. Unigram + POS, unigram +NER ve unigram +CHUNK özniteliklerinin ayrı ayrı sınıflandırma algoritmalarına tabi tutulması ile elde edilen doğruluk sonuçları.

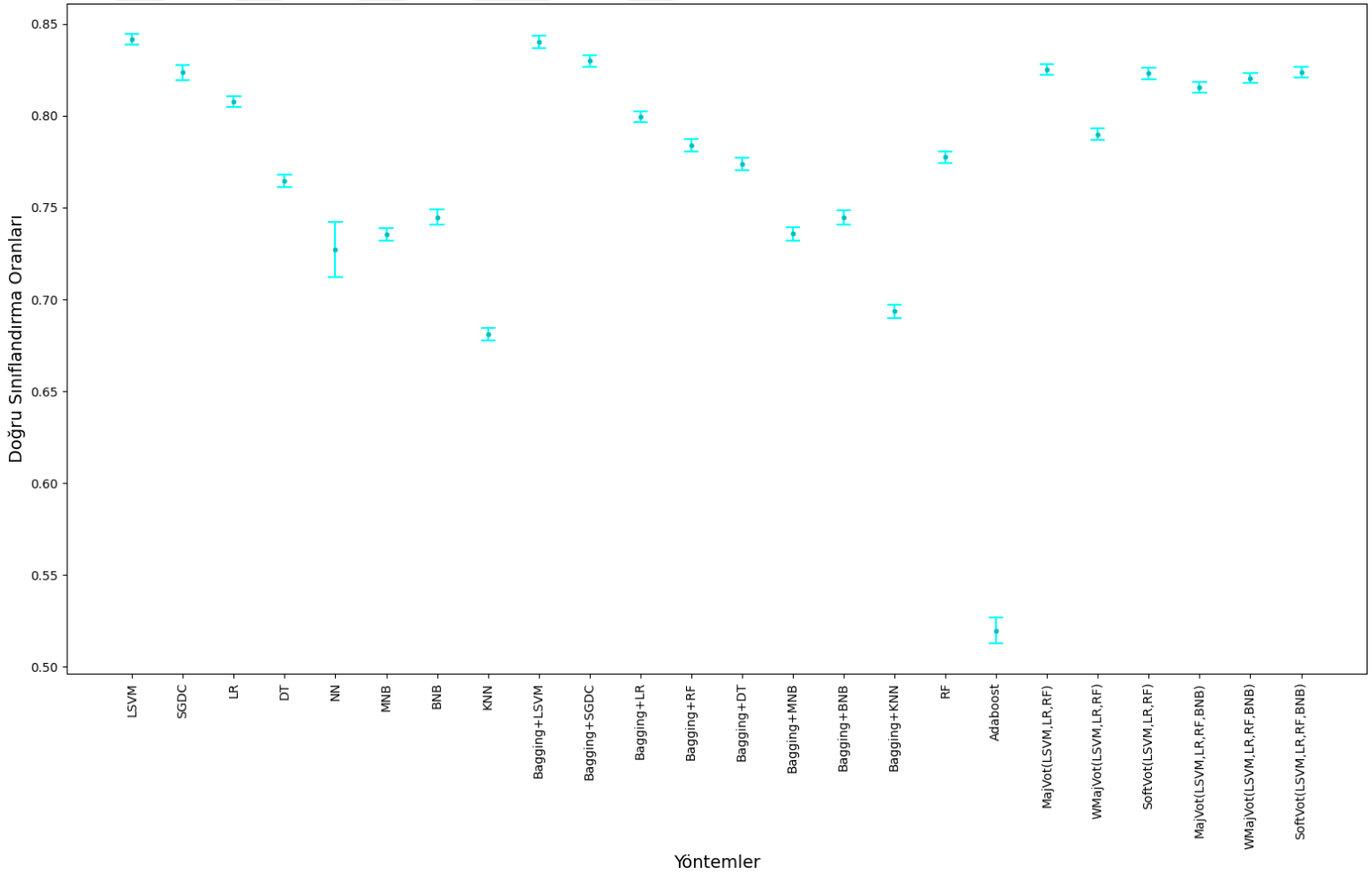
Sınıflandırıcı	**Unigram + POS**		**Unigram + NER**		**Unigram + CHUNK**	
	Ortalama	**Standart Sapma**	**Ortalama**	**Standart Sapma**	**Ortalama**	**Standart Sapma**
Linear SVM	**0.8465**	0.0142	**0.8387**	0.0127	**0.8415**	0.0152
SGDC	0.8327	0.0154	0.8321	0.0132	0.8235	0.0209
LR	0.7982	0.0170	0.7986	0.0165	0.8075	0.0152
DT	0.7376	0.0185	0.7672	0.0179	0.7646	0.0176
NN	0.7405	0.0707	0.7235	0.0632	0.7271	0.0767
Multinomial NB	0.7208	0.0187	0.7451	0.0205	0.7354	0.0173
Bernoulli NB	0.7304	0.0164	0.7402	0.0186	0.7447	0.0210
KNN	0.6661	0.0174	0.6316	0.0170	0.6809	0.0174
Bagging + LSVM	0.8464	0.0160	0.8376	0.0133	0.8402	0.0169
Bagging + SGDC	0.8351	0.0138	0.8320	0.0125	0.8298	0.0153
Bagging + LR	0.7885	0.0159	0.7905	0.0161	0.7995	0.0153
Bagging + RF	0.7830	0.0180	0.8017	0.0159	0.7841	0.0171
Bagging + DT	0.7609	0.0175	0.7727	0.0179	0.7737	0.0164
Bagging+ Multinomial NB	0.7198	0.0182	0.7429	0.0166	0.7358	0.0187
Bagging+ Bernoulli NB	0.7297	0.0200	0.7399	0.0176	0.7446	0.0203
Bagging+ KNN	0.6719	0.0205	0.6492	0.0195	0.6935	0.0176
RF	0.7594	0.0170	0.7920	0.0171	0.7774	0.0171
Adaboost	0.4835	0.0259	0.5178	0.0426	0.5198	0.0350
Majority Voting (LSVM, LR, RF)	0.8242	0.0135	0.8243	0.144	0.8251	0.0156
Weighted Majority Voting (LSVM, LR, RF)	0.8123	0.0161	0.8130	0.0161	0.7898	0.0158
Soft Voting (LSVM, LR, RF)	0.8388	0.0134	0.8277	0.0152	0.8231	0.0164
Majority Voting (LSVM, LR, RF, BNB)	0.8092	0.0157	0.8153	0.0131	0.8155	0.0151
Weighted Majority Voting (LSVM, LR, RF, BNB)	0.8296	0.0145	0.8248	0.0156	0.8205	0.0146
Soft Voting (LSVM, LR, RF, BNB)	0.7747	0.0171	0.8369	0.0162	0.8235	0.0147



Şekil 5.4. Unigram + POS özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.



Şekil 5.5. Unigram + NER özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.



Şekil 5.6. Unigram + CHUNK özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.

En iyi sonuçların unigram ile POS etiketlerinin birleştirilmesinden elde edilen öznitelik vektörü ile alınmasından dolayı NER ve CHUNK öznitelikleri ayrı ayrı bu öznitelik vektörü ile birleştirilerek sınıflandırma işlemi tekrarlanmıştır. Bu işlem sonucunda elde edilen deneysel bulgulara Tablo 5.4'te değinilmiştir.

Tablo 5.4. Unigram+POS+NER ve unigram+POS+CHUNK özniteliklerinin ayrı ayrı sınıflandırma algoritmalarına tabi tutulması ile elde edilen doğruluk sonuçları.

Sınıflandırıcı	Unigram + POS +NER		Unigram + POS + CHUNK	
	Ortalama	Standart Sapma	Ortalama	Standart Sapma
Linear SVM	0.8453	0.0142	**0.8457**	0.0142
SGDC	0.8282	0.0180	0.8162	0.0220
LR	0.7974	0.0161	0.7995	0.0160
DT	0.7349	0.0167	0.7341	0.0175
NN	0.7588	0.0424	0.7415	0.0612
Multinomial NB	0.7066	0.0174	0.7053	0.0210
Bernoulli NB	0.7294	0.0168	0.7287	0.0183
KNN	0.6367	0.0188	0.6672	0.0188
Bagging + LSVM	**0.8459**	0.0149	0.8455	0.0148
Bagging + SGDC	0.8317	0.0141	0.8273	0.0145
Bagging + LR	0.7888	0.0168	0.7904	0.0158
Bagging + RF	0.7822	0.0166	0.7804	0.0176
Bagging + DT	0.7604	0.0184	0.7614	0.0178
Bagging+ Multinomial NB	0.7061	0.0176	0.7046	0.0207
Bagging+ Bernoulli NB	0.7312	0.0206	0.7317	0.0168
Bagging+ KNN	0.6423	0.0191	0.6736	0.0186
RF	0.7549	0.0177	0.7482	0.0167
Adaboost	0.4868	0.0261	0.4863	0.0213
Majority Voting (LSVM, LR, RF)	0.8235	0.0155	0.8234	0.149
Weighted Majority Voting (LSVM, LR, RF)	0.8263	0.0142	0.8320	0.0134
Soft Voting (LSVM, LR, RF)	0.8257	0.0136	0.8173	0.0177
Majority Voting (LSVM, LR, RF, BNB)	0.8092	0.0179	0.8086	0.0166
Weighted Majority Voting (LSVM, LR, RF, BNB)	0.8113	0.0157	0.8089	0.0161
Soft Voting (LSVM, LR, RF, BNB)	0.8339	0.0147	0.8289	0.0141

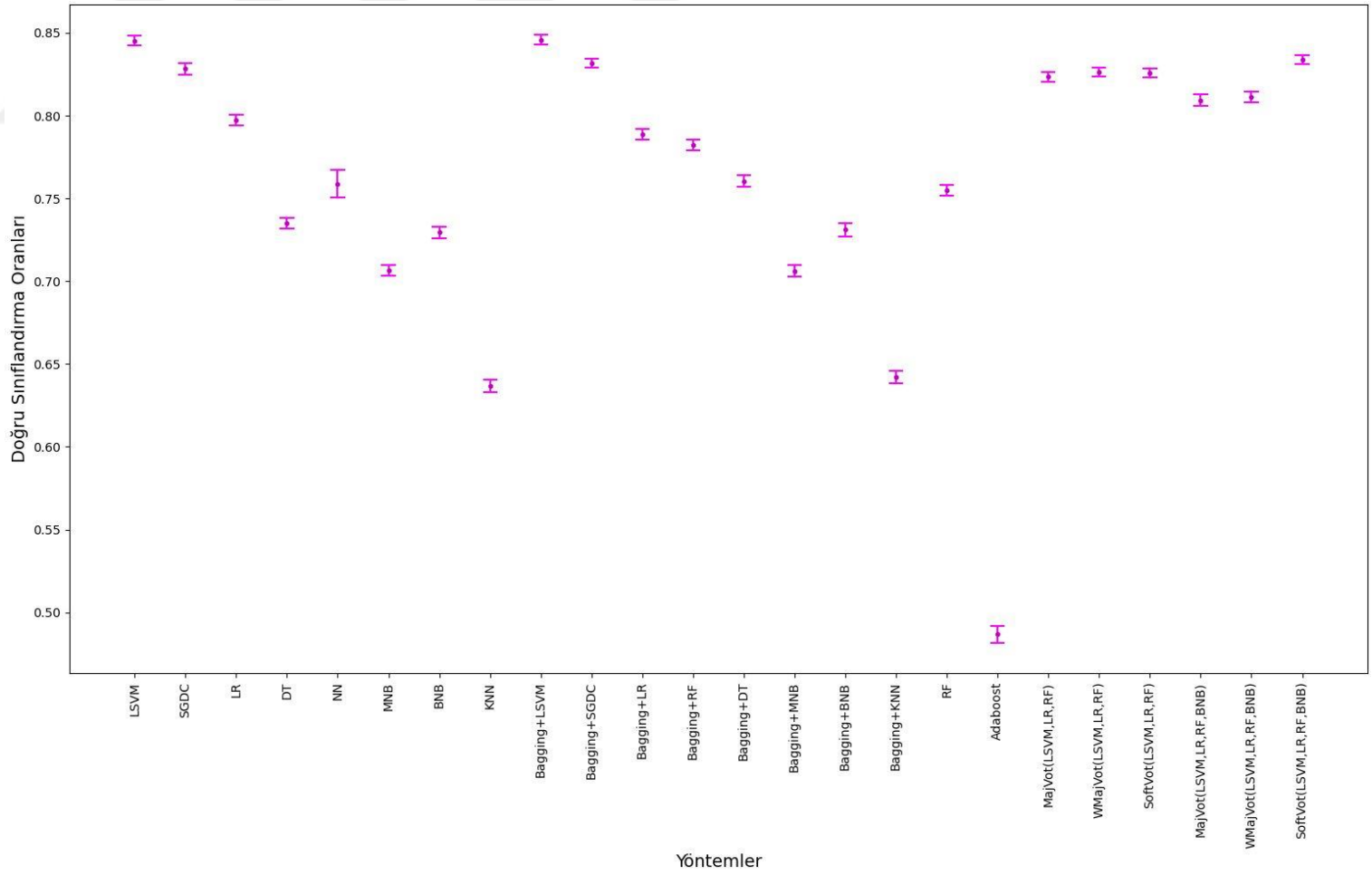
Tabloyu öznitelik vektörlerini baz alarak incelersek, unigram, POS etiketleri ve NER etiketçisinden elde edilen öznitelikler birleştirildiğinde, doğrusal çekirdekli SVM ile birlikte kullanılan Bagging algoritmasının sınıflandırma başarısının daha iyi olduğu gözlemlenmiştir. Soruları unigram, POS etiketleri ve CHUNK öznitelikleri ile birlikte ifade ettiğimizde ise doğrusal çekirdekli SVM algoritması en iyi performansı sergilemiştir. Ayrıca, doğrusal çekirdekli SVM ile birlikte kullanılan Bagging algoritması da doğrusal çekirdekli SVM algoritmasına oldukça yakın sonuçlar vermiştir. Bu tablodaki verileri genel olarak ele alırsak, en iyi

sonuçlar unigram, POS etiketleri ve NER özniteliklerinin birleştirilmesi ile oluşan öznitelik vektörünün doğrusal çekirdekli SVM ile kullanılan Bagging algoritması üzerinde işletilmesi ile elde edilmiştir.

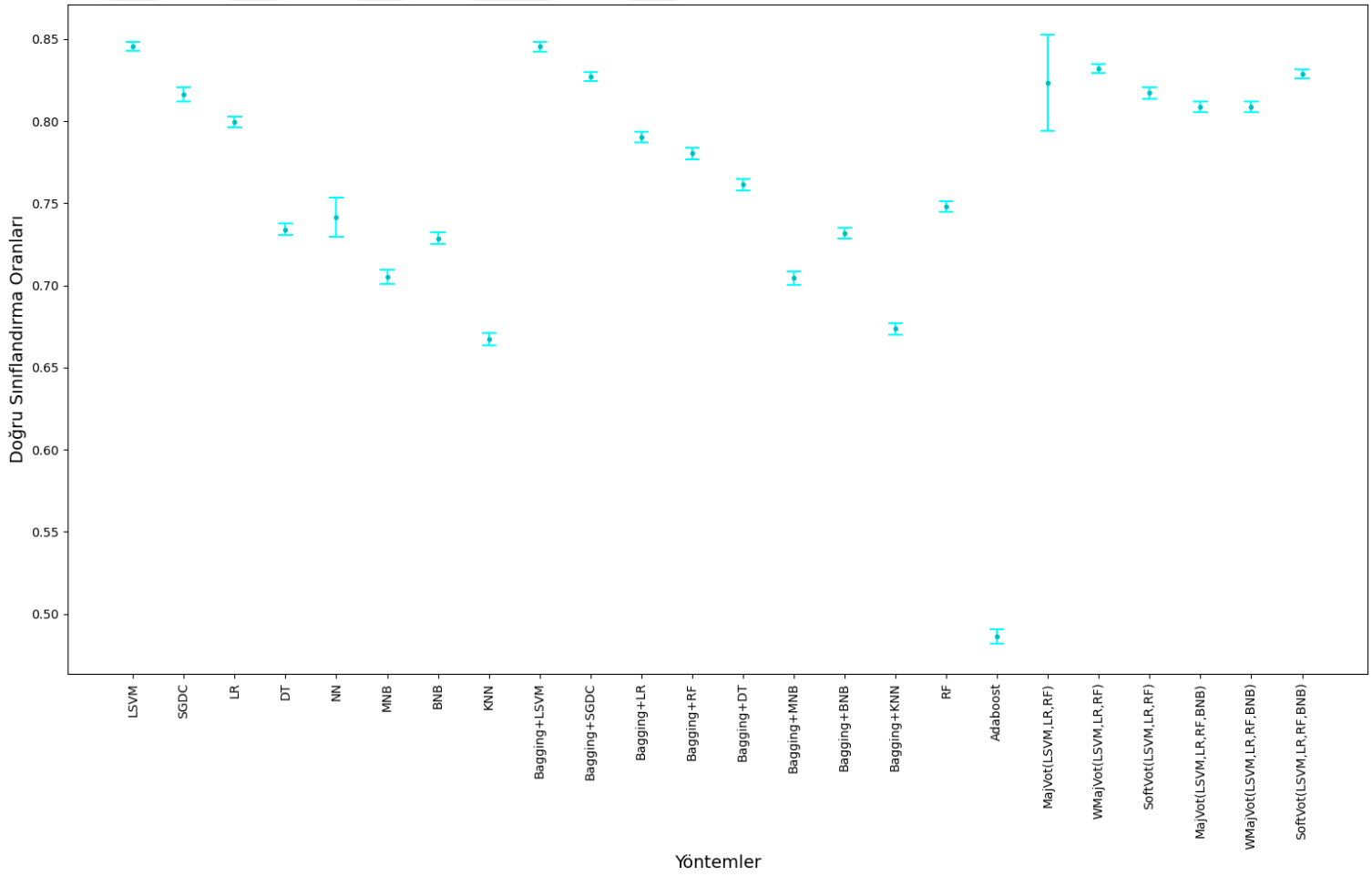
Şekil 5.7'de unigram + POS + NER öznitelikleri ve Şekil 5.8'de unigram + POS + CHUNK öznitelikleri üzerinde karşılaştırılan temel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemlerine ilişkin %95 güven aralığı için güven aralığı grafikleri sunulmaktadır. Şekil 5.7'de, doğrusal çekirdekli SVM sınıflandırıcısı ve Bagging algoritmasının doğrusal çekirdekli SVM ile oluşturduğu sınıflandırıcı topluluğunun diğer sınıflandırıcılara kıyasla istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği görülmektedir. Şekil 5.8'de, unigram + POS + NER özniteliklerinde olduğu gibi, doğrusal çekirdekli SVM sınıflandırıcısı ve Bagging algoritmasının doğrusal çekirdekli SVM ile oluşturduğu sınıflandırıcının diğer sınıflandırıcılara kıyasla istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği gözlemlenmektedir.

