T.C.
İSTANBUL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

DERİN ÖĞRENME TEKNİKLERİNİ KULLANARAK BAĞLAMSAL BENZERLİĞE DAYALI OTANTİK İSLAMİ METİNLERİN AKILLI KÜMELENMESİ

Shakeel Ahmad Sheikh

Enformatik Anabilim Dalı

Enformatik Programı

DANIŞMAN
Prof. Dr. Sevinç GÜLSEÇEN

Prof. Dr. S. Gülseçen
1.8.2019

Ağustos, 2019

İSTANBUL

Bu çalışma 01.08.2019 tarihinde aşağıdaki jüri tarafından Enformatik Anabilim Dalı Enformatik Programında Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Jürisi

Prof. Dr. Sevinç GÜLSEÇEN (Danışman)
İstanbul Üniversitesi
Fen Fakültesi

Prof. Dr. Yücel Yüksel
İstanbul Üniversitesi
Edebiyat Fakültesi

Doç. Dr. Deniz Herand
Marmara Üniversitesi
İşletme Fakültesi

20.04.2016 tarihli resmi gazetede yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, İstanbul Üniversitesi′nin abonesi olduğu intihal yazılım programı kullanılarak Fen Bilimleri Enstitüsü′nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır.

ÖNSÖZ

Öncelikle danışmanım,Prof. Dr. Sevinç GÜLSEÇEN çalışmalarım ve araştırmalarım esnasında verdikleri destekten ve gösterdikleri sabırdan ötürü teşekkür ederim. Ayrıca Yrd. Doç. Kh Mohmad Shafi'ye ve Yrd. Doç. Dr. Tolga Ensari'ye de desteklerinden dolayı özellikle teşekkür ederim ve çalışmalarımda bana yardımcı olan,desteğini esirgemeyen İstanbul Üniversitesi'ndeki tüm arkadaşlarıma müteşekkir olduğumu belirtmek isterim.

İstanbul Üniversitesi'ndeki tüm profesörlerime,bütün modül ve laboratuarlarda bana her zaman yardım ettikleri ve desteklerini benden esirgemedikleri için teşekkür ederim. Son olarak,yıllarca süren çalışmalarım boyunca beni sürekli cesaretlendiren,ilgi ve sevgilerini eksik etmeyen aileme ve arkadaşlarıma teşekkür ederim. Bu başarı onların yardımı olmadan asla gerçekleşmezdi.

Teşekkürler!

Ağustos,2019 Shakeel Ahmad Sheikh

İÇİNDEKİLER

Sayfa No

ÖNSÖZ ..... **iv**

İÇİNDEKİLER ..... **vii**

ŞEKİL LİSTESİ ..... **ix**

TABLO LİSTESİ ..... **x**

ÖZET ..... **xi**

SUMMARY ..... **xii**

1. GİRİŞ **1**

1.1. MOTİVASYON 1

1.2. TERMİNOLOJİ 3

1.3. KÜMELENME & KATEGORİZASYON 4

1.4. KÜME,KORPUS,KELİME,BELİRTEÇ 4

1.5. KÜMELEME TEKNİKLERİ,TEMSİL,BENZERLİK VE KELİME GÖMME 4

1.6. ARAŞTIRMANIN HEDEFİ 5

1.7. METİN KÜMELEMESİ 5

1.8. KÜMELEME ÇEŞİTLERİ 6

2. DERİN ÖĞRENME **7**

2.1. MAKİNE ÖĞRENME ALGORİTMASININ SINIFLANDIRMASI 8

2.1.1. Denetimli Öğrenmesi 8

2.1.2. Denetimsiz Öğrenme 9

2.1.3. Yarı denetimli Öğrenme 9

2.1.4. Pekiştirmeli Öğrenme 9

2.2. YAPAY SİNİR AĞLARI (ARTİFİCİAL NEURAL NETWORKS (ANNS)) 9

2.3. İLERİBESLEMELİ DERİN SİNİR AĞIN MİMARİSİ (FEED FORWARD DEEP NEURAL NETWORK ARCHITECTURES) 11

2.4. EVRİŞİMLİ SİNİR AĞLARI (CONVOLUTIONAL NEURAL NETWORKS (CNNS)) 12

2.5. YİNELEMELİ YAPAY SİNİR AĞLARI (RECURRENT NEURAL NETWORKS (RNNS)) 13
2.6. AKTİVASYON FUNKSİYONU 13
2.6.1. Sigmoid (Lojistik) Funksiyonu 14
2.6.2. Tanh Funksiyonu 15
2.6.3. ReLU Funksiyonu 15
2.6.4. Softmax Funksiyonu 16
3. KELİME KALIPLAMA 17
3.1. KELİME GÖSTERİMLERİ 17
3.2. BAĞLAMSALLAŞTIRMA 19
3.3. GÜNCEL OLAN EN İYİ DURUM POPÜLER KELİME GÖMME YÖNTEMLERİ 20
3.3.1. Word2Vec 20
3.3.2. Doc2Vec 23
3.3.3. FastText 24
3.3.4. GloVe 25
4. METİN KÜMELEMESİ 27
4.1. KÜMELEME ALGORİTMALARI 28
4.1.1. K Ortalamalar Kümeleme 28
4.2. HİYERARŞİK KÜMELEME 29
4.2.1. BIRCH Algoritması 29
4.2.2. Küme Özellikleri ile Hesaplamalar 33
4.3. SPEKTRAL KÜMELEME 34
4.4. KENDİ KENDİNİ DÜZENLEYEN HARİTA VEYA KOHONEN HARİTALARI 34
4.5. *K*-ORTALAMA OLAN DERİN OTOMATİK KODLAYICILAR 36
4.6. DERİN GÖMÜLÜ KÜMELEME (DEEP EMBEDDED CLUSTERING (DEC)) . 36
4.6.1. KL ayrışması ile Kümeleme 37
4.7. METRİKLER 37
4.7.1. Psueod-F Statistic 38
4.7.2. Davies-Bouldin (DB) Indeks 39
4.7.3. Siluet Katsayısı 39
4.7.4. Dunn Indeks 39
4.7.5. Calinski-Harabasz Indeksi (CH Score) 40

5. METODOLOJİ,ARAÇLAR VE DENEYSEL KURULUM 41

5.1. VERİ KÜMESİ 41

5.1.1. Araştırma ve Tasarım Stratejisi 41

5.1.2. Veri Kümesi Ön İşlemesi 41

6. SONUÇLAR 44

6.1. GÖMME MODELLİ *K*-MEANS (*K*-MEANS WİTH EMBEDDING MODELS) 44

6.1.1. Gömme Modelli BIRCH (BIRCH with Embedding Models) 47

6.1.2. Gömme Modelli DEC (DEC with Embedding Models) 47

6.1.3. Gömme Modelli Spektral (Spectral with Embedding Models) 51

6.2. TARTIŞMA 51

7. SONUÇ VE GELECEK ÇALIŞMA 54

KAYNAKLAR 55

ÖZGEÇMİŞ 59

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1: Makine Ögrenmes ve Geleneksel Programlama Arasındaki Fark 8

Şekil 2.2: Çok katmanlı algılayıcı: Bir katmanlı derin sinir mimarisi..... 10

Şekil 2.3: Tek gizli katmanlı ileriye doğru Sinir Mimarisi (Feed Forward Neural Network with Single Hidden Layer) 11

Şekil 2.4: Yinelemeli yapay sinir ağları..... 14

Şekil 2.5: Aktivasyon Funksiyonu..... 15

Şekil 3.1: CBOW Modeli. Cümle :Kedi fareyi yedi. CBOW modeli,sol bağlamdaki "kedi" ve sağdaki "fare" kelimesini tahmin etmeye çalışır 21

Şekil 3.2: Skip-gram Modeli. Burada model sol bağlamı tahmin etmeye çalışır 'kedi' ve sağ 'farenin' merkez kelimesinden 'yedik' 22

Şekil 3.3: Paragraf Vektörünün Dağıtılmış Bellek sürümü..... 24

Şekil 4.1: CFT Genel Yapısı 31

Şekil 4.2: Kümeler (Clusters)..... 33

Şekil 4.3: Kendi Kendini Düzenleyen Harita programı(Self-Organizing Map scheme) 35

Şekil 4.4: Otomatik Kodlayıcı Yapısı(Autoencoder Architecture) 36

Şekil 5.1: Metodoloji (Methodology) 42

Şekil 6.1: Word2Vec esaslı(based) K-Means Kümelenme 45

Şekil 6.2: Doc2Vec esaslı(based) K-Means Kümelenme 46

Şekil 6.3: fastText esaslı(based) K-Means Kümelenme..... 46

Şekil 6.4: GloVe esaslı(based) K-Means Kümelenme..... 46

Şekil 6.5: Word2Vec esaslı(based) BIRCH Kümelenme..... 47

Şekil 6.6: Doc2Vec besaslı(based) BIRCH Kümelenme..... 48

Şekil 6.7: fastText esaslı(based) BIRCH Kümelenme 48

Şekil 6.8: GloVe esaslı(based) BIRCH Kümelenme 48

Şekil 6.9: Word2Vec esaslı(based) DEC Kümelenme 49

Şekil 6.10: Doc2Vec esaslı(based) DEC Kümelenme 49

Şekil 6.11: fastText esaslı(based) DEC Kümelenme..... 50

Şekil 6.12: GloVe besaslı(based) DEC Kümelenme 50

Şekil 6.13: Word2Vec esaslı(based) Spektral Kümelenme..... 51

Şekil 6.14: Doc2Vec esaslı(based) Spektral Kümelenme 52

Şekil 6.15: fastText esaslı(based) Spektral Kümelenme 52

Şekil 6.16: GloVe esaslı(based) Spektral Kümelenme 52

TABLO LİSTESİ

Sayfa No

Tablo 6.1: Gömme Modellerinin Parametreleri 44

Tablo 6.2: Kümeleme Modellerinin Parametreleri 45

Tablo 6.3: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile K-Means 45

Tablo 6.4: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile BIRCH 47

Tablo 6.5: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile DEC 49

Tablo 6.6: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile Spectral 51

ÖZET

YÜKSEK LİSANS TEZİ

DERİN ÖĞRENME TEKNİKLERİNİ KULLANARAK BAĞLAMSAL BENZERLİĞE DAYALI OTANTİK İSLAMİ METİNLERİN AKILLI KÜMELENMESİ

Shakeel Ahmad Sheikh

İstanbul Üniversitesi

Fen Bilimleri Enstitüsü

Enformatik Anabilim Dalı

Danışman: Prof. Dr. Sevinç GÜLSEÇEN

Bazı sosyal veya dini olaylarla ilgili karar verirken,İslam hukuk uzmanları (Islamic Jurists) meseleyle doğrudan ilişkili ya da bağlamsal olarak benzer (contextually analogic) ya da onunla ilişkili meseleleri bulmak için tüm temel metinleri taramalıdır. Bir hakim heyetinin veya hakimin vereceği hükmün,eğer meseleyle ilgili mevcut kaynak ve hukümleri dikkate alırsa muhkem olacağı açıktır. Eğer bir hukum sığ bir seviyede verilmişse,o zaman tarihte görüldüğü üzere toplumda bölünme ve çatışmalara yol açacaktır. Bu tezde biz sorgulama yapıldığından bağlamı çıkaran ve iligili külliyatı (Corpus) heyetin onune getiren,derin öğrenmeye dayalı farklı teknikler öneriyoruz.

Bağlamsal benzerlik ve kıyasa dayalı kümeleri geliştirmek için Derin öğrenme teknikleri,özellikle Derin Sinir Ağları ve denetimsiz kümeleme algoritmaları (*K*-Ortalama,BIRCH,Deep Embedded Clustering (DEC) ve Spektral) kullanılmıştır. Korpusun gömülmesi,Google'ın Word2Vec,Doc2vec,Facebook'un FastText ve Stanford'ın GloVe kullanılarak hesaplanmıştır. Gömülenler (embeddings), Word2Vec,Doc2Vec,Facebook'un fastText ve Stanford's GloVe modellerini çok boyutlu bir bağlamda gömmek için eğitilmiş metin korpusundan hesaplanmıştır.

Temmuz 2019, 71 sayfa.

Anahtar kelimeler: Derin Öğrenme,Kelime Gömme,Metin Kümeleme,Doğal Dil İşleme

SUMMARY

M.Sc. THESIS

INTELLIGENT CLUSTERING OF AUTHENTIC ISLAMIC TEXTS BASED ON CONTEXTUAL SIMILARITY USING DEEP LEARNING TECHNIQUES

Shakeel Ahmad Sheikh

Istanbul University

Institute of Graduate Studies in Science & Engineering

Department of Informatics

Supervisor: Prof. Dr. Sevinç GÜLSEÇEN

While issuing the rulings regarding some social or religious affair,the Islamic Jurists (legal experts) has to go into all the basic but authentic texts to find the stuff which is directly related or having the contextual analogy or has an association with it. It is obvious that the verdict or ruling of a jury or jurist will be strong if he takes into account all the contextually related stuff. If the decision is taken just at the shallower level,it potentially leads to the rift and conflict in the society and has been historically observed. In this thesis,we propose various techniques based on deep learning,which upon query,infers a context,and produces before the jury all the related corpus.

Deep learning techniques especially Deep Neural Networks and unsupervised clustering algorithms (K-Means,BIRCH,Deep Embedded Clustering (DEC) and Spectral) were used to develop clusters based on the contextual similarity and analogy. The embeddings were calculated in itself from the text corpus trained on embedding models Word2Vec,Doc2Vec,Facebook's fastText,and Stanford's GloVe in the multidimensional context. Google's Tensorflow along with Keras framework have been deployed to carry out the training of the autoencoder.

Temmuz 2019, 71 pages.

Keywords: Deep Learning,Word Embedding,Text Clustering,Natural Language Processing.

1. GİRİŞ

Yapay Zeka (YZ) makineleri,insan davranışının benzerini veya daha iyisini yapma kapasiteleri açısından değerlendirilir. YZ,Alan Turing'in önerdiği klasik kritere uygun davranış gösterirse bu Turing testini geçtiği anlamına gelir [47]. Eğer bir kullanıcı,bir makine tarafından üretilmiş çıktıya bakar ve bunu bir insanın mı,yoksa makinenin mi ürettiğini ayırt edemezse,YZ'nin Turing testini geçtiği söylenir. Bir chat-bot,insanların akıllı davranışlarını taklit eden bu tür bir makinedir. Chat-bot,son kullanıcılar ile etkileşime girmek için insanlar yerine otomatik programların kullanımına bir örnektir.

Tanımlama 1.1 (Doğal Dil İşleme) *"Doğal dil işleme*[1] *(Natural Language Processing (NLP))"ye yapay zeka(YZ)nin bir alt grubu oldğunu söyleyebiliriz çünkü bu hesaplama cihazlar ve doğal diller arasında etkileşimleriyle ilgileniyor. Özel olarak hesaplama cihazlar ,proses ,okumak ,gruplaştırmak,analiz etmek ve doğal dil mecmua etmek için nasıl programlanırlar bunu gösteriyor.*

Dil işleme uygulamalarını data işleme uygulamalarından ayırtan en önemli şey dil bilgisinin kullanılmasıdır. Unix ***wc*** *programı bayt sayılarını veya bir metin dosya içinde satır sayısını hesaplamak için kullandığı zaman bu bir basit veri işleme uygulamadır. Ama eğer aynı program bir dosya içinde kelime sayısını hesaplamak için kullanılırsa,bir dil işleme uygulaması sayılır.*

1.1. MOTİVASYON

Gantz [18]'a göre,önümüzdeki 10 yil içinde dijitalleştirilmiş verinin 90%'ı yapılandırılmamış veri olacaktır. Özellikle metinler yapılandırılmamış veri olup bunlar elektronik belgeler,web sayfaları,sosyal ağ siteleri gibi her yerde bulunur. İnsan için,doğal dil kolayca okunabilir ve anlaşılabilirdir,ancak metinsel verilerin büyük miktarlarından ötürü,insanların bu tür metinsel veri hacimlerini işlemesi ve bu kadar büyük olan anlam ve kalıpları kategorilere ayırma,sınıflandırması, neredeyse imkansızdır. Metin

[1] Terminoloji için: https://github.com/deeplearningturkiye/turkce-yapay-zeka-terimleri/blob/master/turkce-ingilizce.md

analizi,verilerin sınıflandırılmasında,ve düzenlenmesinde ve yapılandırılmamış verilerden çıkarımların,kuralların ve modellerin çıkarılmasına yardımcı olabilir. Metin verilerinden eğilimleri ve bilgi kalıplarını bulmak ve tahmin etmek için çeşitli yaklaşımlar uygulanmıştır [34]. Metin işleme aynı zamanda olası dolandırıcılık vakalarını tespit edebilmek için finansal olaylarda da kullanılabilir [24]. Son zamanlarda tıbbi endüstriler,daha faydalı ilaçları keşfetmek üzere tıbbi belgelerin kullanılması için metin işlemeye başlamıştır.

Metin işleme,faydalı verilerden ve metin verilerinden çıkarımlardan türetmek için çeşitli araç ve teknikleri uygular. Bu teknikleri,sınıflandırma,özetleme,keşif analizi,bilgi edinme (IR) gibi işlevlerine göre sınıflandırabiliriz. Keşif analizi,küme analizi ve konu çıkarımını içerir. Bu nedenle,metin işlemeyi,doğal dilde Derin Öğrenme Sinir Ağları'ndan (DLNN'ler) teknikleri kullanan veri madenciliğinin alt kümesi olarak düşünebiliriz. Bu araştırmada,İslami metinlere bağlamsal benzerliklerine dayanarak ilgili kümeleri elde etmek için çeşitli derin öğrenme teknikleri uygulanmiştir. Bu araştırma,İslami metin alanındaki problem çözme için derin öğrenme tekniklerini kullanan ilk çabalar arasındadır.

Metin verisinin tematik olarak sınıflandırılması,Yedi Epitomes'e dayanmaktadır,M.Ö' oluşturulan ilk sınıflandırma sistemidir [26]. Bu,rapsodiler,sanat,şarkı sözleri askeri metinler,kehanet ve sayılar gibi birçok farklı kategoriden oluşur. Bu şekilde mevcut tüm kitaplar (yaklaşık 10 bin) elle kategorize edilmiştir.