TREC veri seti incelendiğinde *"How many, How long, What city, What metal"* gibi iki kelimeden oluşan sorular olduğu gözlemlenmiştir. Bu nedenle, harf, hece veya kelime dizilimlerinde iki bitişik öğenin o dizilimdeki tekrar oranını bulmaya yardımcı olan bigram modeli uygulanmaya karar verilmiştir. Bigram modeli ile elde edilen öznitelikler, unigram+POS ve unigram+POS+NER öznitelik vektörlerine eklenmiş, ardından soru sınıflandırma işlemi gerçekleştirilmiştir. Sınıflandırma yöntemlerinin performanslarındaki artış oranının daha net bir şekilde görülebilmesi için bigram özniteliği eklenmeden önce elde edilen deney sonuçları ile bigram özniteliği eklendikten sonra elde edilen deney sonuçları Tablo 5.5'te karşılaştırılmıştır. Artış olan yöntemlerin sonuçları tabloda koyu renk ile işaretlenmiştir.

Tablo detaylı olarak incelendiğinde, öznitelik vektörüne bigram modelinden elde edilen öznitelikler de eklendiğinde, çoğu sınıflandırma algoritmasının performansında yaklaşık olarak %1'lik bir artış olduğu gözlemlenmiştir. Tablodan da anlaşılacağı üzere unigram +POS+NER öznitelik vektörüne bigram öznitelikleri eklendiğinde en yüksek doğruluk oranı doğrusal çekirdekli SVM ile elde edilmiştir.



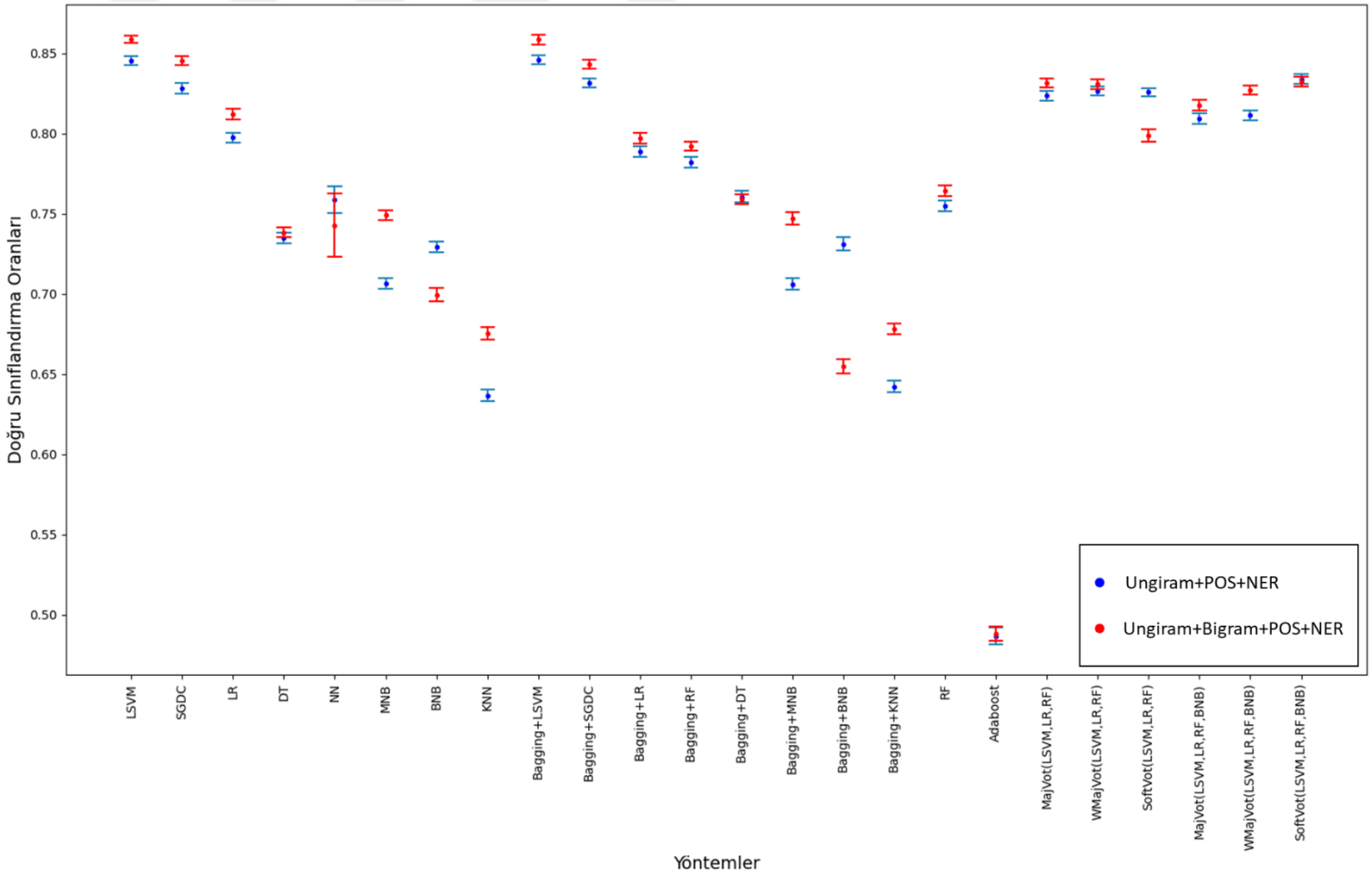
Şekil 5.7. Unigram + POS + NER özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.



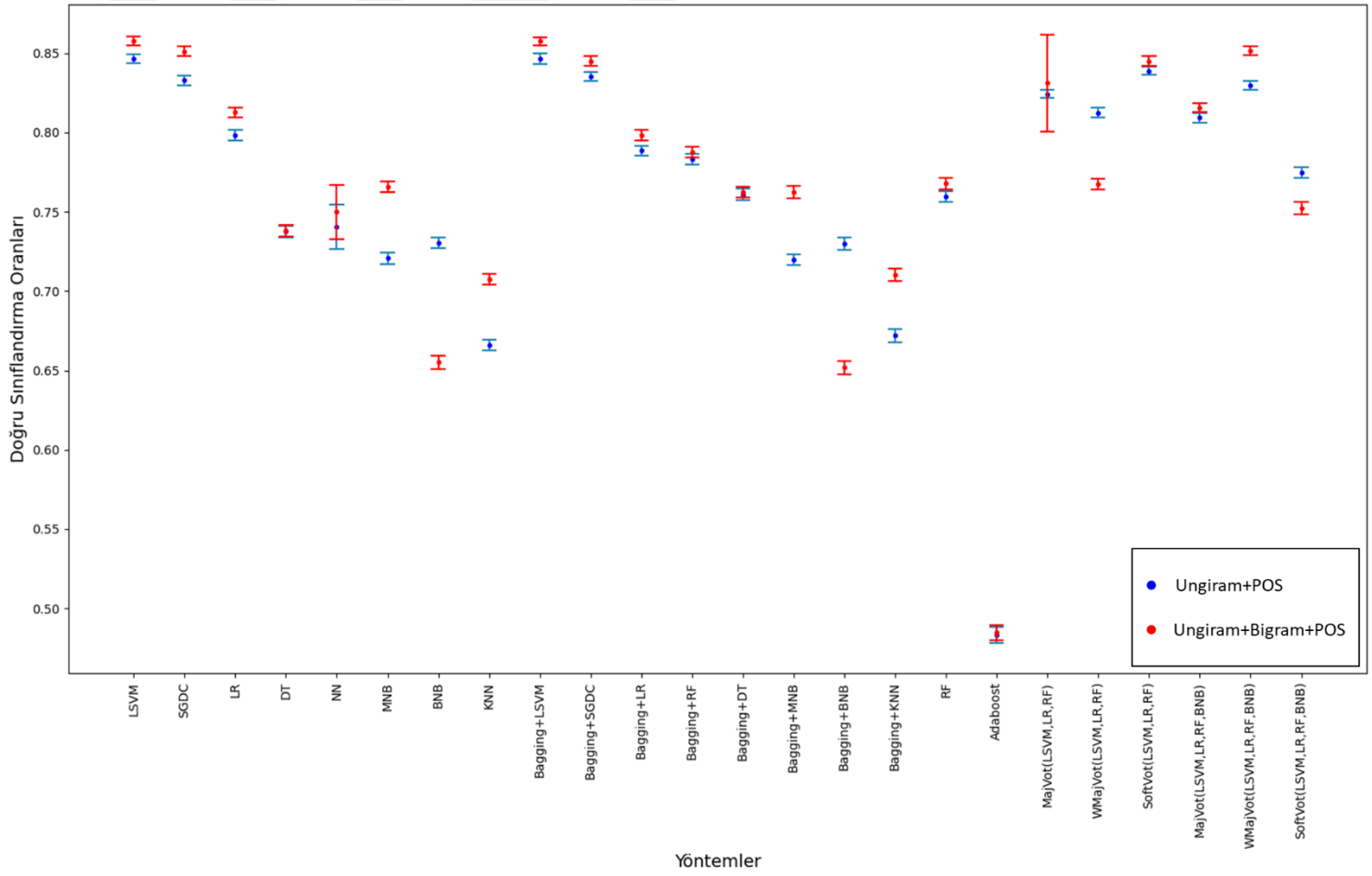
Şekil 5.8. Unigram + POS + CHUNK özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.

Tablo 5.5. Unigram+POS ve unigram+POS+NER özniteliklerinin bigram eklenmiş halleri ile karşılaştırılması.

Sınıflandırıcı	Unigram + POS +NER		Unigram + Bigram + NER + POS		Unigram + POS		Unigram + Bigram + POS	
	Ortalama	Standart Sapma	Ortalama	Standart Sapma	Ortalama	Standart Sapma	Ortalama	Standart Sapma
Linear SVM	0.8453	0.0142	**0.8587**	0.0117	0.8465	0.0142	**0.8577**	0.0143
SGDC	0.8282	0.0180	**0.8453**	0.0147	0.8327	0.0154	**0.8511**	0.0153
LR	0.7974	0.0161	**0.8121**	0.0170	0.7982	0.0170	**0.8126**	0.0149
DT	0.7349	0.0167	**0.7384**	0.0162	0.7376	0.0185	**0.7381**	0.0189
NN	0.7588	0.0424	0.7428	0.1002	0.7405	0.0707	**0.7498**	0.0868
Multinomial NB	0.7066	0.0174	**0.7490**	0.0156	0.7208	0.0187	**0.7658**	0.0170
Bernoulli NB	0.7294	0.0168	0.6995	0.0213	0.7304	0.0164	0.6552	0.0215
KNN	0.6367	0.0188	**0.6752**	0.0202	0.6661	0.0174	**0.7076**	0.0171
Bagging + LSVM	0.8459	0.0149	**0.8585**	0.0148	0.8464	0.0160	**0.8575**	0.0130
Bagging + SGDC	0.8317	0.0141	**0.8432**	0.0138	0.8351	0.0138	**0.8450**	0.0144
Bagging + LR	0.7888	0.0168	**0.7970**	0.0177	0.7885	0.0159	**0.7984**	0.0176
Bagging + RF	0.7822	0.0166	**0.7919**	0.0148	0.7830	0.0180	**0.7875**	0.0171
Bagging + DT	0.7604	0.0184	0.7589	0.0161	0.7609	0.0175	**0.7622**	0.0173
Bagging+ Multinomial NB	0.7061	0.0176	**0.7472**	0.0200	0.7198	0.0182	**0.7622**	0.0194
Bagging+ Bernoulli NB	0.7312	0.0206	0.6551	0.0227	0.7297	0.0200	0.6517	0.0223
Bagging+ KNN	0.6423	0.0191	**0.6783**	0.0169	0.6719	0.0205	**0.7101**	0.0200
RF	0.7549	0.0177	**0.7642**	0.0174	0.7594	0.0170	**0.7677**	0.0173
Adaboost	0.4868	0.0261	**0.4880**	0.0224	0.4835	0.0259	**0.4848**	0.0244
Majority Voting (LSVM, LR, RF)	0.8235	0.0155	**0.8314**	0.0139	0.8242	0.0135	**0.8311**	0.155
Weighted Majority Voting (LSVM, LR, RF)	0.8263	0.0142	**0.8307**	0.0149	0.8123	0.0161	0.7674	0.0183
Soft Voting (LSVM, LR, RF)	0.8257	0.0136	0.7987	0.0195	0.8388	0.0134	**0.8450**	0.0145
Majority Voting (LSVM, LR, RF, BNB)	0.8092	0.0179	**0.8175**	0.0175	0.8092	0.0157	**0.8155**	0.0141
Weighted Majority Voting (LSVM, LR, RF, BNB)	0.8113	0.0157	**0.8270**	0.0143	0.8296	0.0145	**0.8515**	0.0138
Soft Voting (LSVM, LR, RF, BNB)	0.8339	0.0147	0.8321	0.0155	0.7747	0.0171	0.7521	0.0196



Şekil 5.9. Unigram + POS + NER özniteliği ve unigram + bigram + POS + NER özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.



Şekil 5.10. Unigram + POS özniteliği ve unigram + bigram + POS özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.

Şekil 5.9'da unigram + bigram + POS + NER öznitelikleri ve unigram + POS + NER öznitelikleri ve Şekil 5.10'da unigram + bigram + POS öznitelikleri ve unigram + POS öznitelikleri üzerinde karşılaştırılan temel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemlerine ilişkin %95 güven aralığı için güven aralığı grafikleri sunulmaktadır. Şekil 5.9'da, temel öğrenme algoritmalarından NN ve BNB sınıflandırıcıları, Bagging algoritmasının DT ile ve BNB ile oluşturduğu sınıflandırıcılar ve Yumuşak Oylama yönteminin LSVM, LR, RF ile ve LSVM, LR, RF, BNB ile oluşturduğu sınıflandırıcı toplulukları hariç diğer tüm sınıflandırıcıların istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği görülmektedir. Şekil 5.10'da ise, temel öğrenme algoritmalarından BNB sınıflandırıcısı, Bagging algoritmasının BNB ile oluşturduğu sınıflandırıcı, Ağırlıklı Çoğunluk Oylama yönteminin LSVM, LR, RF ile oluşturduğu sınıflandırıcı topluluğu ve Yumuşak Oylama yönteminin LSVM, LR, RF, BNB ile oluşturduğu sınıflandırıcı topluluğu hariç diğer tüm sınıflandırıcıların istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği gözlemlenmektedir.

Deneysel çalışmalar sonucunda elde edilen soru sınıflandırma başarımının arttırılabilmesi için gereksiz özniteliklerin kaldırılarak metinin etkili bir şekilde temsil edilmesini sağlayan öznitelik seçimi işleminin uygulanmasına karar verilmiştir. Yapılan deneyler sonucunda en yüksek başarım, unigram+POS+NER öznitelik vektörüne bigram özniteliğinin eklenmesiyle elde edilmişti. Bu nedenle, öznitelik seçim işlemi bu öznitelik vektörü üzerinde gerçekleştirilmiştir. Öznitelik seçimi için ilk olarak ayrıntıları Bölüm 4.1.1' de verilen Ki-kare testi uygulanmıştır. Ki-kare testi sonucunda, en yüksek skora sahip olan öznitelikler seçilmiştir.

Deneylere ilk olarak, en yüksek skora sahip 500 öznitelik ile başlanmış, ardından 500'er arttırma yapılarak test edilmeye devam edilmiştir. En yüksek doğruluk değeri 8000 öznitelik ile elde edildiğinden öznitelik sayısı 29963'den 8000'e kadar düşürülmüştür. Ki-kare testinin ardından, Python programlama dilindeki Scikit-learn kütüphanesinde yer alan *SelectfromModel* metodu uygulanmıştır. Bu metot, özniteliklerin önemine göre öznitelik seçimi yapmamızı sağlar. Bu metoda özniteliklerin önemini belirlememizi sağlayan bir sınıflandırıcı ve bir *th* eşik değeri parametre olarak verilir. *th* eşik değerinin altında kalan özniteliklerin önemsiz olduğu düşünülerek, bu öznitelikler kaldırılır. Metotta

sınıflandırma algoritması olarak doğrusal çekirdekli SVM kullanılmıştır. Bu yöntem için yapılan deneysel çalışmalar sonucunda en iyi sonucu veren eşik değeri *th=1.15* olmuştur. Doğrusal çekirdekli SVM ve eşik değeri *th=1.15* kullanarak öznitelik seçimi yapıldığında, öznitelik sayısı 1467'ye kadar düşmüştür. Öznitelikler ve büyüklükleri ile ilgili bilgilere Tablo 5.6'da değinilmiştir.

Tablo 5.6. Öznitelikler ve büyüklükleri.

Öznitelik	Büyüklüğü
Unigram	8680
Bigram	21250
POS	30
NER	3
Unigram+Bigram+POS+NER	29963
Ki-kare ile indirgenmiş Unigram+Bigram+POS+NER	8000
Ki-kare ve LSVM ile indirgenmiş Unigram+Bigram+POS+NER	1467

1467 adet öznitelik ile temsil edilen sorular sınıflandırma işlemine tabii tutulduğunda, algoritmaların elde ettiği sınıflandırma başarımları Tablo 5.7'de sunulmuştur.

Sınıflandırma algoritmalarına bireysel olarak baktığımızda, en yüksek doğruluk, doğrusal çekirdekli SVM, SGDC, NN ve doğrusal çekirdekli SVM ile birlikte kullanılan Bagging algoritması tarafından alınmıştır. Bu nedenle, Çoğunluk Oylama, Ağırlıklı Çoğunluk Oylama ve Yumuşak Oylama yöntemleri bu sınıflandırma algoritmaları ile birlikte test edilmiştir. Tablo incelendiğinde, öznitelik seçimi yapılan veri seti üzerinde sınıflandırma algoritmalarının işletilmesi ile elde edilen en yüksek başarım doğrusal çekirdekli SVM algoritmasına aittir.

Şekil 5.11'de unigram + bigram + POS + NER öznitelikleri ve önerilen yöntem ile indirgenmiş unigram + bigram + POS + NER öznitelikleri üzerinde karşılaştırılan temel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemlerine ilişkin %95 güven aralığı için güven aralığı grafikleri sunulmaktadır. Şekilde, temel öğrenme algoritmalarından KNN sınıflandırıcısı hariç diğer sınıflandırıcılar istatistiksel olarak anlamlı ve daha yüksek doğrulukla sınıflandırma başarımı gösterdiği gözlemlenmektedir.

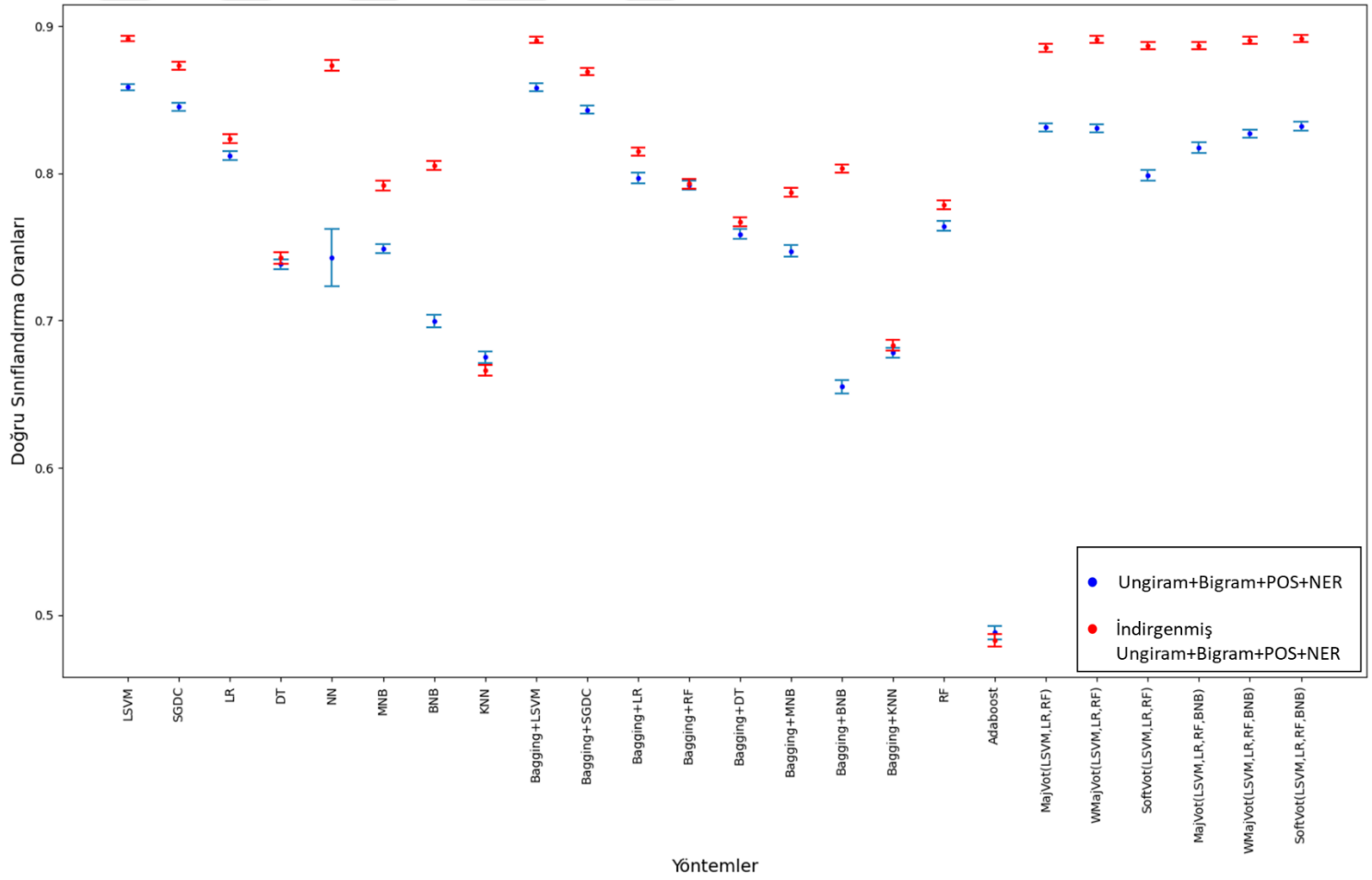
Tablo 5.7. Öznitelik seçimi yapılan veri seti üzerinde sınıflandırma algoritmalarının işletilmesi ile elde edilen doğruluk sonuçları.

Sınıflandırıcı	İndirgenmiş Unigram + Bigram + POS + NER	
	Ortalama	Standart Sapma
Linear SVM	**0.8918**	0.0099
SGDC	0.8732	0.0139
LR	0.8236	0.0143
DT	0.7427	0.0196
NN	0.8734	0.0178
Multinomial NB	0.7919	0.0163
Bernoulli NB	0.8054	0.0156
KNN	0.6661	0.0179
Bagging + LSVM	0.8908	0.0123
Bagging + SGDC	0.8692	0.0127
Bagging + LR	0.8149	0.0152
Bagging + RF	0.7931	0.0166
Bagging + DT	0.7672	0.0160
Bagging+ Multinomial NB	0.7873	0.0159
Bagging+ Bernoulli NB	0.8033	0.0154
Bagging+ KNN	0.6831	0.0191
RF	0.7785	0.0157
Adaboost	0.4829	0.0218
Majority Voting (LSVM, SGDC, Bagging-LSVM)	0.8854	0.0131
Weighted Majority Voting (LSVM, SGDC, Bagging-LSVM)	0.8912	0.0123
Soft Voting (LSVM, SGDC, Bagging-LSVM)	0.8868	0.0114
Majority Voting (LSVM, SGDC, Bagging-LSVM, NN)	0.8869	0.0134
Weighted Majority Voting (LSVM, SGDC, Bagging-LSVM, NN)	0.8905	0.0120
Soft Voting (LSVM, SGDC, Bagging-LSVM, NN)	0.8917	0.0122

Elde edilen tüm sonuçları genel olarak değerlendirecek olursak, yapılan bir çok testte en başarılı olan sınıflandırma algoritması doğrusal çekirdekli SVM algoritması olmuştur. Önerilen öznitelik seçimi yöntemi, doğrudan kullanılan özniteliklerden elde edilen sınıflandırma başarımlarına oranla sınıflandırma algoritmalarının performansında %5'e kadar bir artış sağlamıştır.

5.4.3 Sınıflandırma yöntemlerinin birleştirilmesi deneysel süreci ve sonuçları

Bu bölümde, önerilen birleştirme yöntemi için izlenilen deneysel süreç ve elde edilen deney sonuçları hakkında bilgi verilmektedir. Önerilen yöntemin sınanmasında, soru sınıflandırma için sıklıkla tercih edilen TREC veri seti kullanılmıştır. Yöntemlerin başarımlarının incelenmesinde, 10 kez takrarlanan10-kat çapraz doğrulama işlemi kullanılmıştır.



Şekil 5.11. Unigram + bigram + POS + NER özniteliği ve indirgenmiş unigram + bigram + POS + NER özniteliği için tüm yöntemlere ilişkin güven aralığı grafiği.

Birleştirme yönteminin uygulanması için ilk olarak, unigram ve POS öznitelikleri üzerinde bireysel olarak başarılı olan doğrusal çekirdekli SVM, SGDC ve LR algoritmaları (Bkz. Tablo 5.3) kullanılmıştır. Bu üç sınıflandırma algoritması ayrıntıları Bölüm 4.1.2'de verilen birleştirme yöntemi ile birleştirilmiş ve deney sonuçları alınmıştır. Önerilen yöntem ile SGDC ve LR sınıflandırma algoritmalarının bireysel performansından daha başarılı bir sonuç elde edilmiştir. Algoritmalardan elde edilen başarımın arttırabileceği düşüncesi ile aynı birleştirme yönteminden faydalanılarak, kullanılan tüm sınıflandırma algoritmaları (doğrusal çekirdekli SVM, SGDC, LR, LR, DT, Bernoulli NB, Multinominal NB, KNN ve NN) birleştirilmiş ve deneyler yeniden tekrarlanmıştır. Elde edilen deneysel sonuçlar Tablo 5.8'de sunulmuştur.

Tablo 5.8. Hibrit yöntem ile sınıflandırma algoritmalarının karşılaştırılması.

Sınıflandırıcı	Ortalama	Standart Sapma
LSVM	0.8463	0.0147
SGDC	0.8331	0.0154
LR	0.7981	0.0147
DT	0.7372	0.0196
BNB	0.7301	0.0173
MNNB	0.7215	0.0182
KNN	0.6641	0.0163
NN	0.7494	0.0516
Hibrit Yöntem (SVM+SGDC+LR)	0.8414	0.0206
Hibrit Yöntem (Tüm Sınıflandırıcılar)	0.8366	0.0213

Tablo incelendiğinde, önerilen birleştirme yöntemi ile elde edilen sonuçların SGDC, LR, DT, BNB, MNNB, KNN ve NN algoritmalarının bireysel performanslarından daha iyi bir sonuç verdiği gözlemlenmiştir.

5.5 Soru Yanıtlama Deneysel Sonuçları

Tezin bu bölümünde, soru yanıtlama problemi için uygulanan tahminleme tabanlı öznitelik temsiline, tahminleme tabanlı öznitelik temsili kullanılarak izlenilen deneysel sürece, geliştirilen soru yanıtlama mimarisinde kullanılan öznitelik temsiline, soru yanıtlama için izlenilen deneysel sürece ve kullanılan değerlendirme ölçütleri ile elde edilen deney sonuçlarına yer verilmiştir.

5.5.1 Tahminleme tabanlı öznitelik temsilinin uygulanması

Soru yanıtlamada soruların ve ilgili paragraf cümlelerinin temsili için Bölüm 3.1.2.1'de bahsedilen tahminleme tabanlı kelime gömülmelerinden olan Word2Vec ve FastText yöntemlerinden yararlanılmıştır. Kullanılan yöntemler kelime tabanlı olduğu için, ilk olarak kelimelere ait olan öznitelik vektörleri oluşturulmuştur. Cümlelere ait olan öznitelik vektörlerinin elde edilebilmesi için, ilgili cümleye ait kelimelerin vektörleri toplanmıştır ve bu toplam vektörünün ortalaması alınmıştır. Böylece, ilgili cümleye ait olan bir öznitelik vektörü elde edilmiştir. Bu işlem hem paragraf cümleleri hem de soru cümleleri için gerçekleştirilmiştir. Kullanılan kelime gösterim yöntemlerinde gömme boyutu literatürde de sıklıkla kullanılan değer olan 300 olarak alınmış ve farklı öznitelik vektörleri elde edilmiştir.

5.5.2 Tahminleme tabanlı öznitelik temsili ile soru yanıtlama deneysel süreci ve sonuçları

Bu bölümde, SQuAD veri seti üzerinde Word2Vec ve FastText kelime gömülmeleri için farklı benzerlik ölçütleri kullanılarak elde edilen başarımının değerlendirilmesinde izlenen deneysel süreç ve deneysel sonuçlar sunulmuştur.

Kullanılan öznitelik temsilleri ve benzerlik yöntemleri Python programlama dili ile geçekleştirilmiştir. Deney sonuçları, i7 2.80 GHz işlemcili, 16GB RAM'e ve Windows işletim sistemine sahip bilgisayar üzerinde çalıştırılarak elde edilmiştir.

Cevabı içerebilecek aday cümleler, paragraf cümleleri ve soru cümlesi arasında Overlap ve Dice katsayısı, Kosinüs ve Jaccard benzerliği, Öklid, Kulsinski ve Korelasyon uzaklığı, ve Jensen-Shannon uyuşmazlığı kullanılarak elde edilmeye çalışılmıştır.

5.5.2.1 Word2Vec deneysel süreci ve sonuçları

Bu bölümde, veri setinde yer alan her bir cümlenin Word2Vec gösterim yönteminin 300 gömme boyutu ile oluşturduğu öznitelikler üzerinde

gerçekleştirilen deneysel çalışmalar ve elde edilen deneysel sonuçlardan bahsedilmiştir.

İlk olarak, SQuAD veri setindeki hem soru hem de paragraf cümleleri için Word2Vec öznitelik vektörleri oluşturulmuştur. Overlap ve Dice katsayısı, Kosinüs ve Jaccard benzerliği, Öklid, Kulsinski ve Korelasyon uzaklığı, ve Jensen-Shannon uyuşmazlığı ölçütlerine göre, paragraftaki cümleler sıralanmıştır. İlk sırada yer alan cümle, soru ile en benzer olan cümle olduğu için aday cevap cümlesi olarak seçilmiştir. Bulunan aday cümle ile gerçek cevabın geçtiği cümle karşılaştırılarak kullanılan benzerlik ölçütünün doğruluk değeri hesaplanmıştır. Elde edilen sonuçlara Tablo 5.9'da yer verilmiştir.

Tablo 5.9. Word2Vec (300) öznitelik vektörü üzerinde farklı benzerlik ölçütlerinin kullanılması ile elde edilen deneysel sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap Katsayısı	**0.7126**
Dice Katsayısı	0.6841
Jaccard Benzerliği	0.6841
Kulsinski Uzaklığı	0.6504
Kosinüs Benzerliği	0.3780
Korelasyon Uzaklığı	0.3637
Jensen-Shannon Uyuşmazlığı	0.3256
Öklid Uzaklığı	0.1948

Tablo incelendiğinde, Word2Vec kelime gömülmesi ile temsil edilen SQuAD veri seti üzerinde cevabın geçtiği cümleyi, en doğru şekilde bulan benzerlik ölçütü Overlap katsayısıdır. İkinci olarak, Dice katsayısı ve Jaccard benzerliği, diğer benzerlik ölçütlerine oranla daha iyi sonuçlar vermiştir. Benzerlik ölçütlerinin birbirlerinin açıklarını kapatması yani bazı benzerlik ölçütlerinin yanlış bulduğu aday cümleleri, diğer benzerlik ölçütlerinin doğru bulabileceği düşüncesi ile ölçütler birlikte kullanılmıştır. Word2Vec öznitelik vektörü ile temsil edilen cümleler üzerinde en iyi sonucu veren üç benzerlik ölçütünün diğer benzerlik ölçütleri ile birleştirilmesi ile elde edilen deneysel sonuçlar Tablo 5.10'da sunulmuştur.

Tablo 5.10. Word2Vec (300) öznitelik vektörü üzerinde iki benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Kosinüs	**<u>0.7735</u>**
Overlap + Jensen-Shannon	0.7721
Overlap + Korelasyon	0.7719
Overlap + Öklid	0.7691
Overlap + Kulsinski	0.7663
Overlap + Dice	0.7628
Dice + Jensen-Shannon	**0.7588**
Dice + Kosinüs	0.7564
Dice + Korelasyon	0.7532
Dice + Öklid	0.7247
Dice + Kulsinski	0.6928
Dice + Jaccard	0.6841
Kulsinski + Jensen-Shannon	**0.7369**
Kulsinski + Kosinüs	0.7345
Kulsinski + Korelasyon	0.7306
Kulsinski + Öklid	0.6876

Tabloya bakıldığında, Dice katsayısı ve Jaccard benzerlik yöntemi birleştirildiğinde elde edilen başarım ile bu iki ölçütün bireysel başarımları arasında herhangi bir fark bulunmadığı gözlemlenmiştir. Bu nedenle, Jaccard benzerlik ölçütü diğer ölçütlerle birleştirilmemiştir. Ayrıca, bireysel başarımı yüksek olan ölçütlerden düşük olan ölçütlere doğru bir birleştirme işlemi yapıldığında, birleşimin performansının düştüğü Tablo 5.10'da görülmektedir. Bu nedenle diğer benzerlik ölçütlerinin ikili kombinasyonları test edilmemiştir.

Tablodan da gözlemlendiği üzere, en yüksek başarımı Overlap katsayısı ile Kosinüs benzerliğinin birleştirilmesi ve Dice katsayısı ile Jensen Shannon uyuşmazlığının birleştirilmesi vermiştir. Bundan dolayı diğer denemeler, bu iki kombinasyon üzerinden yapılmıştır. Word2Vec öznitelik vektörü ile temsil edilen soru ve paragraf cümleleri arasındaki benzerliğin bulunabilmesi için benzerlik ölçütlerinin üçlü kombinasyonlarından elde edilen sonuçlara Tablo 5.11'de yer verilmiştir.

Tablo 5.11. Word2Vec (300) öznitelik vektörü üzerinde üç benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Kosinüs + Öklid	**0.8125**
Overlap + Kosinüs + Kulsinski	0.8116
Overlap + Kosinüs + Dice	0.8091
Overlap + Kosinüs + Jensen Shannon	0.8012
Overlap + Kosinüs + Korelasyon	0.7948
Dice + Jensen Shannon + Overlap	**0.8105**
Dice + Jensen Shannon + Korelasyon	0.7932
Dice + Jensen Shannon + Kosinüs	0.7902
Dice + Jensen Shannon + Öklid	0.7873
Dice + Jensen Shannon + Kulsinski	0.7655

Tablo 5.11 incelendiğinde, en iyi performansın Overlap katsayısı, Kosinüs benzerliği ile Öklid uzaklığının birleştirilmesinden elde edildiği gözlenmiştir. Dice katsayısı ve Jensen Shannon uyuşmazlığının, diğer ölçütler ile birleştirilmesinden elde edilen sonuçlarda bireysel başarımlarına göre %6'lık bir artış olsa da Overlap katsayısı, Kosinüs benzerliği ve Öklid uzaklığının bir araya getirilmesi ile elde edilen sonuçlar kadar başarılı olunamamıştır. Bu nedenle, diğer kombinasyonlara bu ölçütler üzerinden devam edilmiştir. Ölçütlerin kombinasyonlarından elde edilen sonuçlar Tablo 5.12'de sunulmuştur.