İslami metinlerin yani Hadis'in tematik olarak sınıflandırılması,Horasanlı alim Abdullah İbn El Mübarek'in 8.*yy* çalismasını dayanmaktadır. Hadis,Hazreti Muhammed'in (S.A.V) çeşitli durumlarda soyledikleri veya yaptıklarına ilişkin raporlardır. Sünnet,Muhammed'in (S.A.V) yaşam şeklini ifade eder. Hadislerin Arapça anlamı "anlatı" veya "baldric" dür ve Sünnet "Gelenekler" olarak çevrilir. Hadislerin,Muhammed(S.A.V) 'in Geleneklerini göstermesi gerekiyordu. Hadisler,Kur'an öğretilerinin anlaşılmasının ve Kur'an'da sunulan genel talimatlar hakkında ayrıntılı bilgi edinmenin anahtarıdır [33]. Hadislerin toplanması Hz. Osman İbn Affan (r.a) döneminde halifeliğin resmi talimatı altında Kur'an'ın toplanmasından,onlarca yıl sonra başladı. Hadislerin,Hz. Muhammed(S.A.V)'in vefatın sonra,peygamberin arkadaşlarının,insanların hadisleri ile Kuran karıştırılmasından çekindiğinden dolayı yazılmadığına inanılmaktadır. Halife zamanında,Müslümanlar arasında uyuşmazlıklara neden olabileceğinden [11] ya da Müslümanların Kuran'ı terk etmesine ve tahrif etmesine yol açabilir Hadislerin toplanması yasaklandı [1]. Ancak krallara

mektuplar,Müslümanlar için kullanım kılavuzları ve Peygamberin (S.A.V) ısrarı nedeniyle kaydedilenler gibi yaklaşık 65 diğer belge kaydedilmiştir [42].

Dinin anlaşılması için; ister Kuran olsun, Hadis olsun ,ister Hristiyanların İncili,Hinduların Vedası olsun,din metninin anlaşılması gerekir. Bu dini metinler kendi dinlerinden uzaklaştırılırsa, anlamını kaybedecektir [48]. İslam alimleri,İslami metinlerin derinliğinin anlaşılması için nitel ve kritik yöntemler kullanırlar. Bu nitel yöntemler,onbinlerce sayfalık düz metnin üzerinden geçmekten ibarettir. Bilgisayarın gücü ve dijital verilerin bulunabilirligi insanların bilgi saklama ve biriktirme yetenekleri büyük ölçüde geliştirmiştir. Gecmişte,korpusun kapsamı dardı ve verileri elle sınıflandırmak zor değildi. Sayısallaştırılmış metin verilerini değerlendirmek ve bunları anlamlı yapılara ayırmak yorucu ve çok zaman alan bir süreçtir ve çok büyük miktarda web verisinin mevcudiyeti ile imkansız hale gelmiştir. Özellikle,verinin bağlamsal benzerliğinden anlamlı bir içgörü elde etmek ve bilgiyi anlama,ve yorumlama yeteneğimizi toplama becerimizle eşleştirmek için İslami verileri düzenleme ve kategorilere ayırma yöntemine yeni bir yaklaşım gerekmektedir. İslami uzmanlara ve hukukçulara,ilgili metni bulmada ve ona göre bir analoji kurarak karar vermede yardımcı olmak gerekmektedir.

1.2. TERMİNOLOJİ

Bu tezde açıklamayı gerektiren bazi terimler kullanılmıştır. Öncelikle bu terimler açıklanacaktır. Bu,İslami Külliyatın bilgisayar bilimlerinde akıllı metin kümelenmesi üzerine bir tezdir. Ayrıca,dil teknolojisinde,metinlerde kümelenme ve metinlerin bağlamsal olarak uyumlu kümeler halinde kategorilere ayrıştırmak için uygulanan,denetlenmeyen derin sinir ağları ile ilgili bir tez olduğunu söyleyebiliriz. Bazıları dil işleme teknolojisinin bilgisayarlı dilbilimin bir alt kümesi,diğerleri ise bilgisayar bilimi alt alanı (özellikle AI) olarak adlandırılması gerektiğini savunmaktadır. IR,genellikle büyük bir korpustan bilgi bulma ve çıkarma ile ilgili bir yaklaşımdır. Metin kümeleme bir IR yöntemi olmanın yanı sıra yapılandırılmamış korpusu birleştirerek yeni bilgiyi keşfetmek ve bulmak için kullanılan denetimsiz bir derin sinir ağı yöntemidir. Tüm alanlara uyan tek bir evrensel kümeleme tekniği yoktur.

1.3. KÜMELENME & KATEGORİZASYON

Otomatik sınıflandırma,bir modelin bir belgeye veya metnin,önceden belirlenmiş kategorilerden oluşan bir alanına ait olduğuna karar vermesine izin vermek anlamına gelir. Kümelemede,model verilen bir metnin nasıl bölümlendirilmesi gerektiğine karar verir. Yeni bir metnin bilinen kategorilere göre sınıflandırılması amaçlandığında,kategorize edildiğinde,daha önce bilinmeyen yeni kümelerin keşfedilmesi ve keşfedilmesi istendiğinde kümelenme uygun olduğu için kategori kullanılabilir. Bu yöntemler görünmeyen korpus setinde merak uyandıran sonuçlar verebilir; sınıflandırma,bunları önceden tanımlanmış düzen ve yapıya göre gruplandırır,kümeleme,belirli metin korpusu kümesinin düzenini ve yapısını gösterir. Bu tez esas olarak,metin verilerinin derin sinir ağı modellerinin yardımı ile bağlamsal benzerliğe dayalı kümelenmesine odaklanmaktadır.

1.4. KÜME,KORPUS,KELİME,BELİRTEÇ

Bu tez çalışmasında,başlıklar,hiperbağlaçlar,belge numaraları vb. meta veriler kullanılmamaktadır. Bu tezde kümelendirilen varlıklar,korpus,metin,dokümanlar vb. olarak adlandırdığımız genellikle bağlama bağlı olarak kelimelerin dizisidir. Kelime ve belirteç,birbirinin yerine kullanılır. Korpus İngilizce dilinde olup genellikle boşluklarla,soru işaretleriyle,virgüllerle ve kullanılan diğer sınırlayıcılarla ayrılmıştır. Aynı alfabe grubuna sahip olan İngilizce ve diğer diller için bu iyi bir açıklama gibi görünmektedir,ancak Arapça,Türkçe ve diğer eklemeli diller için uygun olmayabilir. Bu tezde kelimeler,cümleler ve belgeler (Hadisler) birbirinin yerine kullanılmıştır.

1.5. KÜMELEME TEKNİKLERİ,TEMSİL,BENZERLİK VE KELİME GÖMME

Kümelenme için temel amaç,farklı tiplerde olabilir. Mesela,İslam hukukçuları,kararlarını verirken yardımcı olabilecek hangi metne başvurdukları metinleri,istedikleri şekilde seçtik için,bazı son derece önemli bir zaman noktasında,İslam hukukçularının buna uygun olarak hangi metni seçtiklerini seçmelerine yararlı olabilir. Korpus,çeşitli şekillerde türlere ayrılabilir. Birçok kümeleme tekniği vardır. Birçoğu,varlıklar (corpus) arasındaki (dis) benzerlikleri bilmeyi gerektirir. Bazıları her kelime,belge için temsil edilmesini ve gerektiğinde hesaplanabilecek benzerliğin bir tanımını gerektirir.

Kelimelerin,cümlelerin,belgelerin nasıl temsil edildiği ve benzerlik tanımının uygulamalar arasında nasıl değiştiğini tanımlamak gereklidir. Bunu başarmak için,derin öğrenme alanında,kelimeler,cümleler ve belgeler genellikle kelime gömmelerle temsil edilen çeşitli yöntemler vardır. Gömme kelimesi,kelimelerin,belgelerin anlamı hakkında bir şey ifade eden ve yakalayan kelimelerin ve belgelerin yoğun vektör temsilidir. Sözcük gömmeleri,çok büyük bir metin kümesinde çalışmak için derin bir sinir ağı kullanarak üretilir. Her bir kelime,belge,yüksek boyutlu gömme alanındaki bir nokta ile temsil edilir ve etrafındaki kelimelerin,belgelerin bağlamına göre bu noktalar öğrenilir ve karıştırılır. Farklılık,vektörler arasındaki mesafe olarak tanımlanabilir. Gelecek bölümlerde ele alınacak olan derin öğrenme modellerinde,gömme vektörleri olarak ifade edilen kelimeler veya belgeler örneğin korpus,varlıkları tanımlamak için kullanılan bir dizi özellik olarak kabul edilebilir.

Kümeleme,bölümleri veya ilgili bazı kriterleri yerine getirmeye ve karşılamaya çalışan bir dizi teknik sunar. Her kümedeki varlıkların olabildiğince homojen ve benzer olması gerekir. Bununla birlikte,benzerlik tanımlamasının gerçek varlıklar arasındaki benzerliği yansıttığını düşünmeliyiz. Bu çalışma temel olarak içerik bölümlerine,yani benzer içeriği paylaşan metin kümelerine odaklanmaktadır. Öyleyse soru,korpus'u nasıl temsil ediyor ve aralarındaki bağlamsal benzerliği nasıl tanımlıyoruz? Birçok derin öğrenme uygulamasında kümeler,içinde görünen sözcükler,belgeler ile temsil edilir ve iki belge,cümle (Hadisler) arasındaki benzerlik,Hadislerin vektörel yoğun temsilleri dikkate alınarak tanımlanır.

1.6. ARAŞTIRMANIN HEDEFİ

Bu araştırmanın amacı,farklı kelime gömme modellerinden üretilen temsillerin var olan çeşitli kümelenme teknikleri üzerindeki etkisini analiz etmek ve karşılaştırmaktır. Bu,uygulama odaklı bir araştırma çalışmasıdır,ancak sonuçlar kendi başlarına umut verici ve ilginç görünmektedir.

1.7. METİN KÜMELEMESİ

Kümelenme,metin gövdesi içerisindeki ilişkileri otomatik olarak analiz eden ve bağlamsal benzerliklerine dayanarak tutarlı ve kompakt yapılandırılmış metin kümeleri halinde sınıflandıran ve organize eden en önemli denetimsiz tekniklerden biridir. Bilgisayarla görme

alanında geniş bir uygulama alanı vardır [23],biyoinformatik [61],gibi NLP [59].

Son yıllarda,kümelenme sorunlarını ele almak için sayısız algoritma ortaya atılmıştır [14]. Bununla birlikte,tüm sorun alanlarına uyan tek bir algoritma yoktur. Bu görev alanına ve ele alınacak metin gövdesine (Corpus) bağlıdır. Genel olarak,kümelenme teknikleri-benzerlik tabanlı ve özellik tabanlı kümeleme algoritmaları olmak üzere iki yaklaşım vardır. Bir örneği ele alalım,İslam hukukçusunun ilgili kararının ayrıntılı arka planına ihtiyaç duyduğunu,ancak hadislerin ayrıntılarına bakmasının mümkün olmadığını varsayalım. Bu durumda bütün Hadislerin bağlamsal benzerliklerine dayanarak birçok farklı kümeye kümelemek yerine ne yapabiliriz. Genellikle,algoritmaların çoğu,doğuştan gelen yapıları,alttaki orijinal özellik alanından bulmaya çalışırlar.

1.8. KÜMELEME ÇEŞİTLERİ

Kümeleme genel olarak 2 alt gruba ayrılabilir:

Tanımlama 1.8.1 (Sert Kümelenme) *Bu kümelenme türünde,her bir veri noktası (bağlamımızdaki Hadis) kümelenmeye aittir veya tamamen değildir. Yukarıdaki örnekte olduğu gibi,her hadis tam olarak bir kümeye konur.*

Tanımlama 1.8.2 (Yumuşak Kümelenme) *Yumuşak kümelenmede,her bir Hadis'i ayrı bir kümeye koymak yerine,bu hadislerin bu kümelerde bulunma olasılığı veya olasılığı tespit edilmiştir. Örneğin,yukarıdaki örnekten her Hadise metnin* 50 *kümesinden birinde olma olasılığı verilmiştir.*

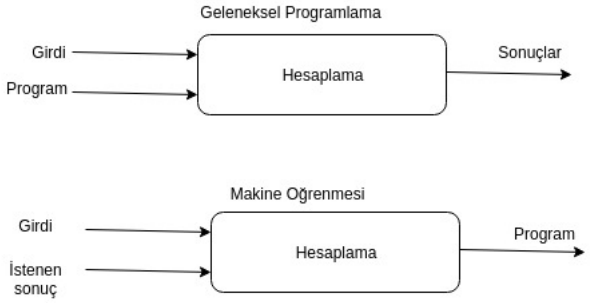
2. DERİN ÖĞRENME

Bu bölümde,derin öğrenme literatüründeki dikkate değer başlıkların genel bir açıklaması verilmiştir. Araştırmanın literatüründeki hali hazırdaki derin öğrenme teknikleri detaylı anlatılacak ve karşılaştırılacaktır. Derin öğrenmenin kullanımı ve uyglama alanına yapılan katkı çalışmanın tartışma bölümünü oluşturmaktadır:belirli bir uygulama için hangi teknoloji kullanılmalıdır? Belirli Derin öğrenmenin algoritmasına bakarak bir olay için kullanılan teknolojiya önsezi oluşturur. Oyle,belirli uygulama uzayına en uygun olan yöntemi elde edebiliriz.

Tanımlama 2.1 *Makine öğrenmesi (Machine Learning (ML)) Yapay zekanın alt alanıdır. Burada bilgisayar yazılım katmadan modeller ve sonuçlara dayanarak zekı bir şekilde davranmaktadır . Makine öğrenmesi (MÖ) ,algoritmaları oluşturarak sağlanan verilerle ilgili tahminlere odaklı bir çalışma alanıdır. Makine öğrenmesi bir fonksiyonu bulmayı ve öğrenmeyi hedefler.*

$$f : X \to Y$$

verilerin özellik giriş uzayı X en uygun tahmin edilen çıkış uzayına Y aktarır [2]. Örnek verilere dayalı bir matematik modeli oluşturur ki,buna eğitim verileri denir. Bu verilere dayanarak dayalı bir tahmın yapmak yada karar almak için bir fonksiyon oluşturur. Eğtim verileri etiketli (çift giriş ve istenen çıkış tahmini) yada etiketsiz verilerden oluşur. f fonksiyonu verileri kesin bir şekilde uymak için,uzayın tipine dayalı özel seçilmiştir. Mitchell [31] makine öğrenmesini şöyle tanımlar 'bir bilgisayarın uygulamasıdır,uygulama T görev sınıfı ve P performans ölçütüne önem vererek E deneyim üzerinden öğrenir,performans görevi T ve ölçü P ise E deneyimle geliştirir'. E burda veri grubu MÖ algoritmasi tarafından geçirir. P ölçü MÖ algoritması iyi bir şekilde çalıştığını gösterir.Sınıflandırma sorunlarında genellikle kesinlik performans ölçü olarak seçilir. MÖ uygulamaları: eposta filteremek,fotograflarin sınıflandırması,metin kümelenmesi,makine tercümesi icin kullanılır. MÖ kestirimsel çözümlemeye de işaret eder. Aaşağıdaki ***Şekil 2.1:*** *MÖ ve normal programlama arasındakı farkları göstermektedir.*



Şekil 2.1: Makine Ögrenmes ve Geleneksel Programlama Arasındaki Fark

2.1. MAKİNE ÖĞRENME ALGORİTMASININ SINIFLANDIRMASI

Dört kategori olarak sınıflandırır:

1. Denetimli Öğrenme (Supervised Learning)
2. Denetimsiz Öğrenme (Unsupervised Learning)
3. Yarı-Denetimli Öğrenme (Semi-supervised Learning)
4. Pekiştirmeli Öğrenme (Reinforcement Learning)

2.1.1. Denetimli Öğrenmesi

Denetimli öğrenmede eğitim süresinde bir çift giriş(X) ve çıkış(Y) olur,ve eşleştirmenin $Y = f(X)$ öğrenmesi bir algoritma elde ederiz.

Bu tür öğrenmede amacımız bu fonksıyonun yaklaşmasıdır,oyleki eğitim süresinde yeni giriş örneğinden çıkışı tahmın etmeliyiz. Denetimli MÖ bir danışman olarak düşünülebilir,burada danışman eğitim süresine danışıyor. Algoritmalar çift eğitim verisi sürekli olarak bir tahmin yapıyor. Algoritmalar hem öğreten tarafından hem de hazır düzeltilen giriş-çıkış çiftlerinder oluşur. En yakın komşu(Nearest Neighbor),Naive Bayes,Karar Ağacı(Decision Trees),Doğrusal Bağıntı(Linear Regression),Destek Vektör Makinası(Support Vector Machine(SVM)) en yaygın Denetimli algoritmalardır. Çözülebilen problemler gerileme görevler ve sınıflandırmadır.

2.1.2. Denetimsiz Öğrenme

Denetimsiz öğrenmesi modelleri etiketsiz veri ile beslenir. Denetimsizde,sadece giriş veri (X) olup denk düşen çıkış değişkenler olmaktadir. Eğitim sürecinde danışman bulunmaz. Amaç veride mevcut bulunan yapı yada kalıbı keşfetmektir. Bu MÖ algoritmaların ailesi tanımlayıcı modelleme ve kalıbı belirlemede kullanılır. Kalıpları bulmak için algoritmalar giriş verileri üzerine yöntemler uygulamaya çalışır,giriş noktaları toplamaya çalışır ,anlamlı,kesin ve derin anlayış geliştirmeye yardım eden kurallar oluşturur. Genellikle buna tanımlayıcı modeldir denir. Bazı genel denetlenmeyen algoritmalar kümelenme ve birleşme kuralını öğrenmesini içerir.

2.1.3. Yarı denetimli Öğrenme

Bu kategori denetimli ve denetimsiz arasında yer alıp bu ikisinin bir karışımıdır. Çok buyuk miktardaki veri girdisinden X sadece küçük bir kisminin Y etiketli oldugu uygulamalar yarı denetimli öğrenmedir. Gerçek hayatta ise butun verileri etiketlenmesi çok masraflıdır,zira bunu yapmak için uzmanlara ihtiyaç duyulur. Böyle bir durumda modelleri kurmak için yari denetimli algoritmalar en iyi seçenek sayılır. Örnek olarak fotoğrafların çoğunun etiketsiz yalnizca çok az kısmının köpek,kedi ve insan gibi etiketli olduğu fotoğraf arşivleri verilebilir.

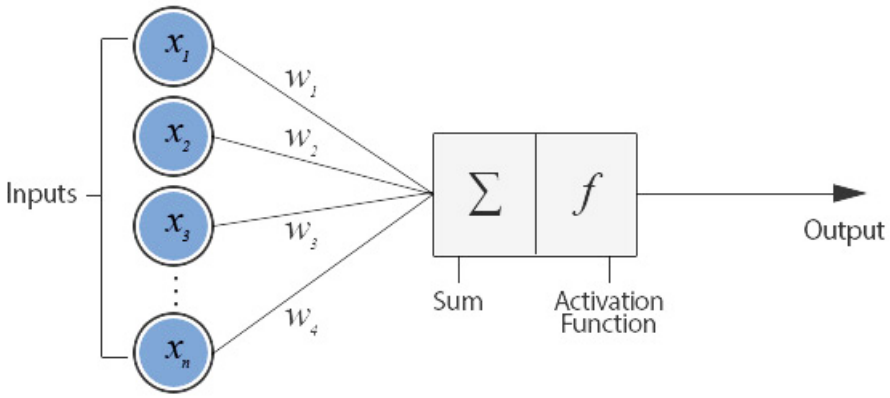
2.1.4. Pekiştirmeli Öğrenme

Bu teknikler,ajanların kazancı en üst seviyeye çıkarmak ya da bazı özel durumlarda riski en aza indirgemek amacli hareket etmeleri gereken ortamlar için idealdir. Ajanlar sürekli olarak yakın çevresinden tekrarlayan bir şekilde öğrenir.

2.2. YAPAY SİNİR AĞLARI (ARTİFİCİAL NEURAL NETWORKS (ANNS))

Yapay sinir ağlarının (YSA) çalışma prensibi *The Organization of behavior* [20] açıkladığı,hücrelerin birlikte aktive olmaları davranisini esas alir. Bu,sinir ağlarında her bir sinir hücresinin ağırlığının nasıl geliştiği ile karşılaştırılabilir:

$$\Delta w = \alpha x.y \tag{2.1}$$



Şekil 2.2: Çok katmanlı algılayıcı: Bir katmanlı derin sinir mimarisi

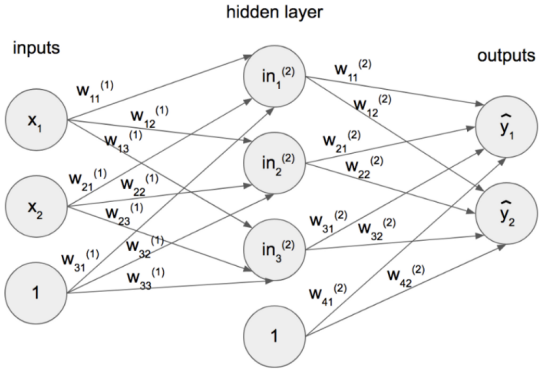
Burda w ağırlıktır ve α güncelleştiren ağırlığın büyüklüğüdür. Hebb'in hayalı biyolojik sistemleri matematiksel olarak modellemektir.

1958'de Rosenblatt,Hebb'in düşüncelerinden yola çıkarak ve **Şekil 2.2:** gibi çok katmanlı algılayıcı geliştirdi. Çok katmanlı algılayıcının derin öğrenme (DL) araştırmalarına çok etkisi bulunmaktadır. Onu bayas toplanmasıyla X giriş vektörünün ve ona denk gelen w ağırlık matrisinin noktasal çarpımı olarak tanımlanır. Bu çarpım σ aktivasyon fonksiyonuna katkılanır.

$$z(x) = w.x + b \quad where \quad \sigma(z) = \begin{cases} 1 & \text{if } z(x) \geq 0 \\ 0 & otherwise \end{cases} \tag{2.2}$$

Sinir ağların terminolojisi kullanarak,çok katmanlı algılayıcı bir katlı önbeslemeli sinir ağ (Feed Forward Neural Networks:FFNN) tarif edilebilir. Çok katmanlı algılayıcının sadece neuron adı olan bir hesap birimi içerir ,aktivasyon fonksiyonu olarak basamak fonksiyonu kullanılır. Minsky [30] çok katmanlı algılayıcılar *XOR* gibi bazı lojik operasyonlar temsil etmeye bildiğini keşfetmiş ve DL için yıllarca araştırma konusu olmuştur.

Şekil 2.3: deki gösterilen derin sinir ağın (DNN) mimarları farklı uygulamalarda (örneğin kalıbı tanıma ve NLP) ileri sürdüğü SVM gibi MÖ yöntemleri bütün dünyanın dikkatini cekmistir. DNN hesaplama birimlerden oluşur ,bunlara sinir hücreleri denir. Genelde DNN'lar üç kattan oluşur: giriş yada 0. Kat (özel aksam giriş vektöründen oluşur),gizli kat (sinir hücrelerini kapsar) ve uygulamalara göre sinir ağın yanıtından oluşan çıkış kat. Gizli kat,giriş katını çıkış katına yönlendirir. FFNN adına göre sadece sinyaların girişten çıkışa



Şekil 2.3: Tek gizli katmanlı ileriye doğru Sinir Mimarisi (Feed Forward Neural Network with Single Hidden Layer)

geçmesini izin verir.

2.3. İLERİBESLEMELİ DERİN SİNİR AĞIN MİMARİSİ (FEED FORWARD DEEP NEURAL NETWORK ARCHITECTURES)

Çok katmanlı algılayıcının çıkışı başkasına giriş olarak katılabilir. Bunun sonucu çok katmanlı algılayıcılardan oluşan derin bilgili yığın meydana gelir. Benko [8] sigmoid aktivasiyon fonksiyonuyle iki yada fazla çok katmanlı algılayıcını yığın yaparak bütün katlarında verilen sinir hücreler herhangi bir fonksiyonun gösterebileceğini ispat etmiştir. Bunun sonucunda denk farklı katlı çok katmanlı algılayıcılar (MLP) ortaya çıkmıştır. MLP'inin bu genelleştirmesine evrensel yaklaştırma özelliği denir. Hornik bunu [21] aktivasyon fonksiyonları olarak tanımlanmıştır. Lineer olmayan,türevlenen aktivasyon fonksiyonları ve geri yayılma algoritmaları [49] tarafından kullanması sonucu çok katmanlı algılayıcıların yığınlanabilmesidir. Geri yayılma gradyan iniş yardımıyla ağırlıkların derin sinir ağına geri uyumlaması için kullanılan bir yöntemdir. **Şekil 2.3:** gösterildiği gibi MLPlar FFNN olarak tanımlanır. Sinir ağların eğitim sağlamlığını geliştirmek için bir kaç teknik ortaya çıkmıştır. Bunlar Duchi [13] tarafından geliştirilen AdaGrad yöntemi,sinir ağının ağırlıkları uyumlaması için gradyan inişi denk bir yöntem. Srivastava [43] Dropout adıyla bir teknik geliştirmiş,bu teknik aşırı uyma sorunun çözülmesine yardım etmiş ve sonuç olarak performans artmıştır.

$$\hat{y_1} = w_{11}^2 in_1^2 + w_{21}^2 in_2^2 + w_{31}^2 in_3^2 + w_{41}^2 * 1 \tag{2.3}$$

$$\hat{y_1} = w_{12}^2 in_1^2 + w_{22}^2 in_2^2 + w_{32}^2 in_3^2 + w_{42}^2 * 1 \tag{2.4}$$

$$in_1^2 = w_{11}^1 x_1 + w_{21}^1 x_2 + w_{31}^1 * 1 \tag{2.5}$$

$$in_2^2 = w_{12}^1 x_1 + w_{22}^1 x_2 + w_{31}^1 * 1 \tag{2.6}$$

$$in_3^2 = w_{13}^1 x_1 + w_{23}^1 x_2 + w_{33}^1 * 1 \tag{2.7}$$

Denklemleri (2.3),(2.4),(2.5),(2.6),(2.7) FFNN **Şekil 2.3:**'daki grafik gösterimini matematik formül cinsinden gösterir.

2.4. EVRİŞİMLİ SİNİR AĞLARI (CONVOLUTIONAL NEURAL NETWORKS (CNNS))

CNNs kavramıyla karşılaştığımızda ilk aklımıza gelen bilgisayarlı görü olur. Sürücüsüz otomobiller ve fotoğraflar katlamalarda olan fotoğraf sınıflandırmasında CNNs büyük ilerlemelerden sayılır.konvelasyonu iyice anlamak için foto piksellerinden oluşan bir matrisin üzerinde uygulanan sürgülü pencere fonksiyonu olarak düşünülmelidir. CNNde Sürgülü pencere filtre,çekirdek yada özellik detector konvolüsyonu bulmak için bütün elemanlarda orjinal matris değerleri çekirdekle çarpıp kaydırması ile toplanır. CNNs adı matematik ve sinyal işlemdeki konvolüsyon operasyondan gelmektedir. CNNs,verileri çeşitli yollarda şişirmek (inflate) için farklı numaralandırmış filtrelerin karışımını kullanır,bunun sonucunda farklı verilerin eş zamanlı test edilmesini sağlar [25]. CNNs çok katlı konvolüsyonlar,bazı lineer olmayan aktivasyon fonksiyonlarıyla birlikte tanımlanabilen tanh,ReLU gibi çıktıları üzerinde uygulanmıştır.

NLP için CNNs foto pikseller yerine giriş olarak kullanılmaktadır. CNN ağına cümleler yada dokümanlar girdi olarak sağlanır. Matrisin her bir satırı vektör şeklinde temsil edilir (kelime yada harf),buna kelime gömülmeler denir. Bu gömülmeler küçük boyut olan temsiller WordVec [28] and GloVe [36] gibi,onlar 3. bölümde detaylı bir şekilde anlatılacaktır ama bu temsiller kelimelerin one-hot kodlaması olabilir. Örneğin 10 kelimeli cümlede eğer 100 boyutsal kelimenin temsili kullanılırsa,10 * 100 olan giriş matrisi elde

edilmiş olur. CNNs'nin en uygun olduğu uygulamalar,mesaj sağanağını belirleme içeren sınıflandırma,konu kategorileme ve duygu analizi gibi görevlerdir. Bununla birlikte Xu [59] CNNs kısa metin kümelenme sorunu için de kullanılabileceğini ifade eder.

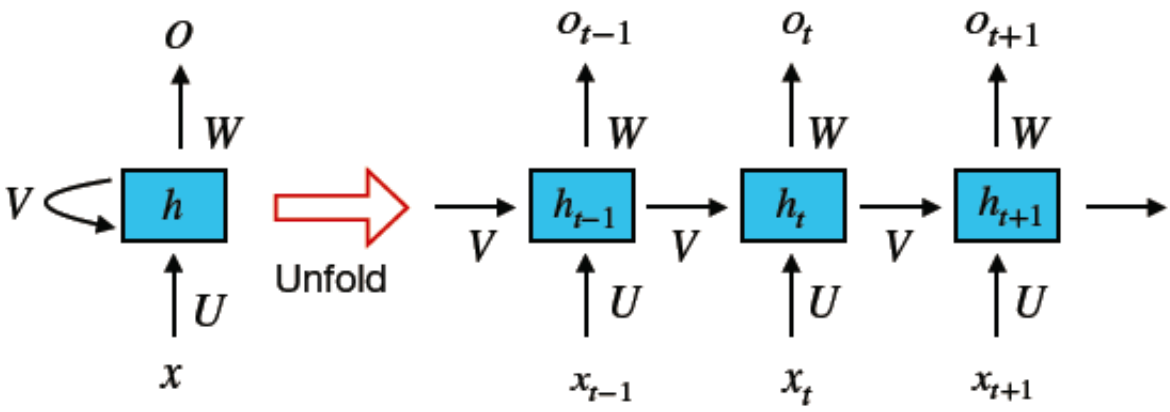
2.5. YİNELEMELİ YAPAY SİNİR AĞLARI (RECURRENT NEURAL NETWORKS (RNNS))

Sinir ağlarının bir başka biçimidir. Jordan [22] grafında yönlendirilen daire olan NNs'ı recurrent olarak sınıflandırmış olup,CNNs bunun tam zıttıdır. Jordan ağı ilk RNN modellerinden biridir. Daha sonra bu [15] tarafından Elman ağı olarak formülize edilmiştir. Bu sunucu modelinde saklanılan katlar üzerinde bilgi besleme konusunda daha esnek hale gelmiştir. Halkalı grafların katması sayısından geri yayılma algoritmanın farklığını keşfedilmesini bir gerek duyurmuştur. Ona göre geri yayılma RNN algoritma (backpropagation through time (BPTT)) meydana gelir.

RNNs NLP görevlerinde çok iyi sonuçlar sağlar. RNNs dildeki sıralı ardışık monte edilmiş birimlerin varlığını işleyebilir ve ele geçirebilir. Bu birimler cümle,kelime yada harf olabilir. RNNs'de hesaplama fonksiyonu mevcut girişi önceki hesaplamayla bağlıdır. **Şekil 2.4:** gösterildiği gibi RNNs'da yinelemeli birimi sürekli olarak giriş metin serisinin düzeltilmiş hacımı olan temsil tarafından katılmaktadır. RNNs'nin önemli bir avantajı önceki hesaplamanın sonucunu kayit edip ve sürmekte olan hesaplamaya katabilmesidir. Bu özellik RNNs giriş seri bağlı olmayan bağlamın modellenmesi için uygun hale getirir. NLP'de RNNs'nın uygulama alanları,makine tercümesi ve dil modellemedir. RNN ve CNN modellerini karşılaştırdığımızda,RNNs'nin bağlam içinde kullanılmış NLP görevleri icin daha başarılı olduğunu söyleyebiliriz. Basit RNNs'ın sorunu yok etme gradyan sorunudur. Bu problem eğitim süresinde parametrelerin öğrenilebilmesini zorlaştırır. Diğer RNNs türleri,uzun kısa vadeli bellek (LSTM) ,geçitli tekrarlayan ünite (GRUs) ve kalıntı ağlar (ResNets)'dir.

2.6. AKTİVASYON FUNKSİYONU

Aktivasyon fonksiyonları,bir sinirin (hesaplama ünitesi) etkinleştirilip etkinleştirilmeyeceğine karar veren YSA'ların temel özelliğidir. Aktivasyon



Şekil 2.4: Yinelemeli yapay sinir ağları

fonksiyonları,bilgi işlem ünitesinin aldığı bilginin göz ardı edilip edilmeyeceğine karar verir. Aktivasyon fonksiyonları,sonraki katmanlara beslenen giriş sinyalinin doğrusal olmayan dönüşümleridir.

$$Y = Activation(\sum(weights * inputs + bias) \tag{2.8}$$

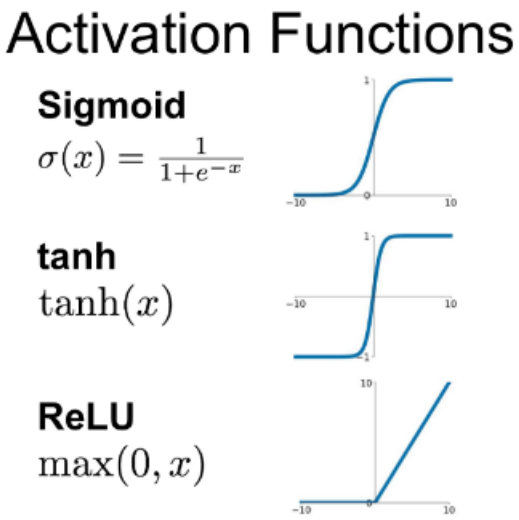
Ağırlıklar ve önyargı olmadan,aktivasyon funksiyonu sadece doğrusal bir dönüşüm yapar. Doğrusal dönüşüm veya doğrusal denklemin çözülmesi çok kolaydır ancak çok daha yüksek karmaşık sorunları çözme kabiliyeti sınırlıdır. Dolayısıyla,aktivasyon fonksiyonu olmayan bir YSA,sadece doğrusal bir regresyon modelinden ibarettir. Bunun nedeni,YSA'lerin karmaşık dönüşümleri öğrenmesini mümkün kılan bu aktivasyon fonksiyonlarıdır. Popüler aktivasyon fonksiyonlarından bazıları şunlardır.

2.6.1. Sigmoid (Lojistik) Funksiyonu

Bu formun bir funksiyonudur:

$$\sigma(z) = \frac{1}{1+\exp^{-z}} \tag{2.9}$$

Sigmoid (Lojistik) fonksiyonu,pürüzsüz ve ayırt edilebilir bir fonksiyondur. İşlev,**Şekil 2.5:**'te gösterildiği gibi bir S şeklinde $0-1$ aralığındadır.



Şekil 2.5: Aktivasyon Funksiyonu

2.6.2. Tanh Funksiyonu

Bu sadece aşağıdaki gibi tanımlandığı gibi sigmoid fonksiyonunun ölçeklendirilmiş bir versiyonudur:

$$\sigma(z) = \frac{\exp^z - \exp^{-z}}{\exp^z + \exp^{-z}} \quad = \quad 2\sigma(2z) - 1 \tag{2.10}$$

Bu aktivasyon funksiyonu,**Şekil 2.5:**'te gösterildiği gibi sigmoid'e benzer şekilde orijin üzerinde simetriktir,ancak -1 ile 1 arasındadır. Tanh sigmoid fonksiyonundan daha dik bir gradyana sahip. Nasıl bir seçim yapılacağı seçimi problem alanına bağlıdır.

2.6.3. ReLU Funksiyonu

Bu en yaygın kullanılan aktivasyon funksiyonudur. **Şekil 2.5:**'te gösterildiği gibi Doğrultulmuş Doğrusal Birim olarak adlandırılır. Olarak tanımlanır:

$$\sigma(z) = max(0, z) \tag{2.11}$$

ReLU kullanmanın en büyük avantajlarından biri,yalnızca girdi pozitifse aktif hale gelmesi

ve böylece negatif değerleri göz ardı ederek YSA'lerin verimli ve kolay hesaplanmasını sağlamasıdır.