Tablo 5.12. Word2Vec (300) öznitelik vektörü üzerinde dört ve üzeri benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Kosinüs + Öklid + Kulsinski	**0.8362**
Overlap + Kosinüs + Öklid + Dice	0.8361
Overlap + Kosinüs + Öklid + Jensen Shannon	0.8347
Overlap + Kosinüs + Öklid + Korelasyon	0.8287
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon	**0.8560**
Overlap + Kosinüs + Öklid + Dice + Jensen Shannon	0.8559
Overlap + Kosinüs + Öklid + Kulsinski + Korelasyon	0.8497
Overlap + Kosinüs + Öklid + Kulsinski + Dice	0.8392
Overlap + Kosinüs + Öklid + Dice + Korelasyon	0.8306
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Korelasyon	**0.8662**
Overlap + Kosinüs + Öklid + Dice + Jensen Shannon + Korelasyon	0.8661
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Dice	0.8585

Tablo 5.12'deki dörtlü kombinasyonlara bakıldığında, Overlap katsayısı, Kosinüs benzerliği, Öklid uzaklığı ile Kulsinski uzaklığının birleştirilmesi ve Dice katsayısının birleştirilmesinden elde edilen sonuçlar birbirine çok yakın çıkmıştır.

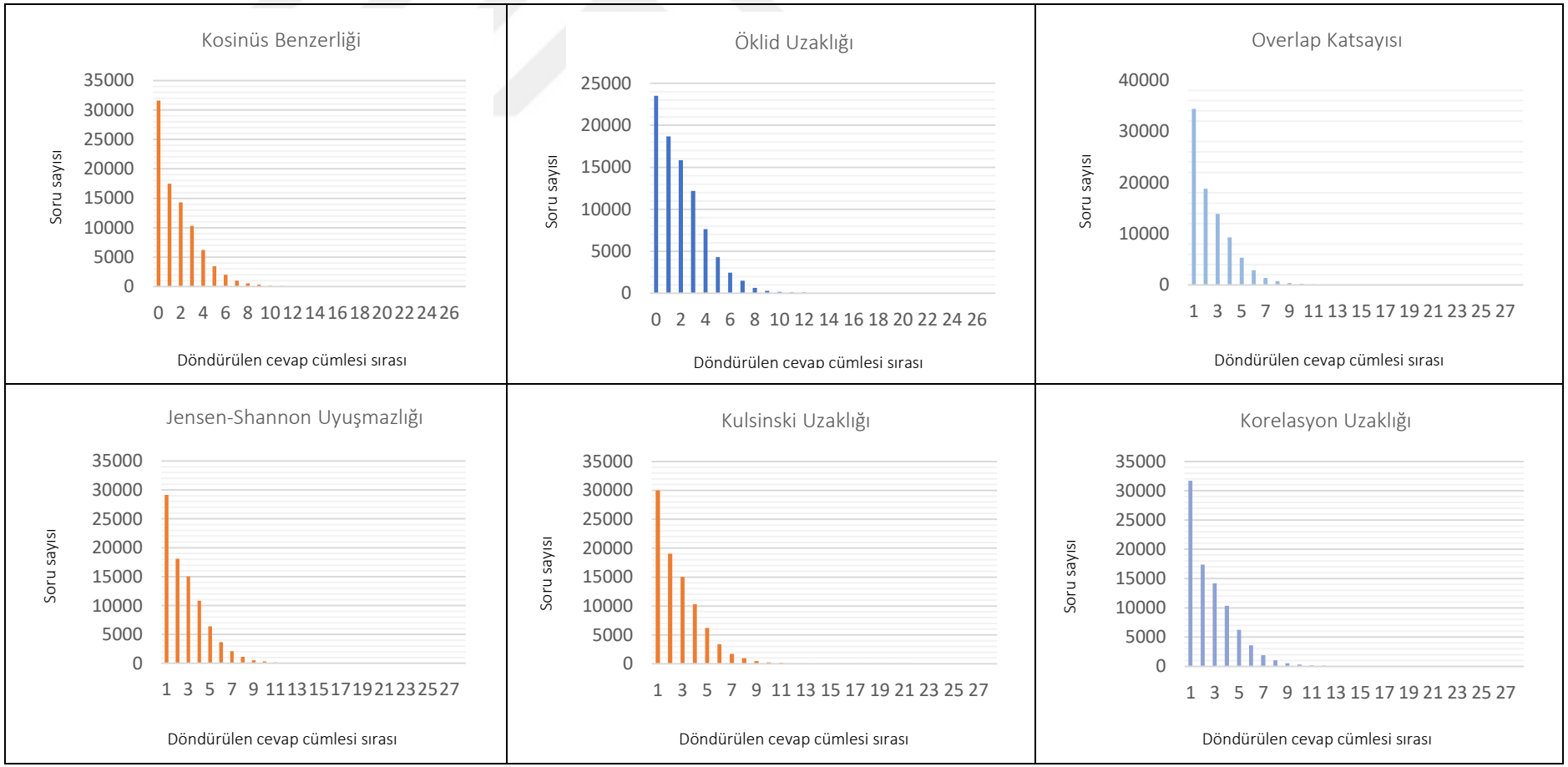
Bu nedenle her iki kombinasyona da beşinci ve altıncı ölçütler eklenerek deneyler tekrar gerçekleştirilmiştir.

Sonuçlar genel olarak incelendiğinde, benzerlik ölçütlerinin 0.71'lerde olan bireysel başarımının, ölçütlerin birleştirilmesi ile 0.15 artarak 0.86'lara kadar yükseldiği gözler önüne serilmiştir.

Overlap, Kosinüs, Öklid, Kulsinski, Jensen Shannon ve Korelasyon ölçütlerinden elde edilen aday cümlelerin cümle id'leri, çok katmanlı yapay sinir ağına verilmiştir. Veri seti, %70 eğitim seti, %30 test seti olarak ayrılmıştır. Word2Vec ile oluşturulan öznitelikler, 100 nörondan oluşan 10 gizli katmana sahip bir yapay sinir ağında test edildiğinde, doğruluk değeri 0.69 olarak bulunmuştur. Gizli katman sayısı artırılarak yine 100 nörondan oluşan 15 gizli katmana sahip bir yapay sinir ağında test edildiğinde, doğruluk değeri 0.71 olarak elde edilmiştir.

Buradan elde edilen başarımın artırılması için farklı yöntemler de denenmiştir. Her bir soru cümlesi için benzerlik ölçütü tarafından bulunan aday cevap cümlelerinin dağılımları hesaplanmıştır. İlgili benzerlik ölçütünün doğru cevap cümlesini kaçıncı sırada bulunduğunu gösteren dağılım grafikleri Şekil 5.12'de gösterilmiştir.

Dağılım grafiklerinin, koordinat sisteminde y-eksenine yakın olması, aday cevap cümlelerinin ilk cümlelerde yığılı olduğunu göstermektedir. Buradan hareketle, bulunan aday cümleler içerisinden ilk iki cümle ve ilk üç cümle, aday cümle olarak döndürülmüş ve deneyler yinelenmiştir. Elde edilen deneysel sonuçlar Tablo 5.13'te sunulmuştur.



Şekil 5.12. Word2Vec ile temsil edilen aday cevap cümlelerinin benzerlik ölçütlerine göre dağılımları.

Tablo 5.13, iki veya üç aday cevap cümlesi olduğunda elde edilen doğruluk değerinin, tek aday cevap cümlesi olduğunda elde edilen doğruluk değerinden daha yüksek olduğunu göstermektedir. Tek aday cevap olduğunda elde edilen doğruluk, iki aday cümle kullanıldığında yaklaşık olarak 0.14, üç aday cümle kullanıldığında ise yaklaşık olarak 0.20 artmıştır.

Tablo 5.13. Word2Vec ile temsil edilen birden fazla aday cümle kullanılması ile elde edilen deneysel sonuçlar.

Benzerlik Ölçütü	1 aday cümle için Doğruluk	2 aday cümle için Doğruluk	3 aday cümle için Doğruluk
Overlap Katsayısı	**0.7126**	**0.8515**	**0.9174**
Dice Katsayısı	0.6841	0.8383	0.9131
Jaccard Benzerliği	0.6841	0.8383	0.9131
Kulsinski Uzaklığı	0.6504	0.8237	0.9070
Kosinüs Benzerliği	0.3780	0.6204	0.7821
Korelasyon Uzaklığı	0.3637	0.6069	0.7759
Jensen-Shannon Uyuşmazlığı	0.3256	0.5636	0.7388
Öklid Uzaklığı	0.1948	0.4048	0.6059

Tablo incelendiğinde, alınan diğer sonuçlarda olduğu gibi, yine en iyi sonucu veren benzerlik ölçütü Overlap katsayısıdır. Burada kullanılan benzerlik ölçütleri, önceki sonuçlara benzer sonuçlar sergilediği için iki ve üç aday cevap cümlesi için Overlap, Kosinüs, Öklid, Kulsinski, Jensen-Shannon ve Korelasyon ölçütleri birleştirilmiş ve deneyler gerçeklenmiştir. Bu işlem ile elde edilen deney sonucuna Tablo 5.14'te yer verilmiştir.

Tablo 5.14. Word2Vec ile temsil edilen birden fazla aday cümle için benzerlik ölçütlerinin birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	1 aday cümle için Doğruluk	2 aday cümle için Doğruluk	3 aday cümle için Doğruluk
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Korelasyon	0.8662	0.9585	**0.9839**

Tablo 5.14'ten de anlaşıldığı üzere, birden fazla aday cevap üzerinde benzerlik ölçütleri birlikte kullanıldığında doğruluk oranı artmaktadır.

5.5.2.2 FastText deneysel süreci ve sonuçları

Bu bölümde, veri setindeki paragraf cümleleri ve soru cümleleri için FastText gösterim yönteminin 300 gömme boyutu ile oluşturduğu öznitelikler üzerinde gerçekleştirilen deneysel çalışmalar ve elde edilen deneysel sonuçlara yer verilmiştir.

İlk olarak, SQuAD veri setinde yer alan hem soru hem de paragraf cümleleri için FastText öznitelik vektörleri elde edilmiştir. Overlap ve Dice katsayısı, Kosinüs ve Jaccard benzerliği, Öklid, Kulsinski ve Korelasyon uzaklığı, ve Jensen-Shannon uyuşmazlığı ölçütlerine göre, paragraftaki cümleler sıralanmıştır. Soru ile en benzer olan cümle olan cümle cevap cümlesi olarak seçilmiştir. Seçilen cevap cümlesi ile gerçek cevabın geçtiği cümle karşılaştırılarak ilgili benzerlik ölçütünün başarımı bulunmuştur. Benzerlik ölçütlerinin başarımları Tablo 5.15'te sunulmuştur.

Tablo 5.15. FastText (300) öznitelik vektörü üzerinde farklı benzerlik ölçütlerinin kullanılması ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap Katsayısı	**0.7126**
Jaccard Benzerliği	0.6841
Kulsinski Uzaklığı	0.6504
Dice Katsayısı	0.6459
Öklid Uzaklığı	0.2837
Jensen-Shannon Uyuşmazlığı	0.2609
Kosinüs Benzerliği	0.2599
Korelasyon Uzaklığı	0.2577

Tablodan görüldüğü üzere, cevabın geçtiği cümleyi, en doğru şekilde bulan benzerlik ölçütü Overlap katsayısıdır. İkinci olarak, Jaccard benzerliği diğer benzerlik ölçütlerine oranla daha iyi sonuçlar vermiştir. Burada da Word2Vec kelime gömülmesi üzerinde gerçekleştirilen deneyler tekrarlanmıştır. Diğer bir deyişle, benzerlik ölçütleri birlikte kullanılarak, benzerlik ölçütlerinin dezavantajlı olduğu yönleri kapatılmaya çalışılmıştır. İki benzerlik ölçütünün birleştirilmesi ile elde edilen deneysel sonuçlar Tablo 5.16'da görülmektedir.

Tablo incelendiğinde, bireysel başarımı yüksek olan ölçütlerden düşük olan ölçütlere doğru bir birleştirme işlemi yapıldığında, birleşimin performansının düştüğü görülmektedir. Bu nedenle diğer benzerlik ölçütlerinin ikili kombinasyonları test edilmemiştir.

Tablo 5.16. FastText (300) öznitelik vektörü üzerinde iki benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Öklid	**0.7748**
Overlap + Kosinüs	0.7726
Overlap + Korelasyon	0.7726
Overlap + Jensen Shannon	0.7717
Overlap + Kulsinski	0.7682
Overlap + Jaccard	0.7628
Jaccard + Öklid	**0.7714**
Jaccard + Kosinüs	0.7566
Jaccard + Korelasyon	0.7566
Jaccard + Jensen Shannon	0.7547
Jaccard + Dice	0.7409
Jaccard + Kulsinski	0.6928
Kulsinski + Öklid	**0.7510**
Kulsinski + Dice	0.7350
Kulsinski + Kosinüs	0.7323
Kulsinski + Korelasyon	0.7323
Kulsinski + Jensen Shannon	0.7297

Tablodan da gözlemlendiği üzere, en yüksek başarımı Overlap katsayısı ile Öklid uzaklığının birleştirilmesi ve Jaccard benzerliği ile Öklid uzaklığının birleştirilmesi ile elde edilmiştir. Bu sebeple diğer denemeler, bu benzerlik ölçütlerinin kombinasyonu üzerinden yapılmıştır. Elde edilen deneysel sonuçlara Tablo 5.17'de yer verilmiştir.

Tablo 5.17 incelendiğinde, en iyi performansın Overlap katsayısı, Öklid uzaklığı ile Kulsinski uzaklığının birleştirilmesinden elde edildiği gözlenmiştir. Jaccard benzerliği ve Öklid uzaklığının diğer benzerlik ölçütleri ile birleştirilmesinden elde edilen sonuçlar, bireysel başarımlarına göre Jaccard benzerliği yaklaşık olarak 0.13, Öklid uzaklığı yaklaşık 0.53 artmıştır. Her ne kadar böyle bir artış olsa da Overlap katsayısı, Öklid uzaklığı ile Kulsinski uzaklığının bir araya getirilmesi ile elde edilen sonuçlar kadar başarılı olunamamıştır. Bu nedenle, diğer kombinasyonlara Overlap katsayısı, Öklid uzaklığı ile Kulsinski uzaklığı

üzerinden devam edilmiştir. Benzerlik ölçütlerinin kombinasyonlarından elde edilen sonuçlara Tablo 5.18'de yer verilmiştir.

Tablo 5.17. FastText (300) öznitelik vektörü üzerinde üç benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Öklid + Kulsinski	**0.8212**
Overlap + Öklid + Dice	0.8210
Overlap + Öklid + Jaccard	0.8182
Overlap + Öklid + Jensen-Shannon	0.8150
Overlap + Öklid + Korelasyon	0.8105
Overlap + Öklid + Kosinüs	0.8104
Jaccard + Öklid + Jensen-Shannon	**0.8133**
Jaccard + Öklid + Dice	0.8186
Jaccard + Öklid + Korelasyon	0.8083
Jaccard + Öklid + Kosinüs	0.8083
Jaccard + Öklid + Kulsinski	0.7786

Tablo 5.18. FastText (300) öznitelik vektörü üzerinde dört ve üzeri benzerlik ölçütünün birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	Doğruluk
Overlap + Kosinüs + Öklid + Kulsinski	**0.8494**
Overlap + Kosinüs + Öklid + Dice	0.8486
Overlap + Kosinüs + Öklid + Jaccard	0.8468
Overlap + Kosinüs + Öklid + Jensen Shannon	0.8378
Overlap + Kosinüs + Öklid + Korelasyon	0.8287
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon	**0.8708**
Overlap + Kosinüs + Öklid + Kulsinski + Dice	0.8670
Overlap + Kosinüs + Öklid + Kulsinski + Jaccard	0.8523
Overlap + Kosinüs + Öklid + Kulsinski + Korelasyon	0.8497
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Dice	**0.8854**
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Korelasyon	0.8733
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Jaccard	0.8709

Tablo 5.18'deki dörtlü kombinasyonlara bakıldığında; Overlap katsayısı, Kosinüs benzerliği, Öklid uzaklığı ile Kulsinski uzaklığının birleştirilmesinden elde edilen sonuçlar, diğer kombinasyonlara göre daha yüksektir. Bu nedenle, bu birleşime beşinci ve altıncı ölçütler eklenerek deneyler tekrar gerçekleştirilmiştir.

Sonuçlar genel olarak incelendiğinde, benzerlik ölçütlerinin 0.71'lerde olan bireysel başarımının, ölçütlerin birleştirilmesi ile 0.17 artarak 0.88'lere kadar yükseldiği gözler önüne serilmiştir.

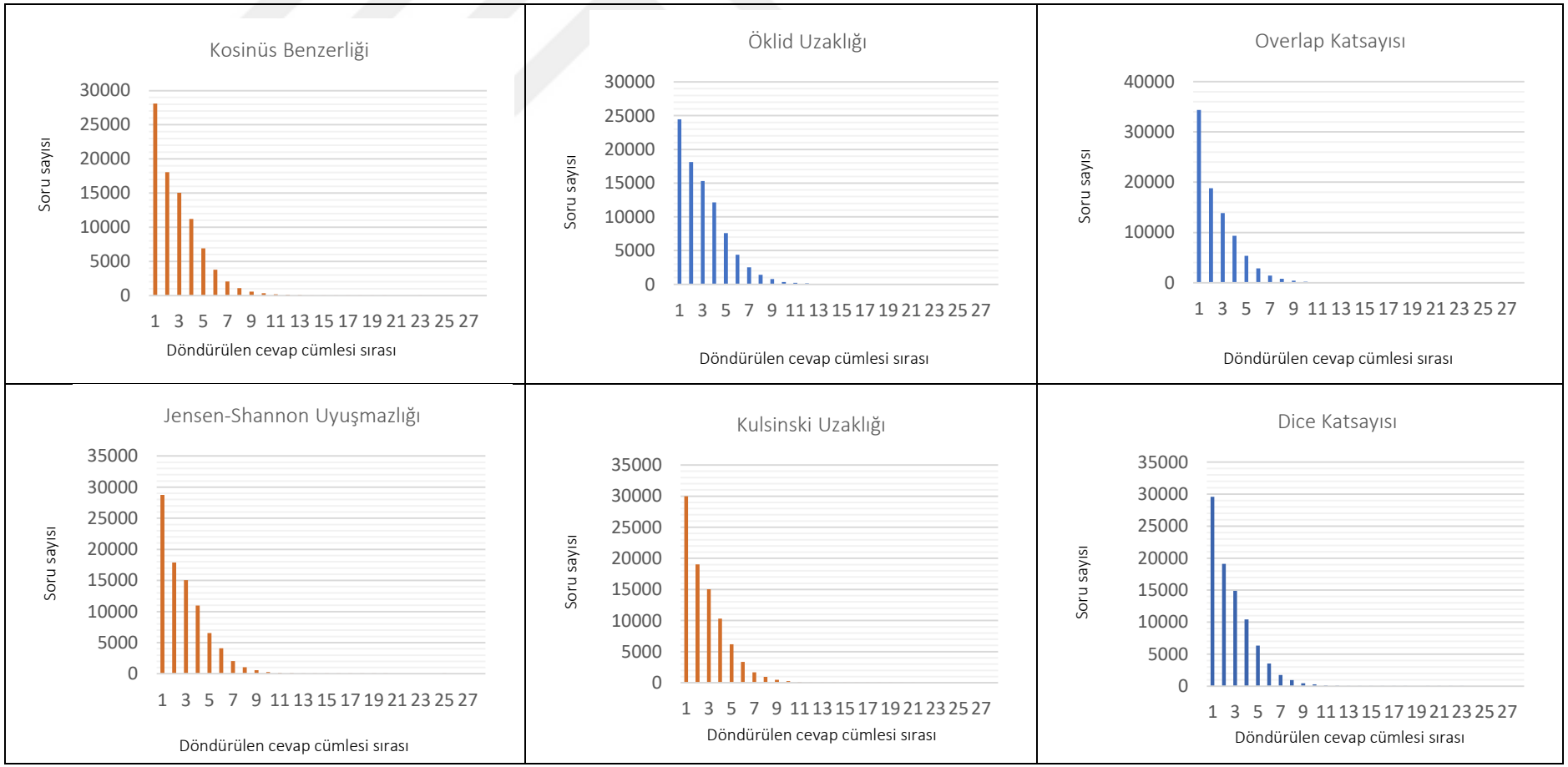
Word2Vec kelime gömülmesinde olduğu gibi; Overlap, Kosinüs, Öklid, Kulsinski, Jensen-Shannon ve Dice ölçütlerinden elde edilen aday cümlelerin cümle id'leri, çok katmanlı yapay sinir ağına verilmiştir. Veri seti %70 eğitim, %30 test seti olarak ayrılmıştır. FastText ile elde edilen öznitelikler, 100 nörondan oluşan 10 gizli katmana sahip bir yapay sinir ağında test edildiğinde, doğruluk değeri 0.69 olarak elde edilmiştir. Gizli katman sayısı artırılarak 100 nörondan oluşan 15 gizli katmana sahip bir yapay sinir ağında test edildiğinde, doğruluk değeri 0.70 olarak bulunmuştur.

Doğruluk değerinin arttırılması için, FastText gösterimi ile 300 gömme boyutu kullanılarak ifade edilen cümleler üzerinde, Bölüm 5.5.2.1'de bahsedilen, her bir soru cümlesi için benzerlik ölçütü tarafından bulunan aday cevap cümlelerinin dağılımlarından faydalanılmıştır. İlgili benzerlik ölçütünün doğru cevap cümlesini kaçıncı sırada bulunduğunu gösteren dağılım grafikleri Şekil 5.13'te gösterilmiştir.

Word2Vec gösterim yöntemi kullanıldığında elde edilen dağılım grafikleri gibi bu grafikler de koordinat sisteminde y-eksenine yakın çıkmıştır. Bu da, aday cevap cümlelerinin ilk cümlelerde yığılı olduğunu göstermektedir. Buradan hareketle, bulunan aday cümleler içerisinden ilk iki cümle ve ilk üç cümle, aday cevap cümlesi olarak döndürülmüş ve sonuçlar alınmıştır. Elde edilen deneysel sonuçlara Tablo 5.19'da yer verilmiştir.

Tablo 5.19. FastText ile temsil edilen cümleler için birden fazla aday cümle kullanılması ile elde edilen sonuçlar

Benzerlik Ölçütü	1 aday cümle için Doğruluk	2 aday cümle için Doğruluk	3 aday cümle için Doğruluk
Overlap Katsayısı	**0.7126**	**0.8515**	**0.9174**
Dice Katsayısı	0.6841	0.8178	0.9020
Jaccard Benzerliği	0.6504	0.8383	0.9131
Kulsinski Uzaklığı	0.6459	0.8236	0.9070
Kosinüs Benzerliği	0.2837	0.4839	0.6751
Korelasyon Uzaklığı	0.2609	0.4839	0.6751
Jensen-Shannon Uyuşmazlığı	0.2599	0.4882	0.6764
Öklid Uzaklığı	0.2577	0.4048	0.7065



Şekil 5.13. FastText ile temsil edilen aday cevap cümlelerinin benzerlik ölçütlerine göre dağılımları.

Tablo 5.19 incelendiğinde, iki veya üç aday cevap cümlesi kullanıldığında elde edilen başarımın, tek aday cevap cümlesi olduğunda elde edilen başarımdan daha yüksek olduğu görülmektedir. Tek aday cevap olduğunda elde edilen doğruluk, iki aday cümle kullanıldığında yaklaşık olarak 0.14, üç aday cümle kullanıldığında ise yaklaşık olarak 0.20 artmıştır.

Tablo incelendiğinde, alınan diğer sonuçlarda olduğu gibi, yine en iyi sonucu veren benzerlik metriği Overlap katsayısıdır. Burada kullanılan benzerlik metrikleri, önceki sonuçlara benzer sonuçlar sergilediği için iki ve üç aday cevap cümlesi için Overlap, Kosinüs, Öklid, Kulsinski, Jensen-Shannon ve Dice metrikleri birleştirilmiş ve deneyler gerçeklenmiştir. Bu işlem ile elde edilen deneysel sonuçlar Tablo 5.20'de sunulmuştur.

Tablo 5.20. FastText ile temsil edilen birden fazla aday cümle için benzerlik ölçütlerinin birleştirilmesi ile elde edilen sonuçlar.

Benzerlik Ölçütü	1 aday cümle için Doğruluk	2 aday cümle için Doğruluk	3 aday cümle için Doğruluk
Overlap + Kosinüs + Öklid + Kulsinski + Jensen Shannon + Dice	0.8854	0.9660	**0.9865**

Tablo 5.20'den de anlaşılacağı üzere, birden fazla aday cevap cümlesi üzerinde benzerlik ölçütleri birlikte kullanıldığında doğruluk oranı artmaktadır.

Elde edilen tüm sonuçları genel olarak değerlendirecek olursak, gerek Word2Vec gerekse FastText için yapılan testlerde tek aday cevap cümlesi üzerinde bireysel benzerlik ölçütleri kullanıldığında elde edilen en iyi doğruluk değeri yaklaşık %71 iken, üç aday cevap cümlesi üzerinde birleştirilmiş benzerlik ölçütlerinin kullanılması ile doğruluk değeri %98'lere kadar yükselmiştir. Yapılan çalışmalar sonucunda 0.27 değerinde bir artış sağlanmıştır.

Veri setindeki cümleleri, Word2vec kelime gömülmesi ile temsil etmenin veya FastText kelime gömülmesi ile temsil etmenin, kayda değer bir başarım farkı oluşturmadığı yapılan deneysel çalışmalar sonucunda gözlemlenmiştir.

5.5.3 Geliştirilen soru yanıtlama mimarisi deneysel sonuçları

Tezin bu bölümünde, geliştirilen soru yanıtlama mimarisi için kullanılan derin öğrenme tabanlı öznitelik temsillerine, geliştirilen soru yanıtlama mimarisi için izlenilen deneysel sürece ve kullanılan değerlendirme ölçütleri ile elde edilen deney sonuçlarına yer verilmiştir.

5.5.3.1 Derin öğrenme tabanlı öznitelik temsilinin uygulanması

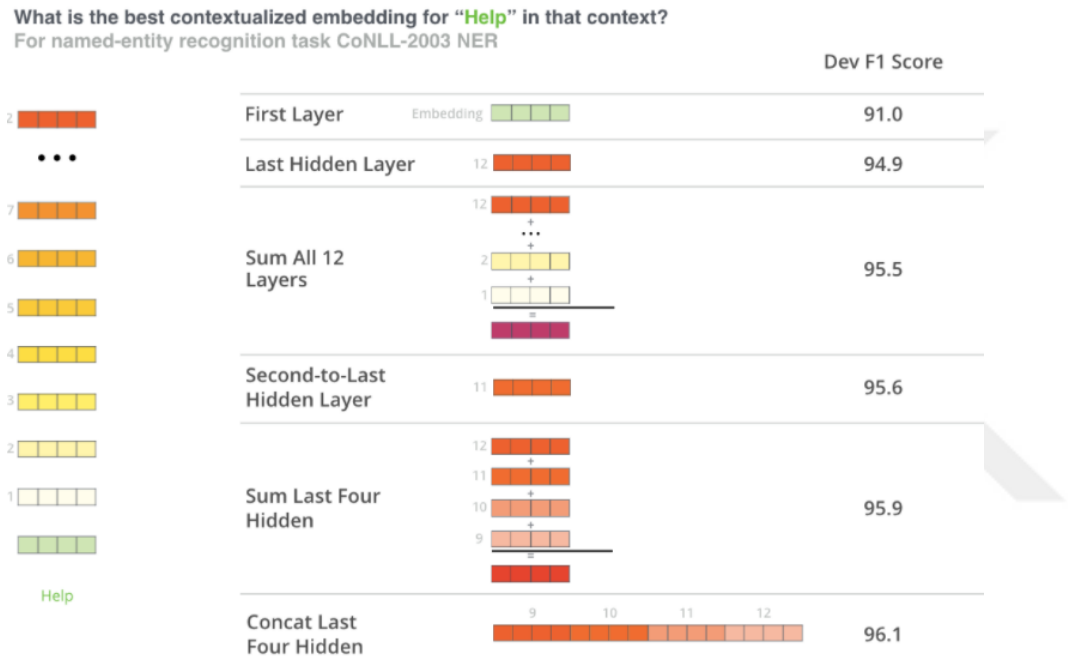
Soru yanıtlamada soruların ve ilgili paragraf cümlelerinin temsili için son zamanlarda NLP'de de sıklıkla kullanılan ve Bölüm 3.2'de bahsedilen derin öğrenme tabanlı kelime gömülmelerinden olan BERT, RoBERTa, DistilBERT, ALBERT, XLM ve XLNet yöntemlerinden yararlanılmıştır. Modellerden öznitelik vektörlerinin oluşturulabilmesi için katmanlardaki vektör çıktılarından faydalanılmıştır. Modellerin türüne göre çıktı bilgileri değişmektedir. BERT, RoBERTa ve ALBERT çıktı olarak hem cümle gösterimini hem de belirteç gösterimini verirken ve vektör uzunluğu 768 iken; XLNet, XLM ve DistilBERT sadece belirteç gösterimi vermekte ve vektör uzunluğu 3072'dir.

Modellerde her katmanda her belirteç için ayrı bir belirteç gömülmesi oluşmaktadır. Kelime vektörlerini elde etmek için literatürde iki yöntem kullanılmaktadır. Bunlar; arka arkaya ekleme (concatenate) ve toplama (summing) işlemidir. Örneğin; BERT-base modelinde, 12 Transformer kodlayıcı kullanıldığından dolayı, her bir belirteç için 12 katmanın her birinden bir belirteç gömülmesi üretilmektedir. Bu gömülmelerden hangisinin kullanılabileceği ile ilgili Devlin et al. (2019), CONLL 2003 NER veri seti üzerinde NER görevi için bazı deneyler gerçekleştirmiştir. Elde edilen deney sonuçlarına Şekil 5.14'te yer verilmiştir.

Sonuçlar incelendiği zaman, son dört katmanın arka arkaya eklenmesi ile elde edilen başarımın diğer birleştirme yöntemlerine nazaran daha iyi sonuçlar verdiği gözlemlenmiştir. Bu doğrultuda, yapılan uygulamada, belirteç vektörlerinin elde edilmesi için son dört katmanı arka arkaya ekleme işlemi kullanılmıştır. Her bir

katmandan 768 uzunluklu bir belirteç vektörü elde edildiği için, son dört katmanın arka arkaya eklenmesi ile 3072 (768x4) uzunluklu bir vektör oluşturulmuş olur.

XLNet, XLM ve DistilBERT modellerinin sadece belirteç gösterimi verdiğinden ilk paragrafta bahsedilmişti. Bu modelleri kullanarak cümle gösterimini elde edebilmek için, ilgili cümleye ait olan belirteç gömülmelerinin ortalaması alınmıştır. Böylece, 3072 uzunluklu bir cümle gömülme vektörü elde edilmiştir.



Şekil 5.14. CONLL 2003 NER veri seti üzerinde NER görevi deney sonuçları (Alammar, 2021).

Yapılan bütün işlemler hem soru cümlesi için hem de paragraf cümleleri için ayrı ayrı gerçekleştirilmiştir. Bu sayede, her modelden hem soru cümlesi hem de paragraf cümleleri için öznitelik vektörleri elde edilmiştir.

Ek olarak, temelini Reimers ve Gurevych (2019),'nin attığı cümle gömülmeleri yaklaşımı kullanılarak da cümle öznitelik vektörleri elde edilmiş ve deneysel sonuçlar alınmıştır. Sentence-BERT (SBERT) olarak adlandırılan çalışmada, Kosinüs benzerliği kullanılarak anlamsal olarak karşılaştırılabilecek cümle gömülmeleri oluşturabilmek için Siyam Ağı (Siamese Network) ve üçlü ağ (triplet network) yapısı kullanılmıştır (Reimers ve Gurevych, 2019).

5.5.3.2 Derin öğrenme tabanlı öznitelik temsili ile soru yanıtlama deneysel süreci ve sonuçları

Bu bölümde, SQuAD veri seti üzerinde derin öğrenme tabanlı öznitelik temsilleri kullanılarak geliştirilen soru yanıtlama mimarisi için izlenilen deneysel süreç ve elde edilen deneysel sonuçlar sunulmuştur.

Kullanılan öznitelik temsilleri ve benzerlik yöntemi Python programlama dili ile geçekleştirilmiştir. Bilgisayar kapasitesinin yetersiz olmasından dolayı deney sonuçları, Google Colaboratory (COLAB) kullanılarak TPU üzerinde elde edilmiştir.

Öznitelik vektörlerinin elde edilebilmesi için bir Python kütüphanesi olan ve belirteç sınıflandırma, metin sınıflandırma gibi birçok görev için kullanılabilen derin öğrenme modellerini uygulamamızı sağlayan Transformers, Sentence-Transformers ve PyTorch kütüphanesinden faydalanılmıştır.

Derin öğrenme yöntemleri için önceden eğitilmiş modeller kullanılmıştır. Hangi yöntem ile hangi modelin ve hangi belirteç ayırıcının kullanıldığı bilgisine Tablo 5.21'de yer verilmiştir.

Tablo 5.21. Öznitelik vektörünü oluşturmak için kullanılan derin öğrenme yöntemleri ve modelleri.

Yöntem	Model	Belirteç Ayırıcı
BertModel	bert-base-uncased	BertTokenizer
RobertaModel	roberta-base	RobertaTokenizer
AlbertModel	albert-base-v2	AlbertTokenizer
XLNetModel	xlnet-base-cased	XLNetTokenizer
XLMModel	xlm-mlm-en-2048	XLMTokenizer
DistilBertModel	distilbert-base-uncased	DistilBertTokenizer

Modele verilecek olan maksimum belirteç sayısı, max_sequence_length olarak ifade edilmiştir ve değeri 128 olarak alınmıştır.

Veri seti okunduktan sonra, derin öğrenme modellerine girdi olarak verilmek için cümlelerden üç bilgi oluşturuldu. Bunlar;

- **input_id vektörü:** Cümledeki belirteçlerin indekslerinin tutulduğu vektördür.
- **input_mask vektörü:** Cümledeki belirteç sayısı, maksimum belirteç sayısından küçük ise "doldurma (padding)" işlemi yapılır. Bu vektör, cümledeki belirteçleri ayırabilmek için kullanılır. Vektörde, cümledeki belirteçler için 1 ile, doldurma olan belirteçler 0 ile ifade edilir.
- **segment_id vektörü:** Birden fazla cümle girdi olarak verildiğinde, hangi belirteçlerin hangi cümleye ait olduğunu ayırt edebilmek için kullanılan vektördür. Birinci cümle için 0, ikinci cümle için 1 değeri kullanılır.

İlk olarak, input_id vektörü için modele özgü belirteç ayırıcı kullanılarak, cümleler belirteçlerine ayrılmıştır. Ardından, Bölüm 3.2.1.1'de bahsedilen [CLS] ve [SEP] belirteçleri kullanılarak; belirteçler, modele uygun bir şekilde hazırlanmıştır. Bu işlem Tablo 5.22'deki kurallar dikkate alınarak gerçekleştirilmiştir. Ardından, belirteçlerin idleri elde edilerek, input_id vektörü oluşturulmuştur.

input_mask vektörü oluşturulurken model için sağdan doldurma gerekiyorsa, input_mask vektörü, belirteç sayısı kadar 1 ile doldurulur ve ardından "*max_sequence_length - belirteç sayısı*" kadar 0 ile doldurulur. Şayet soldan doldurma işlemi gerekiyorsa (XLNet modeli gibi), input_mask vektörü, "*max_sequence_length-belirteç sayısı*" kadar 0 ile doldurulur ve ardından belirteç sayısı kadar 1 ile doldurulur.