2.6.4. Softmax Funksiyonu

2 yerine çoklu sınıflandırma problemi için kullanılan bir tür sigmoid fonksiyondur. Bu işlev,0 ve 1 arasındaki her bir sınıfın çıkışlarını küçültür. Softmax funksiyonu her sınıfın çıkışlarını 0 ile 1 arasında sıkıştırır. Bu,girdilerin belirli bir sınıfta olma olasılık dağılımını verir. Şu şekilde tanımlanmıştır

$$\sigma(z)_i = \frac{exp^{z_i}}{\sum_{k=1}^{K} \exp^{z_k}} \quad for\ i = 1, ..., K \tag{2.12}$$

Diyelim ki çıktılarımız [1.2 ,0.9 ,0.75] olsun,Softmax uyguladıktan sonra bunlar [0.42,0.31,0.27] şekline alır. Şimdi bu olasılıklar her bir çıktı sınıfı için olasılık olarak kullanılabilir. Softmax genellikle NN'nin son çıkış katmanında,her çıkışın sınıfını belirleme olasılıklarını belirlemek için kullanılır.

Etkinleştirme fonksiyonlarını kullanmanın ne zaman iyi veya kötü olduğunu belirleyen bir kural yoktur. Sorunun içeriğine bağlı olarak,yakınsaklıkta daha hızlı olan veya kolay olanı seçmeniz önerilir. Sigmoid genellikle sınıflayıcılarda iyi çalışır. ReLU sadece gizli katmanlarda kullanılmalıdır.

3. KELİME KALIPLAMA

Doğal Dil Sşleme (NLP)'de,(Deep Neural Networks)DNN'lere beslediğimiz girdi genellikle kelimeler,cümleler ,belgeler veya harflerden oluşur. Ancak bu kavram ya da model bilgisayarlar için uygun değildir ve bu metin girişini bir yere eşlemek için bir sayısal gösterimler gerekebilir. Bu tezde temel girdi bileşeni hadislerdir. Bu bölümde en son teknolojiye sahip kelime kalıplama yöntemleri araştırılarak karşılaştırılmıştır.

Girişin en temel gösterimi bir sıcak kodlamadır. Korpusun ön işlenmesinden sonra,tüm temel birimler (kelimeler) bir kelime setinde sıralanır ve bu kelime setindeki her kelimeye indeksi atanır. Kelime haznesindeki bu indeksler rastgeledir ve bu nedenle gerçek anlamı veya bilgiyi yakalamaz.

En gelişmiş denetimsiz öğrenme algoritmaları,sözcüklerin vektörel gömmelerindeki sözdizimsel ve anlamsal ilişkileri yakalamaya işaret eder. Söz konusu iki kelimenin gömülü iki vektörünün benzerliği,cümleler,cisimler (hadisler) arasındaki benzerlik kosinüs benzerliği açısından hesaplanır.

3.1. KELİME GÖSTERİMLERİ

Bir kelimenin anlamının yanı sıra cümle,paragraf,belge (bizim durumumuzdaki hadisler) gibi daha büyük birimlerin formulünü anlamak Doğal Dil İşleme'nin (NLP) temel arayışıdır. Makinelerin sözcük,cümle ve belgelerin (Hadisler) derinlemesine anlaşılmasını öğrenmelerini sağlamak için,makinelerin kullanabileceği veya üzerinde çalışabileceği kelime ve belgelerin karakterizasyonunu veya gösterimini tanımlamamız gerekir. Bu karakterizasyon veya derin temsili öğrenme terminolojisine "kelime gösterimi veya kelime gömme" denir.

Kelime veya belgelerin bağlam içinde mümkün olduğunca fazla bilgiyi kodlayacak şekilde temsil edilmesi NLP topluluğunun temel zorluklarından biridir. Bengio [3],bir sinir dili modeli eğitimi bağlamında,gömme kelimesini bir araya getiren ilk kişidir. Lobert [7],kelime gömmelerinin / temsillerinin sonraki işlemler için yararlı olduğunu ve harika aday

çözümleri olduğunu ortaya koydu. Makine Öğrenimi (MÖ) alanındaki son gelişmelerden dolayı,giderek daha da karmaşık olan bu basit modellerin bir sonucu olarak,daha karmaşık MÖ modelleri,daha büyük veri setleri üzerinde eğitilmektedir.

Kelime temsilleri üzerine devam etmekte olan araştırma,bu yeni kelime gömülmeleri tarafından üstesinden gelinmiştir. En çok beğeni toplayan,Google haber veri kümesinde [28] ve diğerlerinin arasında önceden hesaplanmış kelime gömmelerinden oluşan Word2Vec paketinin dağılımına örnek olabilir.

GloVe,Common Crawl veri kümesi [36],Doc2Vec [29],Vikipedi veri kümesi [4],ELMO eğitimli fastText ile yaklaşık 30 milyon cümle hakkında eğitim almıştır. Peter [37] ve sonuncusu,BooksCorpus (800 milyon kelime) 2015) ve İngilizce Wikipedia (2.500 milyon kelime) veri kümesinin bir araya getirilmesi konusunda eğitilmiş olan BERT gömülümüdür [10].

Neoterik katkılar,bu deneylerin NLP görevlerinde ortaya konmasının önemini ve çeşitli deneysel çalışmalardaki izlerini vurguladı. Gömülmeleri öğrenme ya kelimenin bağlamından öngörüsünü ya da kelimenin bağlamındaki bağlamı optimize ederek yapılır [4],[36],[29]. Bu,dilbilimci Firth'ün yaklaşımına dayanmaktadır: *"Şirketten bir kelime bileceksiniz o tutar"* [16].

Derin sinir ağı modellerinin eğitim için büyük miktarda veriye sahip olması bir zorunluluk haline gelmiştir,bunların hepsi kelime gömme için eğitim verisinin miktarı devasa olduğu için kullanılabilir. GloVe [36] gömme modelinin örneğini ele alalım,taranan yaklaşık 200 terabayt ham metin içeren Ortak Tarama veri kümesi üzerinde eğitilir.Bağlamı temel alan modelin eğitimi,anlambilim (anlam) yanı sıra sözdizimini yakalayan kelime temsilinde gelişir.

Word2Vec veya GloVe gibi kelimelerin gömülmesi,cümlelerin gömülmesi için onları eğiterek daha da geliştirilebilir [50]. Ortalama alma yoluyla kompozisyon için bu gömmeler anlamsal metinsel benzerlik görevleri için belirgin gelişmeler göstermiştir.

FastText'in gömülmesi [4],her cümle ve her cümle içindeki n-gram gösterimlerini öğrenerek Word2Vec üzerine kuruludur. Her eğitim adımında temsillerin değerlerinin bir vektörde ortalaması alınır. Bu,kelime oluşumu alt kelime bilgisini yakalamak için yardımcı olur. FastText vektör gösterimlerinin Word2Vec gösterimlerinden daha doğru

olduğu gösterilmiştir. ELMO modeli,göründüğü bağlamına dayalı olarak bir kelimenin gömülmelerini oluşturur,böylece aynı kelimenin farklı bağlamlarda farklı vektör temsillerini oluşturur [37].

Son zamanlarda Google yapay zeka dilinden araştırmacılar,BERT (Transformers'ın Çift Yönlü Kodlayıcı Sunumları) adında yeni bir gömülü ön eğitimli modeli tanıttı; bu,NLP topluluğunda Doğal Dil Çıkarımı (NLI) gibi çeşitli NLP görevlerinde en son sonuçları göstererek heyecanlanmasına neden oldu. BERT'in yenilemesinde ana husus,dil modellemesine dikkat modeli olan iki yönlü trafo eğitimi uygulamasıdır. BERT modeli,bir korpusa soldan sağa veya birleşik sağdan sola ve sağdan sola antrenmanlara bakan önceki gömme model yaklaşımlarının aksine farklı bir korpusa bakar. BERT modeli,kelimelerin tüm metin dizisini bir kerede okur,böylece iki yönlü hale getirir. İki yönlü tabir yerine yönsüz olarak adlandırmak uygun olur. Bu özel özellik,BERT modelinin,yakındaki çevre kelimelere dayanarak verilen metin kelimenin içeriğini öğrenmesini sağlar.

3.2. BAĞLAMSALLAŞTIRMA

Gömmeler,genellikle bağlamdaki bitişik kelimeler dikkate alınarak öğrenilir. Grice [19] 'nin teorisi,dilbilimciler arasında yapılan referans çalışmasıdır. Bağlamsallığın anlambilim açısından önemi,bağlamın sadece 'siz','ben','burası' gibi endeksik kelimelerin ve bağlam duyarlı yapıları olan zamanların anlamları için anlamlı olduğunu varsaydığı için Grice tarafından marjinalleştirilir. Ancak [40],[46],[41] korpus içerisindeki bağlamın anlamsal ilişkisinin olduğunu ileri sürmektedir. Bir örnek verilirse: " iki ya da daha fazla kişi bir toplantı için gidiyorsa ya da "Ben zaten oradayım",eğer kendisi ile randevu almış olan birine yönelikse,"Ben kapıdayım","Haydi çabuk ol!" olarak anlaşılabilir. Bağlam,gerçek dünya bağlamı,çevresi ve bir kelimenin veya ifadenin hangi amaçlarla yapıldığı veya yakın ifadeler açısından dilsel bağlam olarak tanımlanabilir.

Bağlamın daha sonra açıklanması anlam kazanırsa ve NLP'deki kelimelerin vektör ifadelerinin ortaya çıkmasından bu yana NLP'deki gelişmeyi dikkate alırsa,bu gelişme kavramlar için argüman olarak kullanılabilir [46],[41].

3.3. GÜNCEL OLAN EN İYİ DURUM POPÜLER KELİME GÖMME YÖNTEMLERİ

Doğrudan denetlenmemiş DNN'lerde ilerlemeden sonra,kelime gömme yöntemleri popüler hale geldi. Sözcük dağarcığından,daha küçük boyutlu $\mathbb{R}$ ile gerçek değerli bir vektör uzayına,kelime başına bir boyuta sahip matematiksel alandır. Gömme,makine çevirisi [45],duyarlılık analizleri [12],doküman kümeleme,otomatik metin özetleme [60],soru cevaplama [44],belge alımı [17] vb olmak üzere çeşitli NLP uygulama alanları için mevcut DNN'lerin vazgeçilmez bir parçasıdır. NLP'nin DNN'lerinin birçoğu,LSA [9],kahverengi kümeler [5] ve diğerleri gibi geleneksel dağıtım özelliklerini başarıyla değiştirmiştir.

3.3.1. Word2Vec

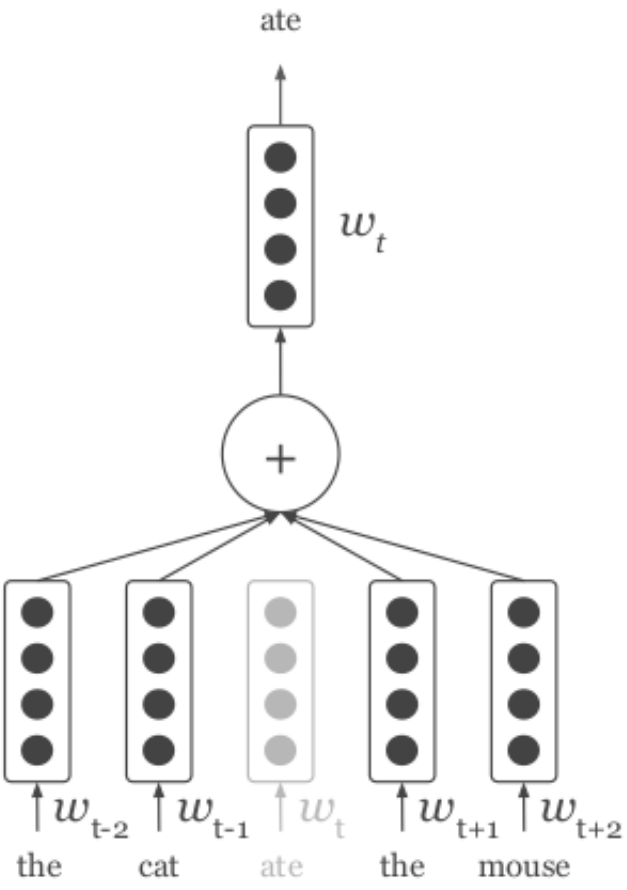
En çok tercih edilen gömme temsil modeli,popülaritenin yükselişinde haber veren ve birkaç yüzlerce makaleyle sonuçlanan modeldir. Word2Vec modeli 2013 yılında tanıtıldı [28]. Bu,doğrusal olmayan tek bir gizli katmandan oluşan,sığ bir derin öğrenme modelidir. [28] büyük bir metin kurumundan sözcük temsillerini öğrenmek için iki yeni denetimsiz mimariyi önermiştir. İkisinden birincisi,**Şekil 3.1:**'de gösterildiği gibi CBOW modeli olarak adlandırılır. Continuous bag-of-words model (CBOW) modeli,"yedik" sözcüğünü,belirli bir pencereyi kullanarak komşu bağlam sözcük vektörlerinin toplamından tahmin etmeye çalışır.

İkincisi,**Şekil 3.2:**'de gösterildiği gibi atlama gram modeli olarak adlandırılır. Bu model önceki CBOW modelinin tam tersidir,verilen kelimeden bağlamı tahmin etmeye çalışır. Önceden eğitilmiş Word2Vec gömmeleri,atlama gram mimarisi kullanılarak eğitilmiştir.

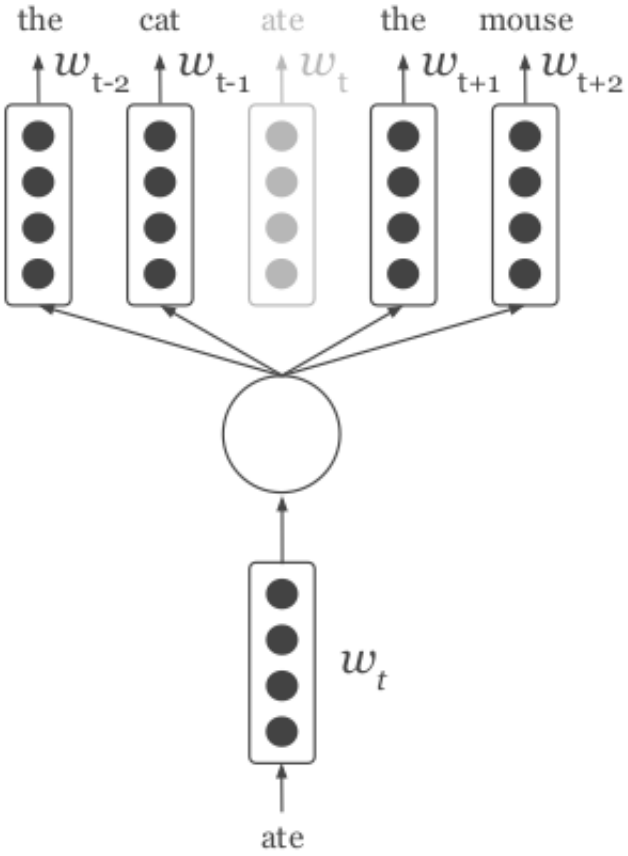
Atla gram mimarisindeki her kelimeye,M'nin pencere boyutunda bulunan (w) sözcüğünün yakınında görünen bağlam kelimelerini (c) tahmin etmek için kullanılan hem hedef (v_w) hem de bağlam vektörleri (u_w) atanır. Olasılık dağılımı,softmax fonksiyonu adı verilen aktivasyon olarak ifade edilir.

$$p(w/c) = \frac{exp^{u_c^T v_w}}{\sum_{i=1}^{n} \exp^{u_i^T v_w}} \tag{3.1}$$

Eğitimi hızlandırmak için,negatif örnekleme veya hiyerarşik softmax gibi yöntemler kullanılabilir [29].



Şekil 3.1: CBOW Modeli. Cümle :Kedi fareyi yedi. CBOW modeli,sol bağlamdaki "kedi" ve sağdaki "fare" kelimesini tahmin etmeye çalışır



Şekil 3.2: Skip-gram Modeli. Burada model sol bağlamı tahmin etmeye çalışır ’kedi’ ve sağ ’farenin’ merkez kelimesinden ’yedik’

Bağlam penceresinin koşullu olasılığı,bağlamdaki her kelimenin koşullu olasılıklarının bir ürünüdür.

$$p(w_{-M},...w_M/c) = \prod_{m=-M,m\neq 0}^{M} p(w_m/c) \tag{3.2}$$

Gömme kelimesini bulmak için,3.2,3.3 denklemlerini kullanarak tüm metin korpusunun log olasılığını maksimize ediyoruz.

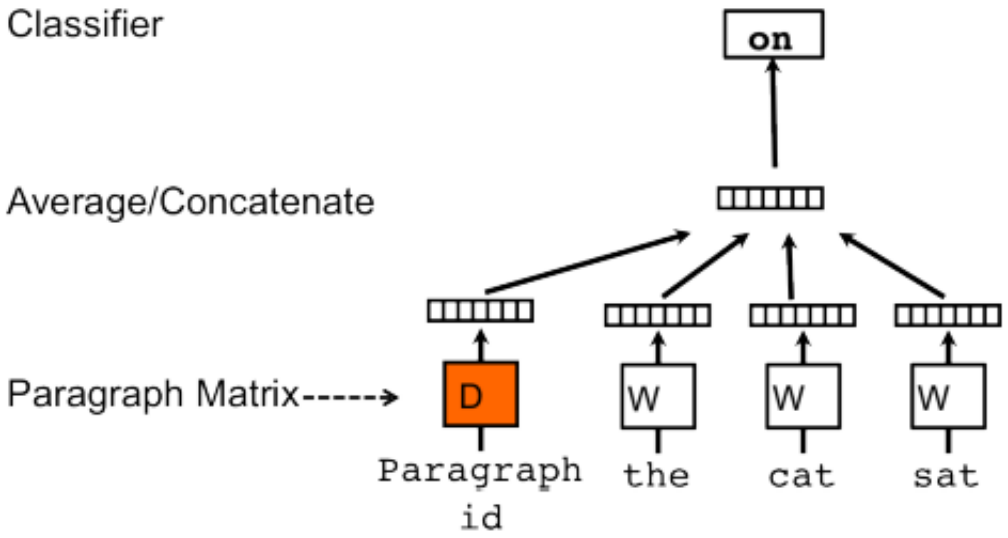
$$\frac{1}{T}\sum_{i=1}^{T}\sum_{m=-M,m\neq 0}^{M} \log p(w_{i+1}/w_i) \tag{3.3}$$

Önceden eğitilmiş kelime gömme [28] tarafından yapılan çalışmalar ile yayınlandı. Bu embeddngs,Google'ın 100 milyar tokenlik İngilizce haber makalesi büyüklüğünde 300 boyutunda İngilizce haber makaleleri üzerine eğitildi. Bu tezde,zaten halka açık olan önceden eğitilmiş gömülerden faydalanıyoruz.