Uygulamada tek bir cümle girdi olarak kullanıldığı için, segment_id vektörü 0 değerlerinden oluşturulmuştur.

Tablo 5.22. Derin öğrenme modellerine özgü başlangıç girdileri.

Yöntem	Başlangıç Girdisi
Bert	[CLS] + tokens + [SEP] + padding
RoBERTa	[CLS] + prefix_space + tokens + [SEP] + padding
ALBERT	[CLS] + tokens + [SEP] + padding
XLNet	Padding + tokens + [SEP] + [CLS]
XLM	[CLS] + tokens + [SEP] + padding
DistilBERT	[CLS] + tokens + [SEP] + padding

Elde edilen input_id, input_mask ve segment_id vektörü, tensör veri seti haline dönüştürülerek, modele girdi olarak verilmiştir. Modellerden öznitelik vektörlerinin oluşturulabilmesi için Bölüm 5.5.3.2'de anlatılan işlemlerden faydalanılmıştır. Ardından, paragraf cümleleri ve soru cümlesi arasında Kosinüs benzerliği kullanılarak, aday cevap cümleleri elde edilmeye çalışılmıştır. Bunun için sırasıyla BERT, ALBERT, RoBERTa, XLM, XLNet ve DistilBERT modellerinden elde edilen öznitelikler üzerinde deneyler gerçekleştirilmiştir. İlk olarak, paragraftaki cümleler Kosinüs benzerliğine göre büyükten küçüğe doğru sıralanmıştır. İlk sırada yer alan cümle, soru cümlesi ile en benzer olan cümle olduğu için aday cevap cümlesi olarak seçilmiştir. Bulunan aday cümle ile gerçek cevabın geçtiği cümlenin idleri karşılaştırılarak modellerin doğruluk değeri hesaplanmıştır. Elde edilen sonuçlara Tablo 5.23'te yer verilmiştir.

Tablo incelendiğinde, SQuAD eğitim veri seti üzerinde kelime gömülmeleri oluşturulduğunda derin öğrenme modellerinin elde ettiği en yüksek başarım XLM modeline aittir. İkinci olarak DistilBERT, üçüncü olarak ise Roberta diğer yöntemlere kıyasla daha iyi sonuç vermiştir. Cümle gömülmeleri kullanıldığında en yüksek başarıma sahip olan model DistilBERT'tir. İkinci olarak BERT, üçüncü olarak ise ALBERT diğer modellere kıyasla daha iyi sonuç vermiştir. Bir modelin yanlış bulduğu cevap cümlesini, diğer modellerin doğru bulabileceği düşüncesi ile modellerin birlikte analizi yapılmıştır. Elde edilen sonuçlar Tablo 5.24'te sunulmuştur.

Tablo 5.23. SQuAD eğitim veri seti kelime ve cümle gömülmeleri üzerinde derin öğrenme modellerinin bireysel başarımları.

		Bert	Albert	XLNet	Roberta	DistilBert	XLM
Kelime gömülmeleri	Doğruluk	0.2365	0.2497	0.2539	0.3478	0.4631	**0.6264**
	Mikro F1-Skor	0.2365	0.2497	0.2539	0.3478	0.4631	**0.6264**
	Makro F1-Skor	0.1032	0.1003	0.0993	0.1985	0.3333	**0.5136**
	Ağırlıklı F1-Skor	0.2420	0.2531	0.2571	0.3537	0.4642	**0.6268**
Cümle gömülmeleri	Doğruluk	0.6735	0.5568	0.3557	0.4768	**0.6896**	0.3604
	Mikro F1-Skor	0.6735	0.5568	0.3557	0.4768	**0.6896**	0.3604
	Makro F1-Skor	0.4929	0.4077	0.2226	0.3357	**0.5309**	0.1766
	Ağırlıklı F1-Skor	0.6731	0.5579	0.3578	0.4783	**0.6889**	0.3636

Tablo 5.24. Derin öğrenme modellerinin birlikte kullanılması ile elde edilebilecek sonuçlar.

	Yöntem	Doğruluk
Kelime gömülmeleri	XLM + DistilBert + Roberta	0.7845
	XLM + DistilBert + Roberta + XLNet	0.8122
	XLM + DistilBert + Roberta + XLNet + Albert	0.8402
	XLM + DistilBert + Roberta + XLNet + Albert + Bert	**0.8600**
Cümle gömülmeleri	DistilBert + Bert + Albert	0.7861
	DistilBert + Bert + Albert + Roberta	0.8116
	DistilBert + Bert + Albert + Roberta + XLM	0.8475
	DistilBert + Bert + Albert + Roberta + XLM + XLNet	**0.8722**

Tablo 5.24 incelendiğinde, bir modelin kaçırdığı cevap cümlesini diğer modelin yakaladığı gözlemlenmiş ve bu nedenle modellerin birleştirilmesine karar verilmiştir. Bu işlem için soru yanıtlama problemi, bir sınıflandırma problemi gibi ele alınmış, Bölüm 3.3'te bahsi geçen sınıflandırma algoritmaları ve Bölüm 3.4'te detayları anlatılan topluluk öğrenmesi yöntemleri uygulanmıştır. Sınıf sayısı olarak, paragraflardaki maksimum cümle sayısı alınmıştır. SQuAD eğitim seti için, bir paragraftaki en fazla cümle sayısı 26 olduğu için, sınıf sayısı 26 olarak alınmıştır. SQuAD eğitim seti üzerinde bireysel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemleri uygulandığında elde edilen sonuçlara Tablo 5.25'te yer verilmiştir.

Tablo 5.25. Bireysel sınıflandırma algoritmaları ve topluluk öğrenme algoritmalarının SQuAD eğitim seti üzerinde uygulanması ile elde edilen deney sonuçları.

Yöntem	**Kelime gömülmeleri**								**Cümle gömülmeleri**							
	Doğruluk	Std	Mikro F1-Skor	Std	Makro F1-Skor	Std	Ağırlıklı F1-Skor	Std	Doğruluk	Std	Mikro F1-Skor	Std	Makro F1-Skor	Std	Ağırlıklı F1-Skor	Std
LR	0.2971	0.003	0.2971	0.003	0.0520	0.005	0.2277	0.003	0.3062	0.003	0.3062	0.003	0.0514	0.002	0.2382	0.003
Adaboost	0.3951	0.018	0.3951	0.018	0.0976	0.034	0.3002	0.0135	0.4247	0.010	0.4247	0.010	0.0777	0.023	0.3223	0.008
KNN	0.4186	0.122	0.4186	0.122	0.1506	0.100	0.3805	0.153	0.4714	0.165	0.4714	0.165	0.2049	0.155	0.4359	0.198
DT	0.4615	0.116	0.4615	0.116	0.2167	0.127	0.4348	0.147	0.5295	0.158	0.5295	0.158	0.2815	0.171	0.5049	0.189
NB	0.4836	0.112	0.4836	0.112	0.2298	0.1063	0.4646	0.136	0.5723	0.136	0.5723	0.136	0.3048	0.143	0.5568	0.162
SVM	0.4994	0.120	0.4994	0.120	0.2383	0.117	0.4788	0.148	0.5666	0.151	0.5666	0.151	0.3117	0.158	0.5476	0.179
RF	0.4995	0.108	0.4995	0.108	0.2532	0.112	0.4833	0.131	0.5874	0.129	0.5874	0.129	0.3324	0.147	0.5742	0.153
GRB	0.5170	0.110	0.5170	0.110	0.2601	0.108	0.5002	0.135	0.5954	0.136	0.5954	0.136	0.33368	0.1389	0.5840	0.156
Voting (KNN+ LR+SVM+NB +DT+RF +GRB)	0.5272	0.106	0.5272	0.106	0.2700	0.105	0.5119	0.130	0.6057	0.124	0.6057	0.124	0.3505	0.136	0.5953	0.149
Bagging	**0.5766**	0.005	**0.5766**	0.005	**0.4298**	0.058	**0.5750**	0.005	**0.6608**	0.004	**0.6608**	0.004	**0.4854**	0.0524	**0.6593**	0.004

Tablo incelendiğinde, bireysel sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemleri kullanıldığında, gerek kelime gömülmeleri için gerekse cümle gömülmeleri için en iyi performansı Bagging algoritması sergilemiştir. Kelime gömülmeleri ve cümle gömülmeleri için Bagging algoritmasının elde ettiği başarım, XLM'in ve DistilBERT'in tek başına elde etmiş olduğu başarıma ulaşılamamıştır. Buradaki başarımı arttırabilmek için, Bölüm 4.2'de detayları anlatılan soru yanıtlama yaklaşımları önerilmiştir. Önerilen soru yanıtlama yaklaşımları bir sınıf etiketi döndürmektedir. Döndürülen bu sınıf etiketi, ilgili soru için tahmin edilen cevap cümlesinin id'sini belirtmektedir. Önerilen yaklaşımlar SQuAD eğitim veri seti üzerinde test edilmiş ve yaklaşımların başarımı, tahmin edilen cevap cümlesinin id'si ile doğru cevabın geçtiği cümlenin id'si karşılaştırılarak ölçülmüştür. Elde edilen deney sonuçları Tablo 5.26'da sunulmuştur.

Tablo 5.26. Önerilen soru yanıtlama yaklaşımlarının SQuAD eğitim seti üzerinde uygulanması ile elde edilen deney sonuçları.

Yöntem	**Kelime gömülmeleri**				**Cümle gömülmeleri**			
	Doğruluk	Mikro F1-Skor	Makro F1-Skor	Ağırlıklı F1-Skor	Doğruluk	Mikro F1-Skor	Makro F1-Skor	Ağırlıklı F1-Skor
Yaklaşım 10	0.5440	0.5440	0.4757	0.5423	0.6722	0.6722	0.5149	0.6710
Yaklaşım 8	0.5663	0.5663	0.4581	0.5591	0.6480	0.6480	0.4733	0.6442
Yaklaşım 1	0.5713	0.5713	0.4867	0.5690	0.6835	0.6835	0.5224	0.6823
Yaklaşım 7	0.5769	0.5769	0.4682	0.5713	0.6505	0.6505	0.4773	0.6469
Yaklaşım 9	0.5893	0.5893	0.4909	0.5863	0.6572	0.6572	0.5016	0.6552
Yaklaşım 4	0.5941	0.5941	0.4837	0.5904	0.6614	0.6614	0.4866	0.6582
Yaklaşım 11	0.5967	0.5967	0.4982	0.5940	0.6829	0.6829	0.5371	0.6815
Yaklaşım 5	0.6031	0.6031	0.4901	0.6000	0.6654	0.6654	0.5020	0.6624
Yaklaşım 12	0.6061	0.6061	0.5026	0.6050	0.6840	0.6840	0.5356	0.6830
Yaklaşım 2	0.6133	0.6133	0.5022	0.6112	0.6856	0.6856	0.5356	0.6842
Yaklaşım 6	0.6144	0.6144	0.5008	0.6133	0.6711	0.6711	0.5169	0.6694
Yaklaşım 3	0.6225	0.6225	0.5078	0.6220	0.6857	0.6857	0.5360	**0.6848**
Yaklaşım 13	**0.6227**	**0.6227**	**0.5085**	**0.6224**	**0.6858**	**0.6858**	**0.5374**	0.6846

Tablo incelendiğinde, Kosinüs benzerliği ile aynı sınıf etiketini döndüren modellerin, doğruluk, kesinlik ve F1-Skor değerlerinin çarpılarak, ortalamasının

alınmasıyla (Yaklaşım 13) elde edilen sonuçların daha iyi olduğu gözlemlenmiştir. Önerilen ağırlıklı çoğunluk oylama yaklaşımları ile en yüksek başarıma sahip olan XLM ve DistilBERT modellerinin elde ettiği başarım ile aynı başarım elde edilmiştir.

Yapılan deneysel çalışmalara ek olarak, cümle gömülmeleri farklı modeller ile de oluşturulmuştur. Cümle gömülmelerini oluşturmak için kullanılan modeller ile ilgili bilgilere Tablo 5.27'de değinilmiştir.

Tablo 5.27. Cümle gömülmelerini oluşturmak için kullanılan modeller ve özellikleri.

Yöntem	Max_Sequence_Length	Boyut	Temel Model
distilroberta	256	768	distilroberta-base
miniLM_L3	128	384	miniLM-L3-H384-uncased
miniLM_L6	128	384	miniLM-L6-H384-uncased
mpnet	512	768	mpnet-base
nli_distilroberta	75	768	distilroberta-base
nli_mpnet	75	768	mpnet-base
nli_roberta	75	768	roberta-base
stsb_distilroberta	75	768	distilroberta-base
stsb_mpnet	75	768	mpnet-base
stsb_roberta	75	768	roberta-base

Önerilen yaklaşımlar (Bkz. Bölüm 4.2), Tablo 5.27'de bahsi geçen önceden eğitilmiş cümle gömülme modelleri (distilroberta, miniLM_L3, miniLM_L6, mpnet, nli_distilroberta, nli_mpnet, nli_roberta, stsb_distilroberta, stsb_mpnet, stsb_roberta) ile de kıyaslanmıştır. SQuAD eğitim veri seti üzerinde elde edilen deney sonuçlarına Tablo 5.28'de yer verilmiştir.

Tablo 5.28'de, cümle gömülme modellerinin bireysel başarıları ele alındığında, en yüksek başarım miniLM_L3 modeline aittir. Sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemleri baz alındığında, en iyi performans topluluk öğrenmesi algoritmalarından olan Bagging algoritmasına aittir.

Tablo 5.28. SQuAD eğitim seti üzerinde, önerilen yaklaşımların, cümle gömülme modellerinin, sınıflandırma algoritmalarının ve topluluk öğrenmesi yöntemlerinin uygulanması ile elde edilen sonuçlar.

Yöntem	Doğruluk	Mikro F1-Skor	Makro F1-Skor	Ağırlıklı F1-Skor
nli_roberta	0.7050	0.7050	0.5846	0.7030
stsb_mpnet	0.7105	0.7105	0.5714	0.7085
nli_mpnet	0.7121	0.7121	0.5297	0.7099
mpnet	0.7293	0.7293	0.6331	0.7273
miniLM_L6	0.7347	0.7347	0.6556	0.7329
stsb_roberta	0.7498	0.7498	0.6620	0.7483
stsb_distilroberta	0.7508	0.7508	0.6442	0.7494
distilroberta	0.7513	0.7513	0.6540	0.7495
nli_distilroberta	0.7518	0.7518	0.6339	0.7502
miniLM_L3	**0.7530**	**0.7530**	**0.6720**	**0.7514**
LR	0.4430	0.4430	0.0829	0.3829
Adaboost	0.4516	0.4516	0.0931	0.3400
KNN	0.5960	0.5960	0.3328	0.5649
DT	0.6463	0.6463	0.4185	0.6249
SVM	0.6743	0.6743	0.4649	0.6576
NB	0.6849	0.6849	0.4600	0.6712
RF	0.6980	0.6980	0.4852	0.6863
GRB	0.7055	0.7055	0.4741	0.6872
Voting (KNN+ LR+SVM+NB +DT+RF +GRB)	0.7134	0.7134	0.4934	0.7072
Bagging	**0.7602**	**0.7602**	**0.6078**	**0.7588**
Yaklaşım 4	0.6705	0.6705	0.6483	0.6623
Yaklaşım 7	0.6705	0.6705	0.6483	0.6622
Yaklaşım 8	0.7145	0.7145	0.6662	0.7103
Yaklaşım 5	0.7151	0.7151	0.6695	0.7110
Yaklaşım 9	0.7176	0.7176	0.6655	0.7166
Yaklaşım 6	0.7193	0.7193	0.6686	0. 7184
Yaklaşım 1	0.7246	0.7246	0.6775	0.7208
Yaklaşım 3	0.7631	0.7631	**0.7242**	0.7622
Yaklaşım 2	0.7636	0.7636	0.7131	0.7609
Yaklaşım 10	0.7665	0.7665	0.6853	0.7648
Yaklaşım 11	0.7699	0.7699	0.6965	0.7683
Yaklaşım 12	0.7701	0.7701	0.6959	0.7687
Yaklaşım 13	**0.7707**	**0.7707**	0.6985	**0.7691**

Tabloda, önerilen yaklaşımlar kendi içerisinde değerlendirildiğinde, doğruluk, mikro F1-skor ve ağırlıklı F1-skor değeri en yüksek olan yaklaşım,

Kosinüs benzerliğini ağırlık olarak kullanan Yaklaşım 13'tür. Makro F1-skor'a bakıldığında ise en iyi performans Yaklaşım 3'e aittir. Tablodaki sonuçlar genel olarak değerlendirildiğinde, cümle gömülme modellerinin bireysel başarılarında, önerilen yaklaşımlar sayesinde gerek doğruluk değeri gerekse F1-skor değerleri üzerinde yaklaşık %2.5 oranında bir artış sağladığı yapılan deneysel çalışmalar ile gözler önüne serilmiştir.

Yukarıda bahsi geçen cümle gömülme modelleri ve önerilen yaklaşımlar SQuAD geliştirme veri seti üzerinde de deneysel çalışmalara tabi tutulmuştur. Elde edilen deney sonuçları Tablo 5.29'da sunulmuştur.

Tabloda yer alan deney sonuçları, cümle gömülme modelleri bazında değerlendirildiğinde, stsb_roberta modeli doğruluk değeri, mikro F1-skor ve ağırlıklı F1-skor bakımından en iyi performansı sergilemiştir. Makro F1-skor bakımından en iyi performans, nli_distilroberta cümle gömülme modeline aittir. Deney sonuçları sınıflandırma algoritmaları ve topluluk öğrenmesi yöntemleri bakımından değerlendirildiğinde, en yüksek başarıma sahip yöntem topluluk öğrenme algoritmalarından olan Bagging algoritmasıdır. Sonuçlar önerilen yaklaşımlar açısından değerlendirildiğinde, doğruluk değeri, mikro F1-skor ve ağırlıklı F1-skor değeri en yüksek olan yaklaşım, ödüllendirme yaklaşımlarından biri olan Yaklaşım 12'ye aittir. Makro F1-skor'a göre en yüksek başarım Yaklaşım 3'e aittir.