3.3.2. Doc2Vec

Doc2Vec modelinin temel amacı,belgelerin sayısal temsillerini (uzunluğuna bakılmaksızın) oluşturmaktır. Mikolov [29] fikri basit ama çok akıllıca bir yaklaşım olarak anılmaktadır.Word2Vec mimarisini paragraf için ek bir vektörle kullandılar. **Şekil 3.3:**'de örnek verilmiştir.

Doc2Vec'te (ayrıca Paragraf Vektörü - Dağıtılmış Bellek PV-DM de denir),her bir paragraf,matris (D) sütunu ile gösterilen benzersiz vektör gösterimi ile eşleştirilir ve her bir kelime,matris ile temsil edilen W'deki sütun benzersiz vektörüyle temsil edilir [29]. Bu vektörlerin daha sonra içeriğe dayalı bir sonraki kelimeyi tahmin etmek için ortalaması alınır. Paragraf belirtecini,aynı funksiyonu gören başka bir kelime olarak düşünülürse hafıza elemanı,bağlamdaki eksik bilgiyi hatırlar. Bu bağlam vektörleri sabit uzunluktadır ve aynı paragrafta üretilen,ancak paragraflar boyunca üretilen tüm bağlamlarda paylaşılır. Bununla birlikte,vektör matrisi W kelimesi,i paragrafları arasında korunmaktadır. Örneğin,"peygamber" için vektör,tüm paragraflarda aynıdır. Bu mimarileri eğitmek için geri yayılım ile birlikte Stokastik gradyan inişi kullanılmıştır.



Şekil 3.3: Paragraf Vektörünün Dağıtılmış Bellek sürümü

3.3.3. FastText

Bu gömme modeli,skipgram mimarisinin bir uzantısı olan [4] tarafından tanıtıldı. Bu modelde,tekil değer ayrıştırması (SVD) yoluyla 4 gram karakter gösterimini öğrenen [39] 'den esinlenen fikir,n-gram karakterli çanta kelimelerden ziyade atomik olarak kabul edildi. Karakter ngramlarının çantaları için bir örnek düşünün. $N = 3$ olan " lions " sözcüğü karakter ile gösterilecektir n-gram: $< li, lio, ion, ons, ns >$ ve ek $< lions >$ dizisi ile.

Bu yöntemin en büyük avantajı,kelimeler arasında anlamların aktarılmasının mümkün hale gelmesidir; bu,yeni kelimelerin yorumlanmasına yol açan,şimdi n-gramların önceden öğrenilmiş olan gömmelerinden tahmin edilebilir. Uygulama morfolojide açıktır. Eğitim verilerinde "lion" kelimesinin bulunduğunu,ancak *"lionesque"* kelimesinin olmadığını düşünün. Karakter çantalarının yardımıyla $< esque >$ ekinin ve $< lion >$ kök kelimesinin anlamının bu şekilde *"lionesque"* in anlamını tahmin edebileceğini biliyoruz.

Eğitim,aşağıdakileri en aza indirgemekve *negatif örneklemeyle* gram atlamak için gerçekleştirilir:

$$\sum_{i=1}^{T}\Big(\sum_{m=-M,m\neq 0}^{M} l(s(w_i,w_m)) + \sum_{n\in N} l(-s(w_i,w_m))\Big) \tag{3.4}$$

$l(x) = \log(1 + e^{-x})$ and w_n negatif örneğe karşılık gelir. s funksiyonu,karakterdeki n-gram karakterinin çantasındaki toplam olarak gösterilir.

$$s(w,c) = \sum_{g \in G_w} z_g^t v_c \tag{3.5}$$

burada G_w,karşılık gelen w ve z kelimesi için tüm n-gramların koleksiyonunu temsil eder,öğrenilen kelime gömmeleridir. Belirli bir kelimenin gömülmelerini elde etmek için,sadece n-gram karakterini eklememiz gerekir.

Vikipedi çöplüklerinde eğitilmiş yazarlar tarafından 294 farklı dil[2] için 300 boyutta önceden eğitilmiş vektörler yayınlandı.

Tezimizde ***Sahih-Buhari***'nin İngilizce versiyonunun veri kümesi üzerinde çalıştığımız için korpus ile ilgili gömüler oluşturmak için gömme modelini ingilizce versiyonda eğitiyoruz.

3.3.4. GloVe

Kelimelerin bu vektör gösterimi,skipgram modelinden esinlenmiştir [36]. Anlamsal bilgi,iki özel sözcüğün birlikte ortaya çıkma olasılıkları oranında gösterildiği şekilde bulunur [36]. Bu yaklaşım,TF-IDF [38] yaklaşımına biraz benziyor olsa da eğitim sırasında bağlamsal bir sözcüğe önem vermektedir. Her bir öğenin skipgram mimarisine benzer şekilde,M'nin bir pencere boyutu içindeki j sözcüğü ile birlikte ortaya çıkma sayısını temsil ettiği devasa matris X. Bu birlikte oluşturma matrisi daha sonra gömülme kelimesini şu şekilde tanımlar:

$$w_i^T w_j + b_j = \log(X_{ij}) \tag{3.6}$$

En uygun gömüleri w_i ve w_j elde etmek için aşağıdaki en düşük kare maliyet funksiyonu en aza indirilmelidir.

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - \log(X_{ij}))^2 \tag{3.7}$$

f,sadece son derece yaygın kelime çiftlerinden öğrenmek yerine,tüm korpustan öğrenmeye yardımcı olan bir ağırlıklandırma funksiyonudr.

[2] The pretrained fastText embeddings can be found at: https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md.

Yazarlar tarafından çalışmaları [3] boyunca çeşitli farklı temsiller yayınlanmıştır. Süslemeler,çeşitli boyutlarda Vikipedi maddeleri olan tweet'lerde eğitildi. Kelime gömmeleri,100 milyardan fazla jetondan oluşan Common Crawl[4] veri kümesinden eğitildi. Common Crawl veri kümesinde eğitilmiş olan gömmeler,kümelenme sorunu için bu tezde kullandığımız,2.2 milyon kelime büyüklüğüne sahip 300 boyutuna sahip gömmelerden oluşur. Bu tez çalışmasında,GloVe gömme modelini gömmeleri oluşturmak için kendi korpusumuza eğitiriz.

[3] The pretrained GloVe embeddings can be found at:https://nlp.stanford.edu/projects/glove
[4] Common Crawl dataset can be found at:http://commoncrawl.org

4. METİN KÜMELEMESİ

Bu bölümde,birkaç kümeleme tekniklerini kısaca açıklıyoruz,bunların arasında tez çalışması için kalıplamaların ile birlikte birkaç teknikleri araştırdı.Kümeleme,bir grup kümeyi,aynı gruptaki varlıkların (küme adı verilen),diğer gruplardakilere (kümeler) göre daha benzer (bir anlamda) olacak şekilde gruplandırmasıdır [53]. Kümelenme Çok değişkenli veri madenciliğindeki geleneksel bir metodoloji olup,söz konusu kurumda doğuştan gelen kalıpları bulmak ve araştırmak için tasarlanmıştır,aynı kümedeki belgeler benzer özellikleri sahiptir ve diğer kümelerdeki varlıklardan farklıdır. Kümeleme teknikleri çeşitli alanlarda uygulanır örüntü tanıma,tıbbi araştırma,ekonomi vb. alanlar. Üç boyuta kadar olan görevler için insanlar kolayca kümeleme yapabilir; örneğin iki boyutlu bir haritaya bakarken,biri bölgelerin birbirine ne kadar yakın yerleştirildiğine göre çeşitli farklı bölgeleri otomatik olarak ve kolayca tanıyabilir. Fakat sezgisel değerlendirmelerin varlıkların yorumlanması daha yüksek boyutlara ulaştığında elde edilmesi zordur.

"Küme" kavramını tam olarak tanımlamak zordur. Bu,bu kadar çok kümeleme algoritmasının var olmasının ana nedenlerinden biridir. Bir kümenin tanımı,farklı araştırmacılar farklı kümelenme modeli kullandığından dolayı özelliklerinde önemli ölçüde değişiklik gösterir. Bu küme modellerinin sezgisine sahip olmak,çeşitli algoritmalar arasındaki farkları anlamada çok önemlidir. Küme modellerinden bazıları şunlardır:

1. Bağlantı modelleri: örneğin,hiyerarşik kümeleme,mesafe bağlantısına dayalı modeller geliştirir.
2. Dağılım modelleri: kümeler istatistiksel dağılımlar kullanılarak modellenmiştir,beklenti maksimizasyon algoritması tarafından kullanılan çok değişkenli normal dağılımlar gibi.
3. Yoğunluk modelleri: DBSCAN ve OPTICS kümeleri veri alanındaki bağlı yoğun bölgeler olarak tanımlamaktadır.
4. Sinirsel modeller: En iyi bilinen denetlenmemiş sinir ağı kendi kendini düzenleyen haritadır ve bu modeller genellikle yukarıdaki modellerin birine veya daha fazlasına benzer olarak nitelendirilebilir ve sinir ağları bir Asıl Bileşen Analizi (ABA) veya

Bağımsız Bileşen Analizi biçimini uyguladığında alt uzay modelleri de dahil olur [53].

4.1. KÜMELEME ALGORİTMALARI

Kümeleme,veri noktalarının gruplandırılmasını içerir(bizim durumumuzda hadisler). Bir takım Hadis dokümanı verildiğinde,her bir dokümanı belirli bir gruba sınıflandırmak için kümeleme tekniklerini kullanabiliriz. Kümeleme denetlenmemiş öğrenme tekniğidir ve doğal dil işleme (NLP)'de kullanılan en yaygın yöntemlerden biridir. Bir sonraki bölümde,bu tez çalışmasında kullanılan bazı kümeleme algoritmaları kısaca tarif edeceğiz.

4.1.1. K Ortalamalar Kümeleme

K-ortalama kümeleme yöntemi,denetlenmemiş zor bir kümeleme yöntemidir. Bu,N varlıklarını / belgelerini (Hadisleri) $d_1, d_2, ..., d_N$ tam olarak K'nin önceden belirlenmiş kümelerine $C_1, C_2, ..., C_K$ ayırır. Optimal K küme sayısı önceden bilinmemektedir,korpustan çıkarılmalıdır [27]. K-ortalama kümeleme algoritmasının amacı,Denklem 4.1'de gösterilen kare hatasını en aza indirmektir veya küme içi toplam varyans.

$$J = \sum_{i=1}^{K} \sum_{j=1}^{N} \left\| d_j^i - K_i \right\|^2 \tag{4.1}$$

K-Ortalama algoritması nispeten etkili bir tekniktir. Bununla birlikte,küme sayısını önceden

Algorithm 1 K-Ortalama Algoritması

1: Veri noktalarını önceden tanımlanmış K gruplarına kümeleyin;;
2: Küme merkezleri olarak K rasgele noktaları seçin;;
3: Kullanılan uzaklık funksiyonuna göre veri noktalarını (Hadisler) en yakın kümelerine atayın;
4: Her bir kümedeki tüm veri noktalarının (Hadisler) yeni merkezlerini/ortamalarını hesaplayın;
5: Yakınsama kadar 2,3,4 adımları tekrarlayın.

belirtmesi gerekir ve son kümeler rastgele başlatmaya karşı çok hassastır ve yakınsama garantisi vermez ve sık sık yerel olarak optimum duruma sıkışır. Yakınsama zamanla üssel. K-ortalama kümelemesi sadece küme merkezlerini dikkate alırama şekli değil. Bu,kümelerin küresel olarak simetrik şekillere sahip olduğunu ve kürelerde eşit boyutlarda olduğunu varsaymakla eşdeğerdir,yani tüm boyutların eşit önemde olduğu anlamına gelir. Malessef,en uygun küme sayısını bulmanın bir yolu yoktur. Pratik bir yaklaşım yaklaşımı birden fazla çalışmanın sonuçlarını farklı K ile karşılaştırmak ve önceden belirlenmiş bir kritere göre en

iyisini seçmektir. Genel olarak,K büyükse,muhtemelen hatayı azaltır ancak aşırı uydurma riskini arttırır. Kümeler farklı boyutları olduğunda K-ortalama başarısız olur.

4.2. HİYERARŞİK KÜMELEME

Bu tür algoritmalar küme hiyerarşisi oluşturur. Bir uçta,kümeleme yapısı tüm veri noktalarından oluşurken,diğer ucunda veri noktalarına eşit sayıda küme bulunmaktadır. Bu kümeleme tekniğinde önceden oluşturulmuş kümelenmeleri yeniden düzenlemek mümkün değildir. Başlangıçta,hiyerarşik kümeleme yalnızca sert kümeler içindi,ancak daha sonra yumuşak kümelenme için de kullanılabileceği gösterildi. Hiyerarşik kümeleme,aglomeratif veya bölmeli kümeleme olabilir. Bölücü kümeleme tekniği yukarıdan aşağıya bir yaklaşımdır,tek bir kümeden maksimum küme sayısına başlar. Aglomeratif,aşağıdan yukarıya yaklaşma yaklaşımıdır ve öncekinin tersidir: her gözlem kendi kümesinde başlar ve hiyerarşiye doğru ilerlerken kümelerin grupları tek bir kümede birleştirilir. En yakın küme benzerliği ölçüsüne sahip kümeler,son küme veya önceden tanımlanmış küme sayısı kalana kadar sürekli olarak birleştirilir. Basit aglomeratif kümeleme algoritması aşağıdaki gibidir:

Algorithm 2 Aglomeratif Kümeleme Algoritması

1: Tüm küme çiftleri arasındaki benzerliği hesaplayın,yani ij^{th} girişi i^{th} ile j^{th} küme arasında benzerlik veren benzerlik matrisini hesaplayın.;
2: En benzer (en yakın) iki kümeyi birleştirin.;
3: Yeni küme ile orijinal kümeler arasındaki çift benzerliği yansıtmak için benzerlik matrisini güncelleyin.;
4: Yalnızca son bir tek küme veya önceden tanımlanmış kümeler kalana kadar 2. ve 3. adımları tekrarlayın..

Aglomeratif algoritmalar kullandıkları kümeler arası benzerlik önlemlerine dayanarak sınıflandırılır. Bunların arasında tek bağlantı,grup ortalaması ve tam bağlantı bulunur. Tek bağlantıda,herhangi bir çift veri noktası arasındaki minimum mesafe (kümelerden çizilen) kümeler arasındaki mesafedir. Ortalama bağlantıda ortalama mesafedir ve tam bağlantıda maksimum mesafedir.

4.2.1. BIRCH Algoritması

Hiyerarşileri Kullanarak Dengeli İteratif Azaltma ve Kümeleme anlamına gelen BIRCH,çok büyük veri kümelerinde hiyerarşik kümelemeyi gerçekleştirmek için veya yalnızca tek bir veri taraması ile çok iyi kümeleme çözümleri keşfetme yeteneği nedeniyle akış verileri için

kullanılan denetimsiz bir makine öğrenme algoritmasıdır. Bu teknik,verilerde daha fazla tarama yaparak kümeleme kalitesini artırabilir. BIRCH kümeleme algoritması tarafından elde edilen yüksek verimlilik,büyük veri noktaları kümesini temsil eden daha küçük özet istatistik kümesinin akıllıca kullanılmasıdır. Bu özet istatistiklere,gerçek veri kümesinin bolca değiştirilmesini temsil eden kümeleme amaçları için CFs denir. BIRCH algoritmasına giriş,gerçek değerli vektörler ve istenen sayıda küme K olarak temsil edilen,bir N veri noktası kümesidir [52]. BIRCH kümeleme algoritması 4 aşamada çalışır.

BIRCH kümelenme algoritmasının ilk aşaması,yüksekliği dengeli bir CF ağacı oluşturur,aşağıdaki gibi tanımlanmıştır:

1. Belirli bir N boyutlu veri noktası kümesi için CF,,üçlü $CF = (N, LS, SS)$ olarak tanımlanır. LS ve SS aşağıdaki gibi tanımlanır.

   (a) **Doğrusal Toplam (Linear Sum):LS:**LS Bireysel koordinatların toplamı,kümenin koordinatlarının (konumlarının) ölçüsüdür.