Tablo 5.29'daki sonuçlar genel olarak değerlendirilecek olursa, önerilen yaklaşımlardan Yaklaşım 2, Yaklaşım 3, Yaklaşım 10, Yaklaşım 11, Yaklaşım 12 ve Yaklaşım 13, cümle gömülme modellerinin bireysel olarak elde ettiği en yüksek doğruluk değerinden daha iyi bir performans göstermiştir. Önerilen yaklaşımlar, cümle gömülme modellerinin bireysel başarılarına oranla yaklaşık olarak %2 daha yüksek bir başarım elde etmiştir.

Tablo 5.29. SQuAD geliştirme veri seti üzerinde, önerilen yaklaşımların, cümle gömülme modellerinin, sınıflandırma algoritmalarının ve topluluk öğrenmesi yöntemlerinin uygulanması ile elde edilen sonuçlar.

Yöntem	Doğruluk	Mikro F1-Skor	Makro F1-Skor	Ağırlıklı F1-Skor
stsb_mpnet	0.7123	0.7123	0.5429	0.7099
nli_roberta	0.7160	0.7160	0.5695	0.7133
nli_mpnet	0.7199	0.7199	0.5105	0.7172
mpnet	0.7316	0.7316	0.6066	0.7292
miniLM_L6	0.7376	0.7376	0.6014	0.7522
distilroberta	0.7509	0.7509	0.5124	0.7488
nli_distilroberta	0.7532	0.7532	**0.6801**	0.7511
miniLM_L3	0.7543	0.7543	0.5987	0.7522
stsb_distilroberta	0.7545	0.7545	0.6027	0.7527
stsb_roberta	**0.7556**	**0.7556**	0.6496	**0.7536**
LR	0.4594	0.4594	0.1099	0.4005
Adaboost	0.4620	0.4620	0.0924	0.3518
KNN	0.5997	0.5997	0.3172	0.5679
NB	0.6445	0.6445	0.4221	0.6379
DT	0.6455	0.6455	0.3947	0.6234
RF	0.6637	0.6637	0.4521	0.6578
SVM	0.6714	0.6714	0.4373	0.6538
GRB	0.6787	0.6787	0.4531	0.6696
Voting (KNN+ LR+ SVM+NB +DT+RF +GRB)	0.6893	0.6893	0.4693	0.6810
Bagging	**0.7582**	**0.7582**	**0.5971**	**0.7560**
Yaklaşım 7	0.6727	0.6727	0.6036	0.6630
Yaklaşım 4	0.6730	0.6730	0.6022	0.6634
Yaklaşım 8	0.7082	0.7082	0.6163	0.7013
Yaklaşım 5	0.7092	0.7092	0.6162	0.7022
Yaklaşım 9	0.7179	0.7179	0.6458	0.7167
Yaklaşım 6	0.7182	0.7182	0.6460	0.7169
Yaklaşım 1	0.7266	0.7266	0.6656	0.7219
Yaklaşım 2	0.7588	0.7588	0.6743	0.7545
Yaklaşım 3	0.7685	0.7685	**0.6858**	0.7673
Yaklaşım 10	0.7691	0.7691	0.6697	0.7668
Yaklaşım 11	0.7724	0.7724	0.6692	0.7701
Yaklaşım 13	0.7724	0.7724	0.6706	0.7702
Yaklaşım 12	**0.7735**	**0.7735**	0.6708	**0.7716**

6. SONUÇ VE GELECEK ÇALIŞMALAR

Bu tez çalışmasında, Soru Yanıtlama Sistemleri için melez makine öğrenmesi tekniklerine dayalı bir sistem geliştirilmesine odaklanılmıştır. Bunun için, soru yanıtlama problemi, bir sınıflandırma problemi olarak ele alınmış, etkin bir soru yanıtlama mimarisi geliştirebilmek için temel makine öğrenmesi yöntemlerinden faydalanılmış ve temel makine öğrenmesi yöntemlerinin soru yanıtlama problemindeki başarımının arttırılması sağlanmıştır.

Soru yanıtlama probleminde en iyi sonuçları elde edebilmek adına, ilk olarak, sorunun analizi için soru sınıflandırma probleminde bir performans artışı sağlanmaya çalışılmıştır. Başarımın arttırılması için çeşitli yöntemler uygulanmış ve deneysel sonuçlar değerlendirilmiştir. İkinci olarak, soru yanıtlama problemi için derin öğrenme tabanlı kelime gömülmeleri ve cümle gömülmeleri hibritlenerek kullanılmış, soru yanıtlama problemi sınıflandırma problemi gibi ele alınarak sorunun cevabını içeren cümle tespit edilmeye çalışılmıştır. Buradaki başarımın gözlemlenebilmesi için çeşitli yöntemler denenmiş ve deneysel sonuçlar değerlendirilmiştir.

Bir çok NLP alt görevinde, verilerin doğru bir şekilde temsil edilmesi ve uygun özniteliklerin seçilmesi, makine öğrenmesi yöntemlerinin performansını doğrudan etkilemektedir. NLP'de kullanılan veriler seyrek ve yüksek boyutlu olduğu için, verileri mümkün olduğunca doğru bir biçimde temsil etmek ve etkin bir öznitelik seçimi uygulamak, tez çalışmasının temel hedefleri arasındadır. Bu doğrultuda, TREC soru sınıflandırma veri seti üzerinde farklı öznitelik temsilleri ve öznitelik seçimleri denenmiştir. Öznitelik temsili ve öznitelik seçimi için temel sınıflandırma algoritmalarından Destek Vektör Makinesi, Karar Ağaçları, Naif Bayes, Lojistik Regresyon, Çok Katmanlı Algılayıcı, K-En Yakın Komşu, Stokastik Dereceli Azalma algoritmalarının, ve topluluk öğrenme algoritmalarından Rastgele Orman, Adaboost, Bagging ve Oylama algoritmalarının sergilediği performanslar incelenmiştir. Deneysel çalışmalar incelendiğinde, soruları en iyi temsil eden özniteliklerin unigram, bigram, POS ve NER özniteliklerinin birleştirilmesi ile elde edilen öznitelik temsili olduğu gözlemlenmiştir. Önerilen öznitelik temsil yöntemi ile

temel sınıflandırma algoritmalarının başarımlarında yaklaşık %5'e varan bir artış sağlanmıştır.

Soru yanıtlama problemi için derin öğrenme tabanlı kelime gömülmeleri ve cümle gömülmeleri üzerinde, temel sınıflandırma algoritmalarından SVM, DT, NB, LR, KNN algoritmalarının, topluluk öğrenme algoritmalarından Rastgele Orman, Adaboost, Gradyan Boosting, Bagging ve Oylama algoritmalarının, ve derin öğrenme tabanlı algoritmalarının sergilediği performanslar incelenmiştir. Deneysel çalışmalar incelendiğinde, önerilen yaklaşımlardan Yaklaşım 13'ün, kelime gömülme modellerinin en az bireysel başarımları kadar iyi bir performans sergilediği ve cümle gömülme modellerinin bireysel performanslarında, önerilen yaklaşımlar (Yaklaşım 2, Yaklaşım 3, Yaklaşım 10, Yaklaşım 11, Yaklaşım 12 ve Yaklaşım 13) sayesinde yaklaşık %2 - 2.5 oranında bir artış sağlandığı gözler önüne serilmiştir.

Tez çalışması kapsamında önerilen öznitelik temsil yöntemi, soru sınıflandırma problemi için temel makine öğrenmesi yöntemlerinin başarımlarında artış sağlamıştır. Ayrıca, tez kapsamında soru yanıtlama problemi için önerilen mimari de derin öğrenme tabanlı cümle gömülme modellerinin başarımlarında bir artış sağlamıştır. Önerilen öznitelik temsili, soru sınıflandırma probleminde olduğu gibi duygu analizi, cinsiyet tahminleme (gender prediction), metin sınıflandırma (text classification) gibi birçok NLP probleminde uygulanarak etkinliği değerlendirilebilir. Soru yanıtlama problemi için önerilen mimaride son-işlem seviyesinde gerçekleştirilen hibritleme işlemi, öznitelik seviyesinde yapılarak temel sınıflandırma algoritmalarının başarımları değerlendirilebilir.

KAYNAKLAR DİZİNİ

Achananuparp, P., Hu, X., Zhou, X., and Zhang, X., 2008, Utilizing sentence similarity and question type similarity to response to similar questions in knowledge-sharing community, In *Proceedings of QAWeb 2008 Workshop*, 214, Beijing, China,1-8p.

Aggarwal, C.C. and Zhai, C.X., 2012, A survey of text classification algorithms, In *Mining Text Data*, Springer, Boston, 163-222p.

Alammar, J., 2021, "The illustrated BERT, ELMo, and co. (How NLP cracked transfer learning)", http://jalammar.github.io/illustrated-bert/ (Erişim tarihi: 15 Aralık 2021).

Al-Shenak, M., Nahar, K. M. O. and Halwani, K.M.H., 2019, AQAS: Arabic question answering system based on SVM, SVD, and LSI, *Journal of Theoretical and Applied Information Technology*, *97*(2), 681-691p.

Aniol, A., Pietron, M. and Duda, J., 2019, Ensemble approach for natural language question answering problem, In *Proceedings of the 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW)*, Nagasaki, Japan, 180-183p.

Aydoğan, M., ve Karcı, A., 2019, Kelime temsil yöntemleri ile kelime benzerliklerinin incelenmesi, *Çukurova Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi*, *34*(2), 181-196s.

Benamara, F., 2004, Cooperative question answering in restricted domains: The WEBCOOP experiments, In *Proceedings of the Workshop Question Answering in Restricted Domains*, 42nd Annual Meeting of the Association for Computational Linguistics (ACL), 31-38p.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C., 2003, A Neural Probabilistic Language Model, *The Journal of Machine Learning Research*, 3, 1137–1155p.

Bird, S., Klein, E. and Loper, E., 2009, Natural Language Processing with Python: Analyzing Text with The Natural Language Toolkit, O'Reilly Media, Inc., CA, 504p.

KAYNAKLAR DİZİNİ (devam)

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T., 2017, Enriching word vectors with subword information, *Transactions of the Association for Computational Linguistics*, *5*, 135-146p.

Bonaccorso, G., 2017, Machine Learning Algorithms: Popular algorithms for data science and machine learning, Packt Publishing Ltd., Second Ed., 522p.

Breiman, L., 1996, Bagging predictors, *Machine Learning*, 24(2), 123-140p.

Breiman, L., 2001, Random forests, *Machine Learning*, 45(1), 5-32p.

Brill, E., Dumais, S. T., and Banko, M., 2002, An analysis of the AskMSR question-answering system. In EMNLP 2002, 257–264p.

Bu, F., Zhu, X., Hao, Y., and Zhu, X., 2010, Function-based question classification for general QA, In *Proceedings of the 2010 conference on empirical methods in natural language processing*, 1119-1128p.

Clark, A., Fox, C. and Lappin, S., 2010, The Handbook of Computational Linguistics and Natural Language Processing, Jhon Wiley & Sons, First Ed, UK, 118, 801p.

Conneau, A. and Lample, G., 2019, Cross-lingual language model pretraining, In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS '19)*, Canada, 7059–7069p.

Cortes, C. and Vapnik, V., 1995, Support-vector networks, *Machine learning*, *20*(3), 273-297p.

Cortes, E., Woloszyn, V., Binder, A., Himmelsbach, T., Barone, D., Möller, S., 2020, An empirical comparison of question classification methods for question answering systems, In *LREC*, 5408–5416p.

Devlin, J., Chang, M., Lee, K. and Toutanova, K., 2019, BERT: pre-training of deep bidirectional transformers for language understanding, In *Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, 4171-4186p.

KAYNAKLAR DİZİNİ (devam)

Dice, L.R., 1945, Measures of the amount of ecologic association between species, *Ecology*, 26(3), 297–302p.

Dwivedi, S. K., and Singh, V., 2013, Research and reviews in question answering system, International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) Procedia Technology 10, 417–424p.

Ferrone, L., and Zanzotto, F. M., 2020, Symbolic, Distributed, and Distributional Representations for Natural Language Processing in the Era of Deep Learning: A Survey. *Frontiers in Robotics and AI*, *6*, 153p.

Finkel, J.R., Grenager, T. and Manning, C., 2005, Incorporating non-local information into information extraction systems by gibbs sampling, In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, 363-370 p.

Freund, Y., and Schapire, R.E., 1996, Experiments with a new boosting algorithm, *Machine Learning: Proceedings of the Thirteenth International Conference*, 325-332p.

Gehrke, J., 2003, Decision trees, The Handbook of Data Mining, N. Ye (Ed.), Lawrence Erlbaum Associates Publishers, London, 3-24p.

Godbole, S., Grubinska, K. and Kelnreiter, O., 2020, "Economic uncertainty identification using transformers – improving current methods", https://humboldt-wi.github.io/blog/research/information_systems_1920/uncertainty_identification_transformers/# (Erişim tarihi: 15 Aralık 2021).

Goldberg, Y., 2017, Neural Network Methods in Natural Language Processing (Synthesis Lectures on Human Language Technologies), Morgan & Claypool Publisher, First Ed., 310p.

Gomaa, W. H., and Fahmy, A. A., 2013, A survey of text similarity approaches, I*nternational Journal of Computer Applications*, 68(13), 13-18p.

KAYNAKLAR DİZİNİ (devam)

Green, B. F., Chomsky, C. and Laughery, K., 1961, BASEBALL: An Automatic Question Answerer, In *Proceedings of the Western Joint Computer Conference*, New York: Institute of Radio Engineers, 219–224pp.

Guda, V., Sanampudi, S. K. and Manikyamba, I. L., 2011, Approaches for question answering systems, *International Journal of Engineering Science and Technology (IJEST), 3*(2), 990-995p.

Han, J. and Kamber, M., 2006, Data Mining Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, 743p.

Hao, T., Xie, W. and Xu, F., 2015, A WordNet expansion-based approach for question targets identification and classification. In *Chinese computational linguistics and natural language processing based on naturally annotated big data*, Springer, 333-344p.

Haris, S.S. and Omar, N., 2012, A Rule-based Approach in Bloom's Taxonomy-Question Classification through Natural Language Processing, *IEEE*, 410-414p.

Higashinaka, R. and Isozaki, H., 2008, Corpus-based question answering for why-questions, In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP)*, 418-425pp.

Ho, T. K., 1998, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(8), 832-844p.

Hossin, M., and Sulaiman, M. N., 2015, A review on evaluation metrics for data classification evaluations, *International Journal of Data Mining & Knowledge Management Process (IJDKP), 5*(2), 1-11p .

Huang, Z., Thint, M. and Qin, Z., 2008, Question classification using head words and their hypernyms, In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, (EMNLP '08), 927–936p.

KAYNAKLAR DİZİNİ (devam)

Huang, Z., Thint, M. and Çelikyilmaz, A., 2009, Investigation of question classifier in question answering. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, (EMNLP '09)*, 543–550p.

Indurkhya, N. and Damerau, F. J., 2010, Handbook of Natural Language Processing, Chapman & Hall/CRC, Second Edition, Boca Raton, 676p.

Joachims, T., 1998, Text categorization with support vector machines: Learning with many relevant features, In *Proceedings of the European Conference on Machine Learning*, Springer, Berlin, Heidelberg, 137-142p.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T., 2016a, Bag of tricks for efficient text classification, *arXiv preprint arXiv:1607.01759*.

Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., and Mikolov, T., 2016b, Fasttext. zip: compressing text classification models, *arXiv preprint arXiv:1612.03651*.

Jurafsky, D. and Martin, J.H., 2021, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Third Ed. Draft, 623p.

Kolomiyets, O. and Moens, M. F., 2011,A survey on question answering technology from an information retrieval perspective, *Information Sciences, 181*(24), 5412-5434p.

Kuncheva, L., 2014, Combining Pattern Classifiers: Methods and Algorithms, John Wiley & Sons Publishers, New Jersey, 376p.

Lende, S. P. and Raghuwanshi, M. M., 2016, Question answering system on education acts using nlp techniques, In *Proceedings of the World Conference on Futuristic Trends in Research and Innovation for Social Welfare (WCFTR '16)*, IEEE, 1-6 p.

Li, X. and Roth, D., 2002, Learning question classifiers, In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, 556–562p.

KAYNAKLAR DİZİNİ (devam)

Liu, H., and Setiono, R., 1995, Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, 388-391p.

Liu, Q., Kusner, M. J., and Blunsom, P., 2020, A survey on contextual embeddings, *arXiv preprint arXiv:2003.07278*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V., 2019, RoBERTa: A robustly optimized bert pretraining approach, *arXiv preprint arXiv:1907.11692*.

Loni, B., 2011, A survey of state-of-the-art methods on question classification, Delft University of Technology, Tech. Rep, 1-40 p.

Lopez, V., Uren, V., Sabou, M., and Motta, E., 2011, Is question answering fit for the semantic web?: A survey, *Semantic Web*, *2*(2): 125-155p.

Manning, C.D. and Schütze, H., 1999, Foundations of Statistical Natural Language Processing, MIT Press, Cambridge, USA, 704p.

Manning, C.D., Raghavan, P. and Schütze, H., 2009, Introduction to Information Retrieval, Cambridge University Press, England, 482p.

Marsh, B., 2016, Multivariate Analysis of the Vector Boson Fusion Higgs Boson, University of Missouri, 16p.

McCann, B., Bradbury, J., Xiong, C., and Socher, R., 2017, Learned in translation: contextualized word vectors, *NIPS '17: Proceedings of the 31st Conference on Neural Information Processing Systems* (Dec 2017), 6297-6308p.

Metzler, D. ve Croft, W.B., 2005, Analysis of statistical question classification for fact-based questions, *Information Retrieval*, 8, 481–504p.

Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013a, Efficient estimation of word representations in vectorspace, *arXiv preprint arXiv:* 1301-3781p.

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

KAYNAKLAR DİZİNİ (devam)

Mishra, A., and Jain, S. K., 2016, A survey on question answering systems with classification, *Journal of King Saud University-Computer and Information Sciences*, *28*(3), 345-361p.

Mitchell, T. M., 1997, Machine Learning, Burr Ridge, *IL: McGraw Hill*, 45(37), 870-877p.

Mohasseb, A., Bader-El-Den, M., and Cocea, M., 2018, Question categorization and classification using grammar based approach. *Information Processing & Management*, *54*(6), 1228-1243p.