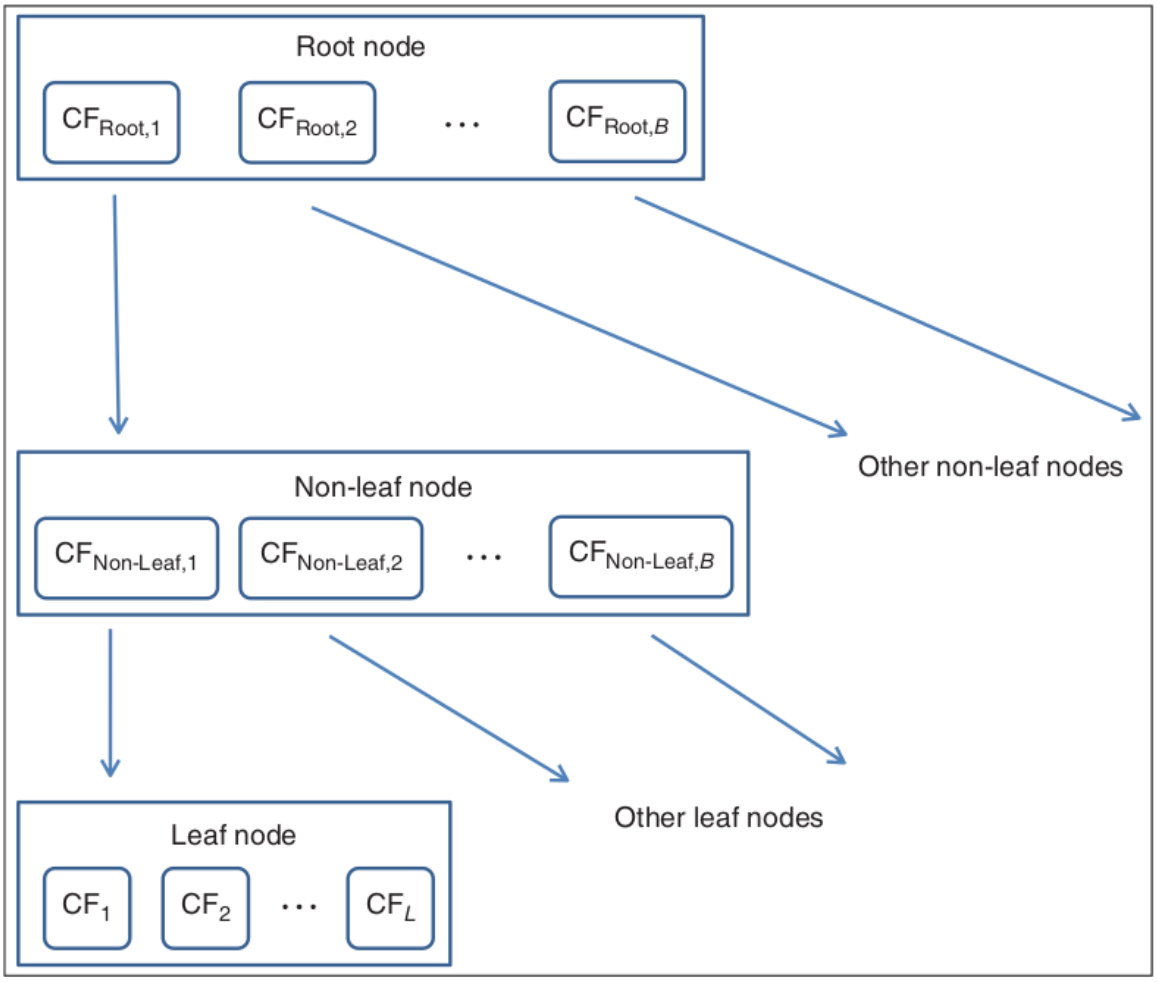
   $$\overrightarrow{LS} = \sum_{i=1}^{N} \overrightarrow{X_i} \tag{4.2}$$

   (b) **Kareler Toplamı(Squared Sum):SS** SS,kümenin yayılmasının ölçüsü olarak tanımlanmaktadır.

   $$\overrightarrow{SS} = \sum_{i=1}^{N} (\overrightarrow{X_i})^2 \tag{4.3}$$

   (c) CF'ler,üç parametreli,yükseklik dengeli bir ağaçta düzenlenir: dallanma faktörü B (yapraksız bir düğüm için izin verilen maksimum çocuk sayısını belirler),eşik T (bir kümenin yaprak düğümündeki yarıçapa bağlı olan üst) ve L (yaprak düğümündeki girişlerin sayısı). CF girişi,yapraksız düğümler ve bir yaprak düğümü için o girişin alt düğümlerindeki CF girişlerinin toplamına eşittir,sadece değeridir. Sıralı kümeleme yaklaşımıyla gerçekleştirilen CF ağacının oluşturulması, bir anda bir veri noktasının bir anda taranması ve mevcut kümeye veya yenisine dahil edilecek belirli bir veri noktasının atanmasına karar verir. Genel CF ağacı **Şekil 4.1:**'de gösterilmektedir.

BIRCH yönteminin orijinal sunumunda bu adım isteğe bağlı olarak işaretlenir. Bu 2.



Şekil 4.1: CFT Genel Yapısı

aşamada BIRCH kümeleme algoritması,ilk *CF* ağacındaki yaprak düğümlerinin tüm girişlerini tarayarak,kalabalık alt kümeleri büyük gruplara ayırarak ve aykırı öğeleri kaldırarak daha küçük bir CF ağacı yeniden yapılandırmaktadır.

BIRCH yönteminin 3. aşamasında,tüm yaprak girişleri varolan bir aglomeratif hiyerarşik kümeleme algoritmasıyla kümelenir (doğrudan *CF* vektörleriyle temsil edilen alt kümelere uygulanır). Bu kullanıcıya esneklik sağlar,kümeler için özelik çapı eşiğini veya küme sayısını belirtmek için. Bu aşamadan sonra elde edilen kümeler kümesi veri noktalarının ana dağılımını yakalar.

Bazı lokalize ve küçük uygunsuzluklar,3. asamasinin bitiminden sonra kalabilir ve daha sonra BIRCH yönteminin son evresinde çözülebilir. Bu 4. aşamada,3. aşamadan üretilen centroidler tohum olarak kullanılır ve veri kümelerini yeni kümeler oluşturmak için veri noktalarının en yakın tohumlarına yeniden dağıtır. Aykırıklar da bu aşamada atılabilir.

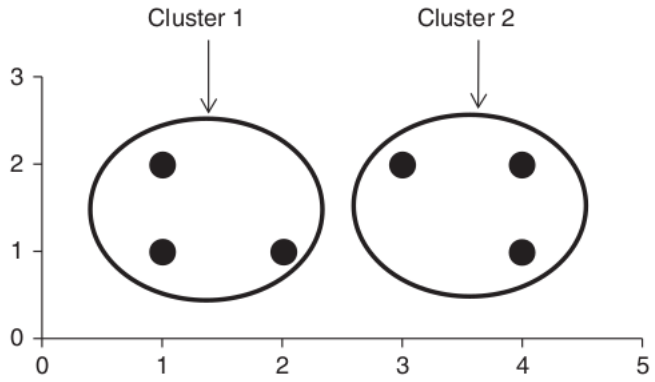
Şekil 4.2:'deki CF hesaplamaları için bir örnek düşünün,sırasıyla küme 1 ve 2'deki veri noktaları sırasıyla $(1,1)$,$(2,1)$,$(1,2)$ & $(3,2)$,$(4,1)$,$(4,2)$.'dir. Küme 1 için Küme özelliği CF_1 içerir:

$$\begin{aligned} CF_1 &= \{3,(1+2+1,1+1+2),(12+22+12,12+12+22)\} \\ &= \{3,(4,4),(6,6)\} \end{aligned} \tag{4.4}$$

Ve küme 2 için CF_2

$$\begin{aligned} CF_2 &= \{3,(3+4+4,2+1+2),(32+42+42,22+12+22)\} \\ &= \{3,(11,5),(41,9)\} \end{aligned} \tag{4.5}$$

2 kümenin CF'lerinin birleştirilmesi,*AdditivityTheorem*'e göre kolayca eklenebilir. Böylece, **Şekil 4.2:**'deki kümeler için,ortaya çıkan *CF*,şöyle görünecektir:m the resulting *CF* would look like:



Şekil 4.2: Kümeler (Clusters)

$$CF_12 = \{3+3, (4+11, 4+5), (6+41, 6+9)\} = \{6, (15,9), (47,15)\} \tag{4.6}$$

4.2.2. Küme Özellikleri ile Hesaplamalar

Sadece $CFs = (N, \overrightarrow{LS}, \overrightarrow{SS})$ olarak verilen değerlerin gerçek bilgisi olmadan yukarıdaki önlemler aşağıdaki gibi hesaplanabilir:

$$Centroid : \overrightarrow{C} = \frac{\sum_{i=1}^{N} \overrightarrow{X_i}}{N} = \frac{\overrightarrow{LS}}{N} \tag{4.7}$$

$$Radius : \sqrt{\frac{\sum_{i=1}^{N} (\overrightarrow{X_i} - \overrightarrow{C})^2}{N}} = \sqrt{\frac{N.\overrightarrow{C}^2 + \overrightarrow{SS} - 2.\overrightarrow{C}.\overrightarrow{LS}}{N}} \tag{4.8}$$

ve $CF_1 = (N_1, \overrightarrow{LS_1}, \overrightarrow{SS_1})$ and $CF_2 = (N_2, \overrightarrow{LS_2}, \overrightarrow{SS_2})$ kümeleri arasındaki ortalama bağlantı mesafesi şöyle hesaplanır:

$$D = \sqrt{\frac{\sum_{i=1}^{N_1} \sum_{j=1}^{N_2} (\overrightarrow{X_i} - \overrightarrow{Y_j})^2}{N_1}.N_2} = \sqrt{\frac{N_1.\overrightarrow{SS_2} + N_2.\overrightarrow{SS_1} - 2.\overrightarrow{LS_1}.\overrightarrow{LS_2}}{N_1.N_2}} \tag{4.9}$$

4.3. SPEKTRAL KÜMELEME

K-ortalama gibi kümelenme algoritmalarının çoğu,verilerin şeklini (radyal temel,spiral vb.) varsaymaktadır ve yerel minimumları bulmak için başlangıçta çoklu yeniden başlatmalar gerektirir. Spektral kümeleme,kümelerin şekilleriyle ilgili varsayımlarda bulunmakla ve başlatma işlemine duyarlı olmadan bu sorunları çözmeye yardımcı olur. Bu teknik,grafik teorisinin gücüne dayanır ve kümeleme yapmadan önce boyutluluk azaltma için benzerlik matrisinin özdeğerlerini kullanır [57]. Algoritma 3 aşamada çalışır:

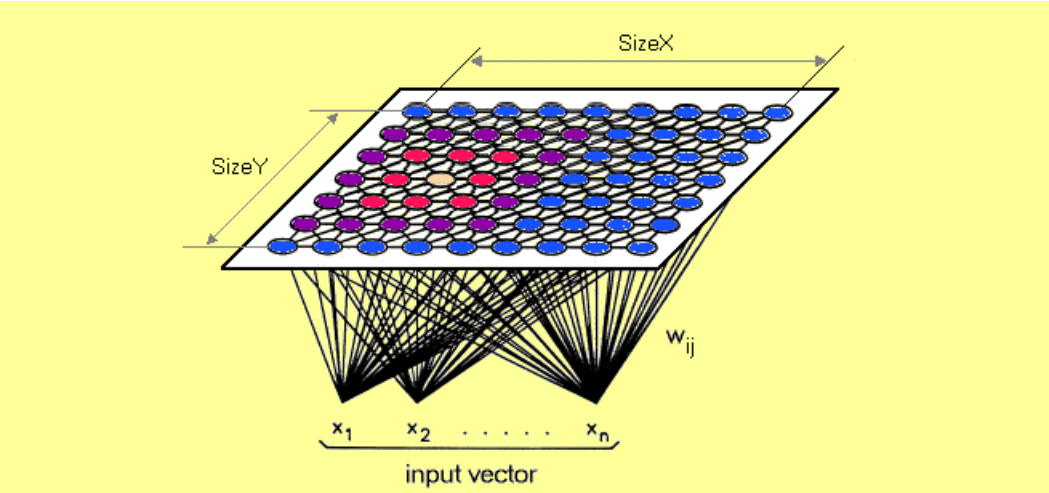
Algorithm 3 Spektral Kümeleme

1: Tüm veri noktaları için benzerlik grafiği (*e.g.*, K Nearest Neighbor grafiği) oluşturun;
2: Kümelerin kategorize edilmesinin daha açık olduğu veri noktalarının Laplacian grafiğinin özvektörlerini kullanarak düşük boyutlu bir spektral yerleşimler oluşturun;
3: Gömme alanını bölmek için K-ortalama gibi herhangi bir klasik kümeleme algoritması uygulayın *i.e.* küme için özvektör seçmek üzere en düşük bir özdeğer kullanın.

4.4. KENDİ KENDİNİ DÜZENLEYEN HARİTA VEYA KOHONEN HARİTALARI

T Kohonen tarafından icat edilen kendi kendini düzenleyen harita (SOM) veya kendi kendini düzenleyen özellik haritası (SOFM),düşük boyutlu (tipik olarak 2D) bir harita olarak bilinen giriş boşluklama örneklerinin isteğe bağlı olarak gösterilmesi için denetimsiz olarak eğitilen bir YSA'lar sınıfıdır. Bu nedenle,boyutluluk azaltma yöntemidir. SOM'lar,rekabetçi düzeltme öğrenimini,hata düzeltme öğrenimi uygulayan (gradyanlı yayılma gibi) diğer YSA'ların tersine uygular. SOM'larda giriş alanının topolojik özelliklerini korumak için bir komşuluk funksiyonu kullanılır [55].

SOM ayrıca bir veri görselleştirme tekniği sağlar,boyutları **Şekil 4.3:**'te gösterildiği gibi bir haritaya indirgeyerek yüksek boyutlu verileri anlamaya yardımcı olur. Kümeleme kavramı benzer veri nesnelerini birlikte gruplayarak SOM ile de temsil edilebilir,böylece SOM kümeleri yakalar ve boyutları azaltır. SOM'da,hesaplama birimleri (nöronlar) iki katman halinde sıralanır: giriş ve rekabet katmanı. Giriş katmanı N nöronlardan oluşur (her giriş değişkeni için) ve rekabet katmanı düşük boyutlu nöronlardan oluşur. SOM'da kümeleme,mevcut veri nesnesi için birden fazla ünite birbirine sahip olarak gerçekleştirilir. Sisteme girilen verilerden sonra,YSA girişlere göre eğitilir. Ağırlık vektörüne mevcut nesneye en yakın olan birimler aktif veya kazanan birim haline gelir. Girdi verilerinde

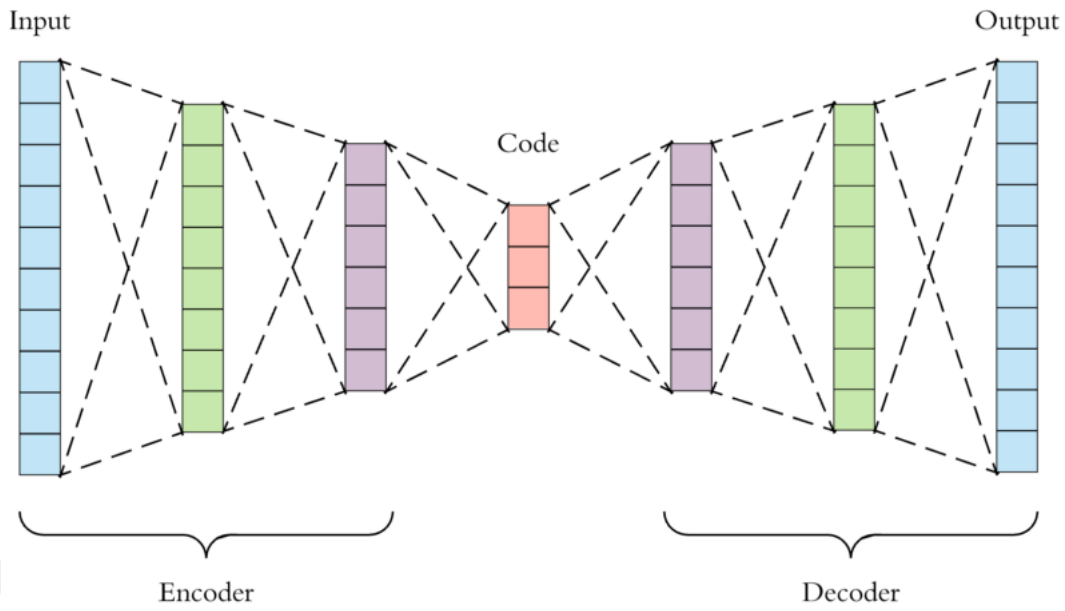


Şekil 4.3: Kendi Kendini Düzenleyen Harita programı(Self-Organizing Map scheme)

varolan komşuluk ilişkileri,eğitim aşamasındaki girdi değişkenlerinin değerlerini kademeli olarak ayarlayarak korunur. Her veri noktası,temsiller için rekabet ederek kendilerini tanır. SOM,ağırlık vektörlerini başlatarak başlar. Ağırlık vektörlerini başlattıktan sonra,bir vektör rastgele örneklenir ve karşılık gelen örneği temsil eden en iyi eşleşmeyi bulmak için ağırlık vektörleri haritasında aranır. Her ağırlık vektörü,yakındaki komşu ağırlıklardan oluşur. Seçilen ağırlık ve komşuları,haritanın rastgele olarak büyümesine ve 2D şekil uzayında farklı şekiller,genellikle,kare / dikdörtgen / altıgen / L şekilleri oluşturmasına izin veren örnek numune vektörünü seçme olasılığını arttırır. Kısacası,öğrenme birkaç adımda ve birçok yinelemede gerçekleşir:

Algorithm 4 SOM Kümeleme Algoritması

1: Her düğümün ağırlıkları başlatıldı;
2: Antrenman verileri setinden rastgele bir vektör seçilir;
3: Her düğüm,hangisinin ağırlığının girdi vektörü gibi olduğunu hesaplamak için incelenir. Kazanan düğüm,En İyi Eşleşme Birimi (BMU) olarak bilinir;
4: Sonra BMI çevre hesaplanır. Komşuların miktarı zamanla azalır;
5: TKazanan ağırlık,daha çok numune vektörü gibi olmakla ödüllendirilir. Komşular da örnek vektörü gibi olurlar. Bir düğüm BMU'ya ne kadar yakınsa,ağırlıkları o kadar fazla değişir ve komşu ne kadar uzak olursa BMU'dan o kadar az şey öğrenir;
6: N yinelemeler için 2. adımı tekrarlayın [5] .



Şekil 4.4: Otomatik Kodlayıcı Yapısı(Autoencoder Architecture)

4.5. *K*-ORTALAMA OLAN DERİN OTOMATİK KODLAYICILAR

Otomatik kodlayıcılar,çıkışın girdiyle aynı olduğu (Feed Forward neural Network) FFNN ailesine aittir. Giriş verileri daha düşük boyutlu bir gizli boşluk temsiline sıkıştırılır ve daha sonra bu gizli boşluk temsilinden çıktıyı yeniden formüle eder. Otomatik dönüştürücü 3 bileşenden oluşur: kodlayıcı,gizli boşluk gösterimi ve kod çözücü. Kodlayıcı,giriş verilerini sıkıştırır ve gizli alan gösterimini yaratır; bu,daha sonra dekoder tarafından girişi **Şekil 4.4:**'te gösterildiği gibi üretmek için kullanılır. Bu yöntemler genellikle sıkıştırma veya boyutluluk azaltma algoritmalarıdır. Otomatik kodlayıcılar kullanarak yüksek girdi boyutluluğunu düşürdükten sonra,oluşturulan kümeleri araştırmak ve analiz etmek için K-ortalama algoritmasına besleriz.

4.6. DERİN GÖMÜLÜ KÜMELEME (DEEP EMBEDDED CLUSTERING (DEC))

Derin gömülü kümeleme (DEC),özellik gösterimlerini ve kümeleme atamasını aynı anda öğrenen [58] bir sinirsel ağı tabanlı yöntemdir. Bir haritalama,girdi veri uzayından daha düşük boyutlu bir özellik uzayına öğrenilir,kümeleme hedefinin yinelemeli olarak optimize edildiği yer.Kümeleme veri noktalarını n $\{x_i \in X\}_{i=1}^{n}$'yi,doğrudan K grubu kümeleri içine,sentroids $\mu_j, j = 1,2,...k$ ile yapmak yerine DEC yöntemi ilk olarak,giriş veri alanının

X,farklı bir alt-boyutlu gizli boşluk Z içine doğrusal olmayan bir eşlemesi gerçekleştirir.

$$f_\theta : X \to Z$$

θ öğrenilebilir parametredir ve Z gizli özellik alanıdır.

Veri noktalarında kümeleme,aynı zamanda Z özellik alanındaki bir takım k küme merkezini ve giriş veri alanını gizli özellik alan θ'ye eşleyen DNN parametrelerini Z öğrenerek yapılır. DEC 2 aşamadan oluşur:

1. Derin otomatik kodlayıcılarla başlatma aşaması.
2. Parametre optimizasyonu (kümeleme),yardımcı bir hedef dağılımının hesaplanması ile Kullback-Leibler (KL) ayrışmasının en aza indirgenmesi arasında yinelenir.

4.6.1. KL ayrışması ile Kümeleme

Önerilen denetimsiz DEC başlangıçta sentroid kümeler ile doğrusal olmayan haritalamanın ilk tahmininde gömülü noktalar arasındaki yumuşak bir atamayı hesaplar. Yumuşak hizalamayı hesapladıktan sonra,f_θ'yi (derin haritalama) günceller ve denklem 4.10 yardımcı hedef dağılımını kullanarak mevcut atamaları öğrenerek,küme merkezlerini yeniden düzenler.

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\| \alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + \left\|z_i - \mu_{j'}\right\| \alpha)^{-\frac{\alpha+1}{2}}} \tag{4.10}$$

4.7. METRİKLER

En ilkel değerlendirme,konuşlandırırken Derin Öğrenme modellerinin kalitesini değerlendirmektir. Her örnek için etiketler bulunduğundan değerlendirme denetimli problemler için daha kolaydır. However there is no such richness in the unsupervised contexts,the notion if testing in this domain is a flawed premise. Ancak,denetlenmeyen modellerin değerlendirilmesi için kaybedilen bir neden değildir. Büyük sayı ölçümleri

denetimsiz kümelenme sonuçlarının kalitesini inceler. Bu metrikler kümelere ilişkin içgörülü olabilir,seçilen algoritma seçimine bağlı olarak değişebilir ve birlikte kategorize etmek için verilerin doğal eğilimi. Metriklerin bazıları şunlardır:

4.7.1. Psueod-F Statistic

Her birinin sırasıyla N veri noktasına sahip K kümeleri olduğunu düşünün. Böylece $\sum n_i = N$(numunenin toplam büyüklüğü). x_{ij}'in küme i'deki j^{th} veri değerini gösterdiğini varsayalım,m_i'in i^th kümesinin küme merkezi olduğunu ve M'nin tüm veri noktalarının genel ortalaması olduğunu varsayalım. Kümeler arasındaki(*sum of squares between* :SSB) kareler toplamı ve (*sum of squares within*:SSE) kümelerdeki kareler toplamı şöyledir:

$$SSB = \sum_{i=1}^{K} n_i.Distance^2(m_i, M) \tag{4.11}$$

$$SSE = \sum_{i=1}^{K} \sum_{j=1}^{n_j} Distance^2(x_{ij}, m_i) \tag{4.12}$$

Ve

$$Distance(a,b) = \sqrt{\sum (a_i - b_i)^2} \tag{4.13}$$

Pseudo-F istatistiği şöyle tanımlanır:

$$F = \frac{\frac{SSB}{K-1}}{\frac{SSE}{N-K}} = \frac{MSB}{MSE} \tag{4.14}$$

Pseudo-F istatistiği,kümeler arasındaki ayrımı hesaplayan kümeler arasındaki (i) MSB ortalama karesinin oranını,(ii) kümelerin içindeki verilerin dağılımını ölçen MSE ortalama kare hatasının oranını ölçer.

4.7.2. Davies-Bouldin (DB) Indeks

DB indeksini hesaplamak için formül:

$$DBI = \frac{1}{n}\sum_{i=1}^{n} max_{j\neq i}(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}) \tag{4.15}$$

burada n kümelerin sayısıdır ve $sigma_i$ küme i'deki tüm veri noktalarının küme sentroid c_i'ye olan ortalama mesafesidir.

Küçük indeks,kümelerin küçük ve merkezlerin birbirinden çok uzak olduğunu belirten iyi kümelere karşılık gelir.Bu nedenle,DB endeksini en aza indiren küme düzeni optimum küme sayısı olarak alınmıştır [51].

4.7.3. Siluet Katsayısı

Bu teknik,her veri noktasının ne kadar iyi kümelendiğinin kompakt ve özlü bir geometrik gösterimini sağlar. Kümeler içinde tutarlılığın doğrulanması ve açıklanması için bir yöntemdir. Bu değer,bir veri noktasının,diğer kümelere (ayırma) acıma göstermede kendi kümesine (uyum) ne kadar benzer olduğunu ölçer [56]. Değer {-1,1} arasındadır. Yüksek değer,veri noktasının kendi kümesine çok iyi uyuştuğunu ve komşu kümelere çok kötü uyuştuğunu gösterir. Veri noktalarının çoğunun yüksek silueti değeri varsa,kümelenme yapısının uygun olduğunu söyleyebiliriz. Çoğunluk düşük veya negatif değerler içeriyorsa,kümeleme yapısı çok az sayıda küme veya çok fazla küme içerir. Siluet metriği,Manhattan veya Öklid mesafesi gibi herhangi bir mesafe metriği ile hesaplanabilir. Siluet değeri şu şekilde belirlendi:

$$S(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{4.16}$$

burada $a(i)$,kendi kümesindeki diğer tüm noktalardan i noktasının ortalama mesafesi olarak belirlenir ve $b(i)$,i'nin diğer kümelerdeki diğer tüm noktalara en küçük ortalama mesafesidir.

4.7.4. Dunn Indeks

Şunu böyle tanımlanmıştır:

$$D = \frac{min_{1 \leq i < j \leq n} d(i,j)}{max_{1 \leq k < n} d^{'}(k)} \tag{4.17}$$

Burada i, j,& k küme endeksleridir,d kümeler arası mesafedir ve $d^{'}$ küme içi mesafedir. Dunn endeksinin değeri ne kadar yüksek olursa kümeleme o kadar uygun olur [54].

4.7.5. Calinski-Harabasz Indeksi (CH Score)

Varyans Oranı Kriteri (VRC) olarak da bilinen Calinski-Harabasz (CH) endeksi,temel gerçek etiketleri önceden bilinmediği takdirde modeli değerlendirmek için kullanılabilir. Yüksek bir CH puanı,modelin daha iyi kümeler oluşturduğu anlamına gelir.

5. METODOLOJİ,ARAÇLAR VE DENEYSEL KURULUM

Bu bölümde,uygulamayı gerçekleştirmek için kullanılan veri kümesi,tasarım yaklaşım ve yazılım araçları açiklanacaktır.

5.1. VERİ KÜMESİ

Denemeler,İslami metinlerin İngilizce versiyonunda yapılmıştır: ***Sahih Bukhari*** [6] . Sahih-i Buhari,7275 hadisi içeren 93 bölümden oluşmaktadırus. Verileri modele aktarmadan önce,veri kümesinde bazı temel ön işlemler gerçekleştirilmiştir. Sık kullanılan kelimeler (durma kelimeler (stop words)) kaldırılmiştir.

5.1.1. Araştırma ve Tasarım Stratejisi

Bu tez boyunca kullanılması planlanan ön işleme ve kümeleme teknikleri mevcut olanlardır. Ancak,Word2Vec,Doc2Vec,GloVe ve fastText gibi metin gömme tabanlı ve kümeleme algoritmalarıyla kullanılan analizleri önermekteyiz.

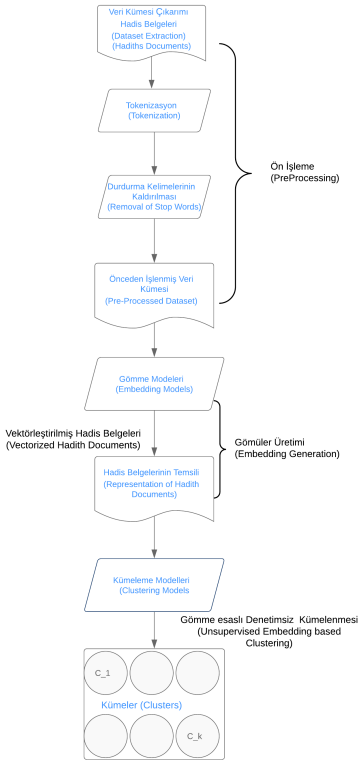
Bu tezde,K-ortalama (K-means),Birch,DEC ve Spektral (clustering) gibi çeşitli kümeleme algoritmaları,belgelerin çeşitli derin temsilleri ile birlikte incelenerek analiz edilmiştir. Bu vaka çalışmasında kullandığımız temsiller Word2Vec,Doc2Vec,GloVe ve fastText'tir. Bu vaka çalışmasının genel araştırma metodolojisi,**Şekil 5.1:**'de gösterilmektedir:

5.1.2. Veri Kümesi Ön İşlemesi

Hadis belgelerini doğrudan denetimsiz kümeleme algoritmalarına beslemeden önce,önce verileri işlemden geçirmemiz gerekir. Ön işleme şunları içerir:

Tokenizasyon Tokenizasyon uzun bir diziyi jeton (token) yani dizgeler olarak adlandırılmış olan sözcükler,belirteçler gibi kurucu parçalarına bölme işlemidir. Bir girdi dizisi (input sequence) olarak *In the name of Allah,the most beneficent and most merciful* ve onun

[6] https://www.quranexplorer.com/hadithebook/english/index.html



Şekil 5.1: Metodoloji (Methodology)

jetonlerini {'In','the','name','of','Allah','the','most','beneficent','and','most','merciful' } olarak dikkate alın. Korpus'un İngilizce versiyonunu kullandığımızdan,korpusumuzdaki tokenizasyon işlemini yürütmek için RegexpTokenizer [7] kullanmiştir.

Durdurma Kelimelerinin Kaldırılması: (Removal of Stop Words *The*,*of*,*an*, *vb.*. gibi dilde sıkça kullanılan durdurma kelimeler faydasız sözcüklerdir. Durdurma kelimelerinin kaldırılmasından sonra,önceki bölümdeki girdi dizisi {'name','Allah','most','beneficent','most','merciful' },gibi görünecektir. Durdurulan Etkisiz kelimelerin kaldırılması,derin sinir ag modellerine geçmeden önce veri kümesini daha iyi hazırlayarak çeşitli NLP metin problemleriyle başa çıkmada için ana adımlardan biridir. Durdurma kelimelerinin kaldırılması için,NLTK[8] araç setinden *stopwords* adlı kütüphaneyi kullanıyoruz.

Hadis Belgelerinin Temsili Kelime gömme modelleri,kelimeleri,cümleleri veya belgeleri yüksek boyutlu uzayda vektörler olarak göstermek için çok etkili yöntemlerdir. Bu

[7] https://www.nltk.org/_modules/nltk/tokenize/regexp.html
[8] https://www.nltk.org/book/ch02.html

tez çalışmasında,Word2Vec,Doc2Vec,GloVe ve fastText gibi çeşitli gömme modelleri kullanarak tüm önceden işlenmiş korpustan gömmeleri oluşturmak ve çeşitli kümeleme teknikleri ile birlikte kullanıldığında sonuçlarını analiz etmek için kullanıyoruz. Bizim çalışmamızda,önceden eğitilmiş gömmeleri kullanmak yerine,gömme modellerini korpusumuzda 300 gömme boyutuna sahip yeni gömmeler oluşturmak için yeniden eğitiyoruz. Gömme modelleri bölüm 3'te ayrıntılı olarak açıklanmıştır. **Kümeleme Algoritmaları** Gömmeleri önceden işlenmiş korpustan ürettikten sonra,bu etiketlenmemiş göbekleri çeşitli denetimsiz kümeleme algoritmalarına besleriz. Bu tez çalışmasında,kelime gömme işlemlerinin K-Ortalama,BIRCH,DEC,Spektral ve SOM denetimsiz kümeleme teknikleri üzerindeki etkilerini 4. bölümde detaylı olarak analiz edip karşılaştırıyoruz.

Yazılım Donanım Araçları Python programlama dilinin bilgisayar araştırmalarında ve derin öğrenme çevresinde yaygın kullanımı nedeniyle,bu tez uygulaması için Numpy [32] ,Scikitlearn [35] Gibi kütüphanelerle Python'u (sürüm 3.7.2) ve Keras [6] Tensorflow'un derin öğrenme kütüphanesi seçtik. Deneyler 2,2 Ghz 4-core Intel Core i7 işlemci üzerinde gerçekleştirildi.

6. SONUÇLAR

Bu çalışmada,kelime kalıplama modellerinin K-means,BIRCH,Spectral ve DEC kümeleme algoritmaları olan 4 farklı kümeleme algoritması üzerindeki etkilerini değerlendirerek karşılaştırıyoruz. Yüksek boyutlu kümeleri görselleştirmek için,t-Dağıtılmış Stokastik Komşu Gömme (t-SNE) tekniğini kullanarak,parçaların boyutlarının sayısını 2(2D - 2 Boyutlu)'ye eşitliyoruz. Bu deneyde,Tablo 6.1'de gösterildiği gibi 300 kalıplama boyutunu ve 0.025 öğrenme oranı ile 1000 çağ kullanıyoruz. Kümeleme algoritmaları için parametreler **Tablo 6.1:**'de ayrıntılı olarak verilmiştir.

6.1. GÖMME MODELLİ *K*-MEANS (*K*-MEANS WİTH EMBEDDING MODELS)

Kalıplama Modeli	Öğrenme Oranı	Çağlar	Kalıplama Boyutu
Word2Vec	0.025	1000	300
Doc2Vec	0.025	1000	300
fastText	0.025	1000	300
GloVe	0.025	1000	300

Tablo 6.1: Gömme Modellerinin Parametreleri

Biliyoruz ki,çok yoğun ve iyi ayrılmış kümeler için CH skorunun daha yüksek olması gerekir. Silhoutte katsayısı,yanlış kümeleme için 1'den,yoğun kümeleme için $+1$'e kadardır. Silhoutte skoru sıfıra yakınsa,o zaman bu,örtüşen kümeler olduğu anlamına gelir. Bu nedenle,çok yoğun kümeler ve iyi tanımlanmış küme modelleri için silhoutte katsayısı daha yüksek olmalıdır. Daha düşük DB endeksine sahip bir model,kümelerin daha iyi tanımlanmış olması ve daha iyi performans gösterdiği,ve model yüksek CH puanına sahipse kümelerin yoğun ve iyi bir şekilde ayrıldığı anlamına gelir.

Kalıplama modellerinden oluşturulan gösterimleri temel alan,verileri kümelemek için K-means algoritmasını kullandığımızda,Doc2Vec tabanlı modelin diğer kalıplama tabanlı kümeleme modellerinden daha iyi performans gösterdiğini analiz ediyoruz. **Tablo 6.3:**'den,Doc2Vec'in verilen ölçütlere göre çok iyi bir puana sahip olduğu,yani daha yüksek silhoutte katsayısına,daha düşük DB endeksine ve daha yüksek CH

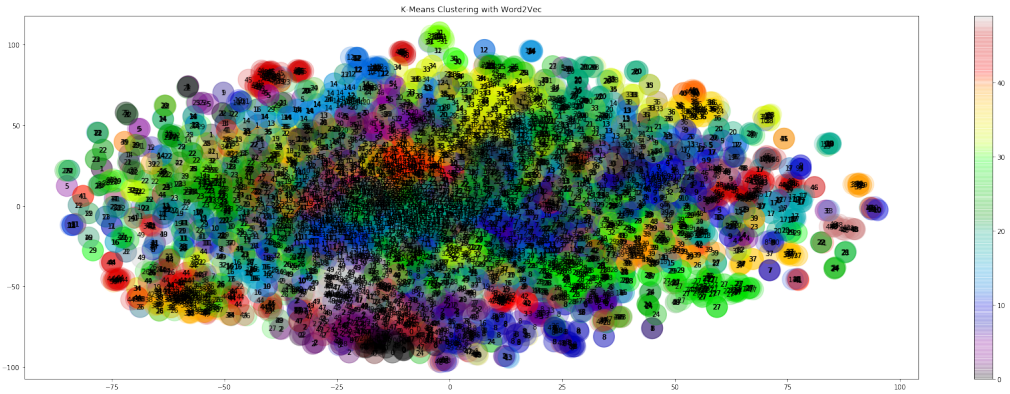
Kümeleme Modeli	Kümelerin Sayısı	Finetune Yineleme	Dallanma Faktörü
K-Means	50	-	-
BIRCH	50	-	50
DEC	50	50	-
Spectral	50	-	-

Tablo 6.2: Kümeleme Modellerinin Parametreleri

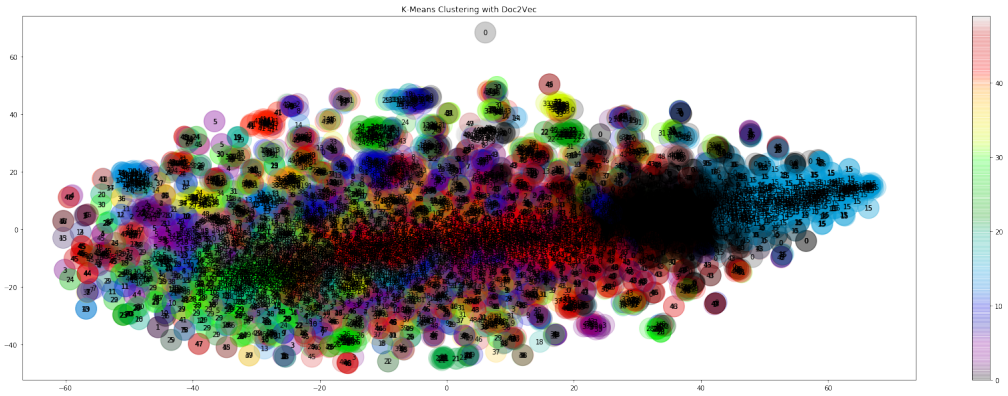
Kalıplama Modeli	Silhouette Katsayısı	DB Indeksi	CH Skoru
Word2Vec	-0.0063	3.4346	44.7168
Doc2Vec	0.0053	2.9048	62.9038
fastText	-0.0385	3.2191	3.2191
GloVe	0.0029	3.0484	50.0802

Tablo 6.3: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile *K*-Means

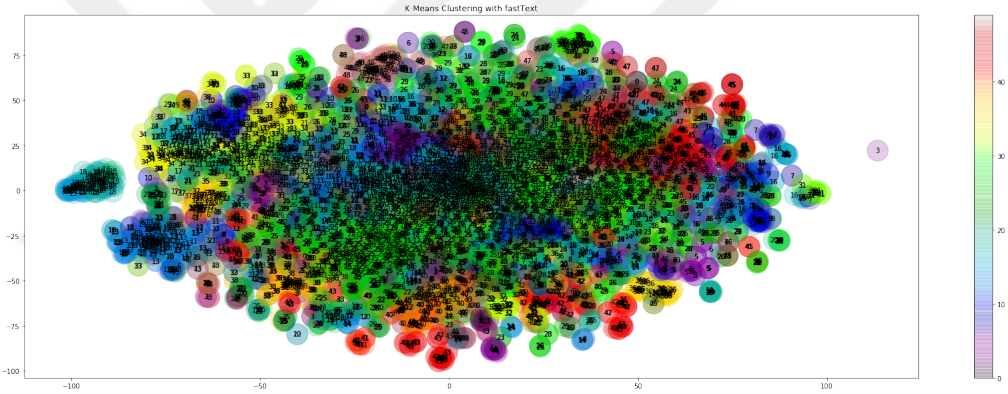
skoruna sahip olduğu,ve dolayısıyla *K*-means kümeleme tekniği ile kullanıldığında Doc2Vec'in diğer kalıplama gösterimlerden daha yoğun ve iyi tanımlanmış kümeler verdiği görülebilir. Bunun nedeni,Doc2Vec tabanlı modelin bağlamsal benzerliğe dayanmasıdır,yani bağlamı hesaba katar ve daha sonra kümeleme gerçekleştirir,diğer modeller ise sadece Hadis bağlamını dikkate almayan kelimelere dayanır. Analiz,Doc2Vec tabanlı kümeleme modelinin yoğun ve iyi tanımlanmış kümeler gerçekleştirdiğini gösteren **Şekil 6.1:**,**Şekil 6.2:**,**Şekil 6.3:**,**Şekil 6.4:** resimlerinden doğrulanabilir.