Moldovan, D., Clark, C., Harabagiu, S., and Maiorano, S., 2003, Cogex: A logic prover for question answering, In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language* Technology, 1, 87-93p.

Mollá, D. and Vicedo, J. L., 2007, Question answering in restricted domains: An overview, *Computational Linguistics*, *33*(1), 41-61p.

Monz, C., 2004, Minimal span weighting retrieval for question answering. In SIGIR Workshop on Information Retrieval for Question Answering, 23–30p.

Mukhopadhyay, S., 2018, Advanced Data Analytics Using Python: With Machine Learning, Deep Learning and NLP Examples, Apress, First Ed., India, 195p.

Naili, M., Chaibi, A. H., and Ghezala, H. H. B., 2017, Comparative study of word embedding methods in topic segmentation. *Procedia computer science*, 112, 340-349p.

Naseem, U., Razzak, I., Khan, S.K., and Prasad, M., 2021, A comprehensive survey on word representation models: From classical to state-of-the-art word representation language models, *Transactions on Asian and Low-Resource Language Information Processing*, *20*(5), 1-35p.

Niuniu, X., and Yuxun, L., 2010, Review of decision trees, In *Proceedings of the Third IEEE International Conference on Computer Science and Information Technology,* 105-109p.

KAYNAKLAR DİZİNİ (devam)

Ojokoh, B., and Adebisi, E., 2019, A review of question answering systems, *Journal of Web Engineering*, 17(8), 717-758p.

Ollagnier, A., and Williams, H., 2019, Classification and event identification using word embedding, *Trans. Associat. Comput. Linguist.*, 7, 91-122p.

Onan, A., 2016, Görüş Sınıflandırma İçin Makine Öğrenmesi Algoritmalarına Dayalı Bir Yöntem Tasarımı ve Gerçekleştirimi, Doktora Tezi, Ege Üniversitesi Fen Bilimleri Enstitüsü, 175s (yayımlanmamış).

Pan, Y., Tang, Y., Lin, L., and Luo, Y., 2008, Question classification with semantic tree kernel, In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* , ACM, 837-838p.

Pasca, M., 2003, Open-Domain Question Answering from Large Text Collections, *CSLI Studies in Computational Linguistics*, 29(4), 665-667p.

Pennington, J., Socher, R., and Manning, C. D., 2014, Glove: global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532-1543p.

Peters, M.E., Ammar, W., Bhagavatula, C., and Power, R., 2017, Semi-supervised sequence tagging with bidirectional language models, *arXiv preprint arXiv:1705.00108*.

Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L., 2018, Deep contextualized word representations, In North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans,Louisiana, 2227–2237p,

Pradhan, N., Gyanchandani, M. and Wadhvani, R., 2015, A review on text similarity technique used in ir and its application, *International Journal of Computer Applications*, 120(9), 29-34p.

Pratiwi O.N. and Syukriyah, Y., 2019, Question classification for e-learning using machine learning approach, International Conference on ICT for Smart Society (ICISS), 1-4p.

KAYNAKLAR DİZİNİ (devam)

Pundge, A. M., Khillare, S. A., and Mahender, C. N., 2016, Question answering system, approaches and techniques: A review, *International Journal of Computer Applications,* 141(3), 34-39p.

Rajpurkar, P., Zhang, J., Lopyrev, K. and Liang, P., 2016, Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392p.

Razzaghnoori, M., Sajedi, H. and Jazani, I. K., 2018, Question classification in Persian using word vectors and frequencies. *Cognitive Systems Research*, *47*, 16-27p.

Reimers, N. and Gurevych, I., 2019, Sentence-bert: Sentence embeddings using siamese bert-networks, In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Hong Kong, China, 3982–3992p.

Rokach, L. and Maimon, O., 2010, Classification trees, Data Mining and Knowledge Discovery Handbook, Maimon, O. and Rokach, L. (Eds.), Springer Verlag, New York, 149-175p.

Saini, A., and Yadav, P.K., 2017, A survey on question-answering system, *International Journal of Engineering and Computer Science*, 6(3), 20453–20457p.

Samuel, A. L., 1967, Some studies in machine learning using the game of checkers, I*I—Recent progress, IBM Journal of research and development,* 11(6), 601-617p.

Sanh, V., Debut, L., Chaumond, J. and Wolf, T., 2019, DistilBERT, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

KAYNAKLAR DİZİNİ (devam)

Shmueli, G., Patel, N. R. and Bruce, P. C., 2010, Data Mining for Business Intelligence: Concepts, Tecniques and Applications in Microsoft Office Excel with XLMiner, John Wiley & Sons Publishers, New Jersey, 726p.

Silva, J., Coheur, L., Mendes, A. and Wichert, A., 2011, From symbolic to sub-symbolic information in question classification, *Artificial Intelligence Review*, 35(2), 137–154p.

Simmons, R. F., 1965, Answering english questions by computer: a survey, *Communications of the ACM* 8(1), 53–70pp.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.M., Ng, A. and Potts, C., 2013, Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1631–1642p.

Soricut, R. and Brill, E., 2006, Automatic question answering using the web: Beyond the factoid, *Information Retrieval*, *9*(2), 191-206p.

Sumathı, S., Rajappa, S., Kumar, L.A., Paneerselvam, S., 2021, Advanced Decision Sciences Based on Deep Learning and Ensemble Learning Algorithms: A Practical Approach Using Python, Nova Science Publishers, New York, 370p.

Sundblad, H., 2007, Question Classification in Question Answering Systems, PhD Thesis, Linköpings University, Linköping Studies in Science and Technology, 94p (unpublished).

Strzalkowski, T., and **Harabagiu,** S., 2006, Advances in Open Domain Question Answering (Text, Speech and Language Technology), Springer Science & Business Media, 32, Netherlands, 590p.

Şahin, G., 2017, Turkish document classification based on Word2Vec and SVM classifier, In *Signal Processing and Communications Applications Conference (SIU), 25th, IEEE*, 1-4p.

Taylor, W. L., 1953, Cloze procedure: a new tool for measuring readability, *Journalism Bulletin*, 30(4), 415–433p.

KAYNAKLAR DİZİNİ (devam)

Tjong Kim Sang, E.F. and De Meulder, F., 2003, Introduction to the conll-2003 shared task: language-independent named entity recognition, In *Proceeding of theSeventh Conference on Natural Language Learning at HLT-NAACL 2003*, 142-147p.

Tomas, D., and Giuliano, C., 2009, A semi-supervised approach to question classification. In *Proceedings of the European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning (ESANN '2009),* Belgium, 35-40p.

Uysal, A. K., and Gunal, S., 2014, The impact of preprocessing on text classification, *Information Processing & Management,* 50(1), 104-112p.

Vapnik, V., 2006, Estimation of Dependences Based on Empirical Data, Springer Science & Business Media, New York First Ed., 506p.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. and Polosukhin, I., 2017, Attention is all you need, *NIPS '17: Proceedings of the 31st Conference on Neural Information Processing Systems* (Dec 2017), 6000–6010p.

Wang, S., Zhou, W. and Jiang, C., 2020, A survey of word embeddings based on deep learning, *Computing*, *102*(3), 717-740p.

Woods, W., 1973, Progress in natural language understanding – An application to lunar geology, In *AFIPS '73 Proceedings of the June 4-8, 1973, National Computer Conference and Exposition,* ACM, 42: 441–450 p.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G. Hughes, M. and Dean, J., 2016, Google's neural machine translation system: Bridging the gap between human and machine translation, *arXiv preprint arXiv:1609.08144*.

KAYNAKLAR DİZİNİ (devam)

Xie, H., Qin, Z., Li, G. Y. and Juang, B. H., 2021, Deep learning enabled semantic communication systems. *IEEE Transactions on Signal Processing, 69*, 2663-2675p.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R. and Le, Q. V., 2019, Xlnet: Generalized autoregressive pretraining for language understanding, In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS '19)*, Canada, 5753–5763p.

Yazıcı, B., Yaslı, F., Gürleyik, H.Y., Turgut, U.O., Aktas, M. S., ve Kalıpsız, O., 2015, Veri madenciliğinde öznitelik seçim tekniklerinin bankacılık verisine uygulanması üzerine araştırma ve karşılaştırmalı uygulama, In *Proceedings of the 9th Turkish National Software Engineering Symposium (UYMS 2015)*,1483, Izmir, 72-82s.

Zhang, C. and Ma, Y., 2012, Ensemble Machine Learning: Methods and Applications Springer Science & Business Media, First Ed., New York, 332p.

Zhang, D., and Lee, W. S., 2003, Question classification using support vector machines, In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, 26-32pp.

Zhang, K. and Zhao, J., 2010, A Chinese question-answering system with question classification and answer clustering, 7th International Conference on Fuzzy Systems and Knowledge Discovery, A.B.D., 6, 2692-2696p.

Zhao, Y., and Zhang, Y., 2008, Comparison of decision tree methods for finding active objects, *Advances in Space Research*, 41(12), 1955-1959p.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A. and Fidler, S., 2015, Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27p.

TEŞEKKÜR

Tez çalışmam süresince değerli görüş ve önerileriyle bana yön veren, hiçbir konuda benden yardım ve ilgisini esirgemeyen danışmanım Sayın Doç. Dr. Hasan BULUT'a teşekkürlerimi sunarım. Tez izleme komitesi toplantılarında, tezin gelişimini izleyen, değerli görüş ve önerileri ile çalışmama katkı sağlayan Sayın Prof. Dr. M. Serdar KORUKOĞLU'na ve Sayın Doç. Dr. Korhan KARABULUT'a teşekkürlerimi sunarım. Ayrıca, tez savunma jürimde yer alan Sayın Prof. Dr. Vecdi AYTAÇ'a ve Sayın Doç. Dr. Aytuğ ONAN'a değerli katkılarından dolayı teşekkürlerimi sunarım.

Her zaman yanımda olan, eğitim hayatım boyunca benden hiçbir desteği esirgemeyen aileme ve hayatımın her alanında olduğu gibi, akademik yaşamımda da beni destekleyen, motive eden sevgili eşim İbrahim ÇINAROĞLU'na teşekkür ederim.

25 / 02 / 2022

Sinem ÇINAROĞLU

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ÇINAROĞLU, Sinem

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Doktora	Ege Üniversitesi/ Bilgisayar Müh.Böl.	Devam ediyor
Yüksek Lisans	Ege Üniversitesi/ Bilgisayar Müh.Böl.	2014
Lisans	Süleyman Demirel Ü./ Bilgisayar Müh.Böl.	2011
Lise	Devlet Lisesi	2005

İş Deneyimi

Yıl	Yer	Görev
2018- …	Necmettin Erbakan Üniversitesi	Araştırma Görevlisi
2012-2018	Ege Üniversitesi	Araştırma Görevlisi
2011-2012	Konya Üniversitesi	Araştırma Görevlisi

EKLER

Ek 1. Türkçe-İngilizce Sözlük

Ek 1. Türkçe-İngilizce Sözlük

Türkçe	İngilizce
1-gram	Unigram
2-gram	Bigram/Digram
2-gram Kodlama	Byte Pair Encoding
3-gram	Trigram
Açık Alan	Open Domain
Adaptif Artırım	Adaboost
Adlandırılmış Varlık Tanıma	Named Entity Recognition
Ağaç Çekirdek	Tree Kernel
Ağırlıklı Çoğunluk Oylama	Weighted Majority Voting
Ağırlıklı Ortalama	Weighted-average
Alt Kelime	Subword
Alt-Doğrusal	Sub-Linear
Anlamlılık Düzeyi	Significance Level
Anlamsal	Semantic
Arıtma Kaybı	Distillation Loss
Arka Arkaya Ekleme	Concatenate
Artırımlı	Boosting
Aşırı Uyumluluk	Overfitting
Bağlam	Context
Belge/Paragraf Alımı	Document/Passage Retrieval
Belirsiz	Ambiguous
Belirteç	Token
Belirteç Gömülmesi	Token Embedding
Belirteçlerine Ayırma	Tokenization

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Benzerlik	Similarity
Biçimbilimsel	Morphologic
Bilgi Alımı	Information Retrieval
Bilgi Çapı	Information Radius
Bilgi Çıkarımı	Information Extraction
Bir-Elemanı-Bir Kodlanmış	One-Hot Encoding
Biriktirme	Pooling
Birlikte Görünme Matrisi	Co-Occurrence Matrix
Boyut	Dimension
Bölümleme	Chunking
Büyüklük	Magnitude
Cevap Çıkarma	Answer Extraction
Cevap Türü Taksonomisi	Answer Type Taxonomy
Cinsiyet Tahminleme	Gender Prediction
Cümle Tamamlama Görevi	Cloze Task
Çapraz Doğrulama	Cross Validation
Çarpanlarına ayrılmış gömülme Parametrelendirmesi	Factorized Embedding Parameterization
Çeviri Dil Modeli	Translation Language Modeling
Çift Yönlü Dil Modeli	Bidirectionallanguage Model
Çoğunluk Oylama	Majority Voting
Çok Sınıflı	Multiclass
Çokanlamlılık	Polysemy

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Denetimli Öğrenme	Supervised Learning
Denetimsiz Öğrenme	Unsupervised Learning
Derlem	Corpus
Destek Vektör Makinesi	Support Vector Machine
Dikkat	Attention
Diller-arası	Cross-lingual
Dinamik Maskeleme Deseni	Dynamic Masking Pattern
Diziden Diziye	Sequence-To-Sequence
Doğal Dil Çıkarımı	Natural Language Inference
Doğal Dil İşleme	Natural Language Processing
Doğru Negatif	True Negative
Doğru Pozitif	True Positive
Doğruluk	Accuracy
Doküman Frekansı	Document Frequency
Doldurma	Padding
Duyarlılık	Recall
Duygu Analizi	Sentiment Analysis
En Yakın Komşu	Nearest Neighbor
F1-Skor	F1-Score
Fiil Öbeği	Verb Phrase
Gerçek Tabanlı Soru	Factoid Question
Gereksiz Kelime	Stopword
Gizli Anlamsal İndeksleme	Latent Semantic Indexing

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Gradyan İnişi	Gradient Descent
Hiyerarşik Softmax	Hierarchical Softmax
İki Yönlü Uzun Kısa-Süreli Bellek	Bidirectional Long Short-Term Memory
İkili	Binary
İki-Yönlü Çapraz Dikkat	Bidirectional Cross Attention
İnce-Ayar	Fine-Tuning
İnce-Taneli	Fine-Grained
İsim Öbeği	Noun Phrase
Kaba-Taneli	Coarse-Grained
Karakter Dizisi	String
Karar Ağaçları	Decision Trees
Karar Sınırı	Decision Boundry
Karmaşıklık Matrisi	Confusion Matrix
Katmanlar arası parametre paylaşımı	Cross-Layer Parameter Sharing
Katsayı	Coefficient
Kelime Çantası	Bag of Words
Kelime Gömülmesi	Word Embedding
Kelime Oluşumu	Word Occurrence
K-En Yakın Komşu	K- Nearest Neighbor
Kesinlik	Precision
Kesinlik Değeri	Certainty Value
Kısıtlı / Kapalı Alan	Restricted / Closed Domain

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Ki-Kare	Chi-Square
Konum Gömülmesi	Position Embedding
Korelasyon	Correlation
Kosinüs Gömülme Kaybı	Cosine Embedding Loss
Kök Bulma/ Kök Çözümleme	Stemming/Lemmatization
Lojistik Regresyon	Logistic Regression
Makine Öğrenmesi	Machine Learning
Makine Çevirisi	Machine Translation
Makro ortalama	Macro-average
Maskelenmiş Dil Modeli	Masked Language Model
Metinsel Gerektirim	Textual Entailment
Mikro ortalama	Micro-average
Model Eşlemesi	Pattern Matching
Naif Bayes	Naive Bayes
Nedensel Sorular	Causal Questions
Negatif Örnekleme	Negative Sampling
Onay Soruları	Confirmation Questions
Optimum ayırt edici düzlem	Optimal Separating Hyper-Plane
Otomatik-azalan	Autoregressive
Oylama	Voting
Öbek	Phrase
Ön-Eğitim	Pre-Training
Önyargı	Bias

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Örüntü Tanıma	Pattern Recognition
Öz-Dikkat	Self-Attention
Öznitelik	Feature
Öznitelik Çıkarımı	Feature Extraction
Paragraf Alımı	Paragraph Retrieval
Parça Gömülmesi	Segment Embedding
Pekiştirmeli Öğrenme	Reinforcement Learning
Pencere	Window
Rastgele Orman	Random Forest
Regresyon	Regression
Sahte Mail Filtreleme	Spam Filtering
Serbestlik Derecesi	Degree of Freedom
Sert Oylama	Hard Voting
Seyrek Matris	Sparse Matrix
Sınırlı Alan	Restricted Domain
Sinir Ağı	Neural Network
Siyam Ağı	Siamese Network
Skaler Çarpım	Dot product
Sonraki Cümleyi Tahminleme	Next Sentence Prediction
Soru Analizi	Question Analysis
Soru Taksonomisi	Question Taxonomy
Soru Yanıtlama	Question Answering
Soru Yanıtlama Sistemleri	Question Answering Systems

Ek 1. Türkçe-İngilizce Sözlük (devam)

Türkçe	İngilizce
Sözcük Dağarcığı	Vocabulary
Sözcük Türü Etiketi	Part-Of-Speech Tag
Sözdizimsel	Syntactic
Sürekli Kelime Çantası	Continuous Bag-of-Words
Terim Frekansı	Term Frequency
Terim Frekansı-Ters Doküman Frekansı	Term Frequency-Inverse Document Frequency
Toplama	Summing
Topluluk Öğrenme	Ensemble Learning
Uyuşmazlık	Divergence
Uzaklık	Distance
Uzun Kısa-Süreli Bellek	Long Short-Term Memory
Üçlü Ağ	Triplet Network
Varsayımsal Sorular	Hypothetical Questions
Yanlış Negatif	False Negative
Yanlış Pozitif	False Positive
Yapay Zekâ	Artificial Intelligence
Yığın Boyutu	Batch-size
Yumuşak Oylama	Soft Voting
Yüz Tespiti	Face detection
Yüzeysel Parçalama	Shallow Parsing