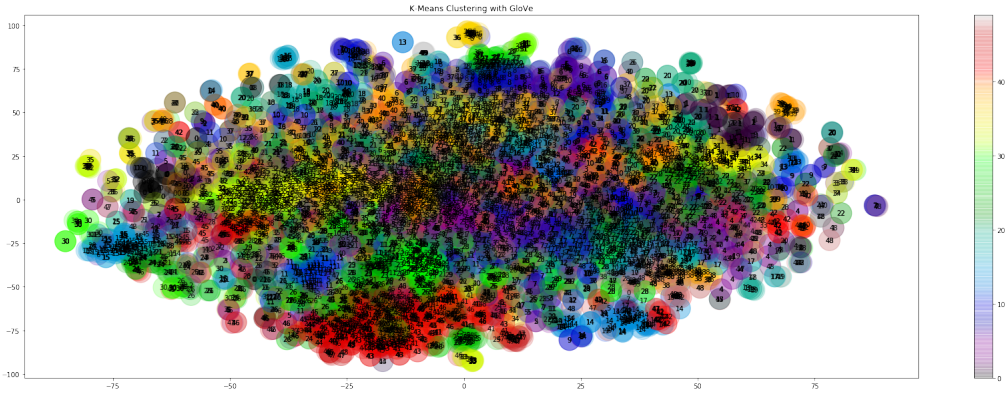
Şekil 6.1: Word2Vec esaslı(based) *K*-Means Kümelenme



Şekil 6.2: Doc2Vec esaslı(based) *K*-Means Kümelenme



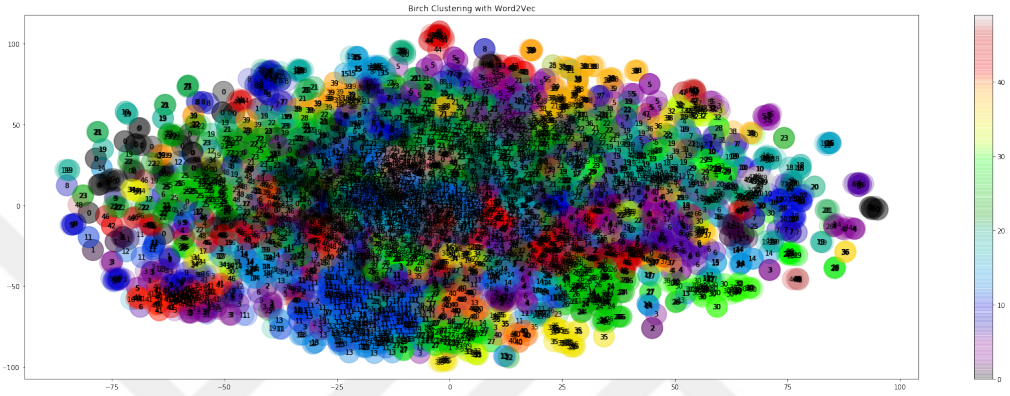
Şekil 6.3: fastText esaslı(based) *K*-Means Kümelenme



Şekil 6.4: GloVe esaslı(based) *K*-Means Kümelenme

Kümeleme Modeli	Silhoutte Katsayısı	DB Indeksi	CH Skoru
Word2Vec	-0.0090	3.7865	37.3402
Doc2Vec	0.0772	3.0303	50.0197
fastText	-0.0009	3.5384	3.5384
GloVe	0.0040	3.3769	43.6556

Tablo 6.4: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile BIRCH



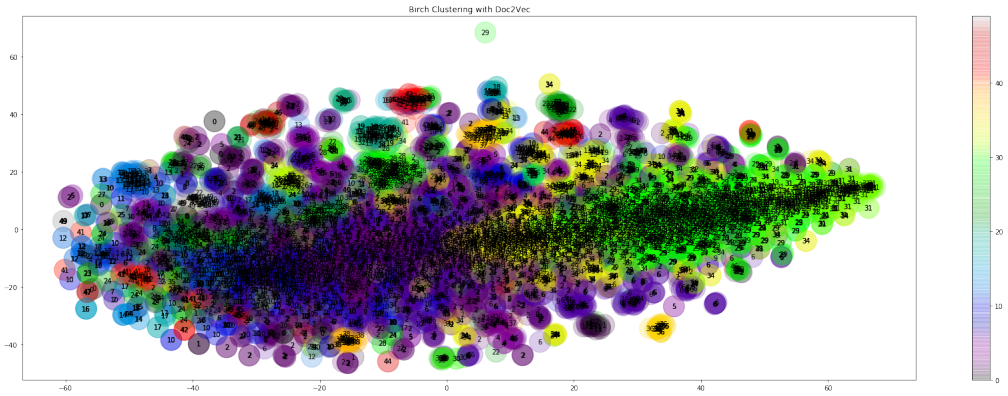
Şekil 6.5: Word2Vec esaslı(based) BIRCH Kümelenme

6.1.1. Gömme Modelli BIRCH (BIRCH with Embedding Models)

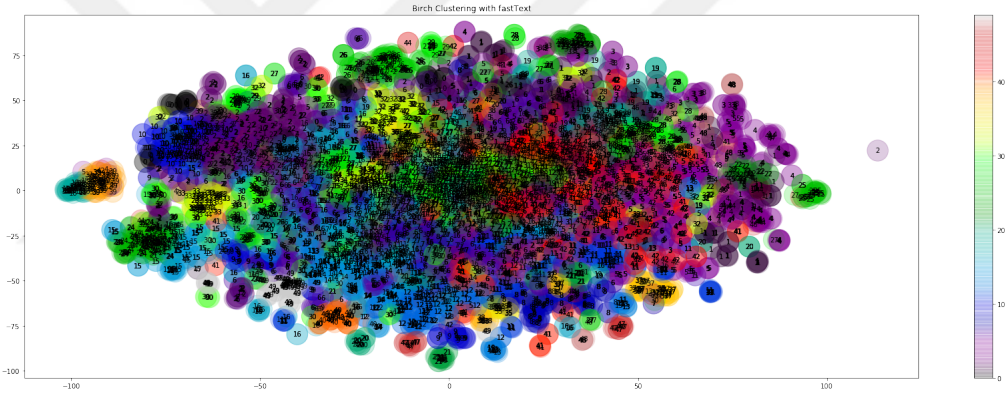
BIRCH kümeleme algoritması için,Doc2Vec gömme modeli yine yüksek silhoutte katsayısı,düşük DB endeksi ve daha yüksek CH puanı ölçütleri ile iyi bir puan gösteriyor. Word2vec,fastText gibi diğer gömme tabanlı BIRCH modeli, **Tablo 6.4:**'te gösterildiği gibi Doc2Vec tabanlı BIRCH ile karşılaştırıldığında zayıf sonuçlar göstermektedir. Bununla birlikte,GloVe tabanlı BIRCH modeli,Doc2Vec tabanlı BIRCH kümeleme modeline çok yakın bir karşılaştırma göstermektedir. Bu,Doc2Vec ve GloVe tabanlı BIRCH'in diğer ikisinden daha iyi ve yoğun kümeler gerçekleştirdiğini gösteren **Şekil 6.5:**,**Şekil 6.6:**,**Şekil 6.7:**,**Şekil 6.8:** grafiklerinden görsel olarak görülebilir. Sebep,Do2Vec'in Hadis belgelerinin bağlamsal benzerliğini dikkate alması ile ilgili olarak daha önce tartışıldığı gibidir.

6.1.2. Gömme Modelli DEC (DEC with Embedding Models)

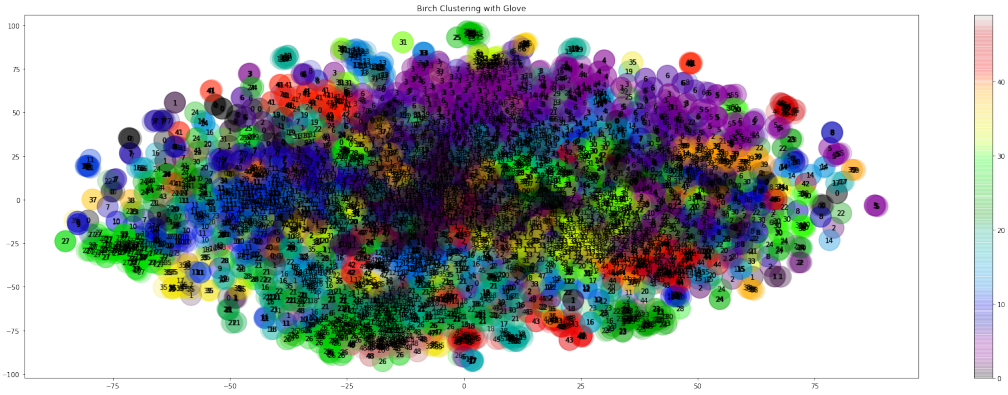
DEC versiyonunda,Word2Vec tabanlı DEC kümeleme modeli,**Tablo 6.5:**'te gösterildiği gibi geri kalanlarından daha iyi sonuçlar elde etmektedir. Ve fastText kalıplama modeli



Şekil 6.6: Doc2Vec besaslı(based) BIRCH Kümelenme



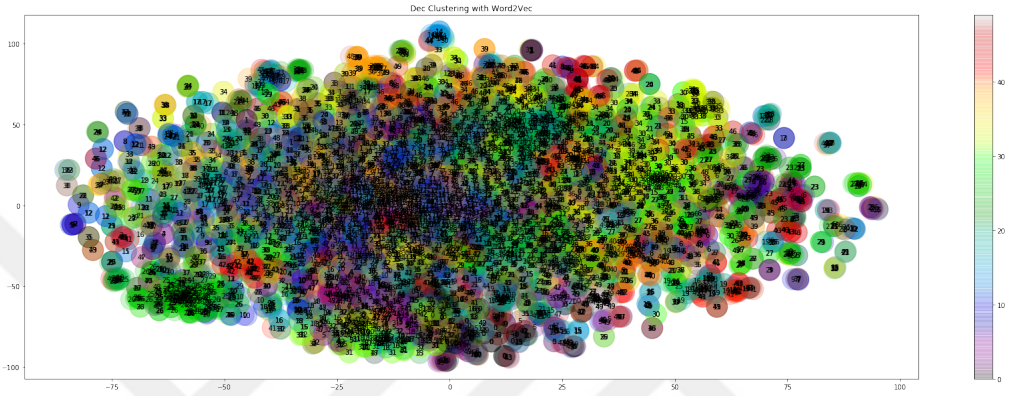
Şekil 6.7: fastText esaslı(based) BIRCH Kümelenme



Şekil 6.8: GloVe esaslı(based) BIRCH Kümelenme

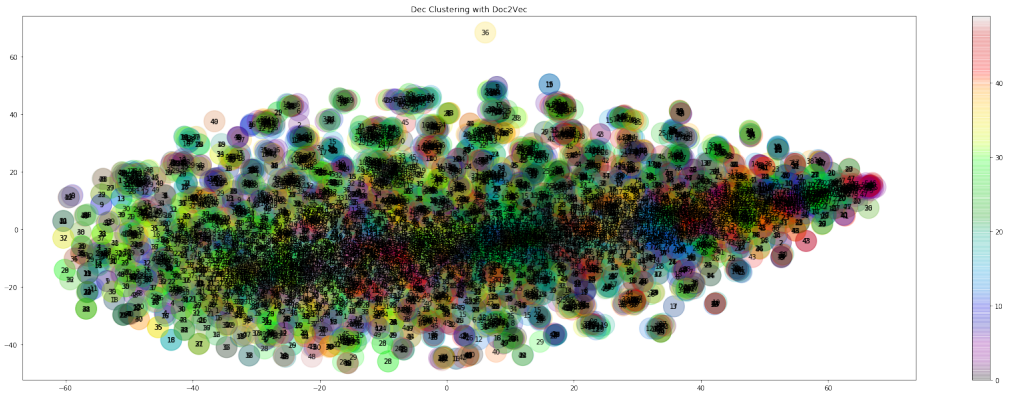
Kümeleme Modeli	Silhoutte Katsayısı	DB Indeksi	CH Skoru
Word2Vec	0.0277	3.9101	40.6479
Doc2Vec	0.1127	8.8679	30.7128
fastText	-0.0235	4.1706	4.1706
GloVe	-0.0194	13.1864	20.7652

Tablo 6.5: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile DEC

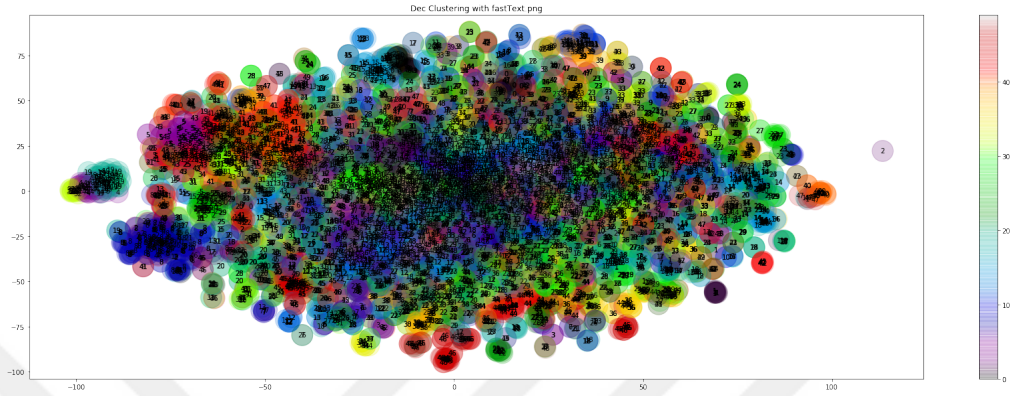


Şekil 6.9: Word2Vec esaslı(based) DEC Kümelenme

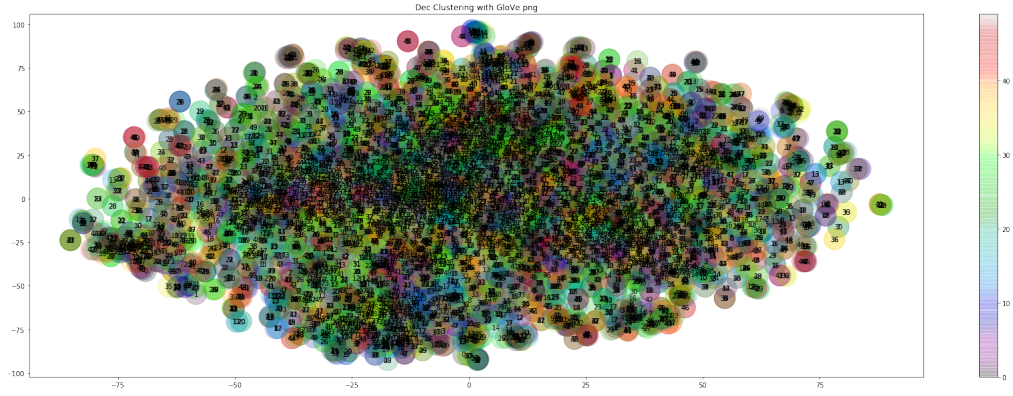
DEC kümeleme algoritması ile kullanıldığında düşük performans gösterir. Doc2Vec tabanlı DEC de,word2Vec tabanlı DEC modeline kıyasla ümit vaat eden sonuçları göstermektedir. Bu,Word2Vec tabanlı DEC'in Doc2Vec tabanlı DEC'den biraz daha iyi performans gösterdiğini gösteren **Şekil 6.9:**,**Şekil 6.10:**,**Şekil 6.11:**,**Şekil 6.12:** grafiklerinden doğrulanabilir. **Şekil 6.11:**'deki yeşil kümelerin uygun bir şekilde gruplanmamış ve grafiğin her yerine dağılmış olduğuna dikkat edin.



Şekil 6.10: Doc2Vec esaslı(based) DEC Kümelenme



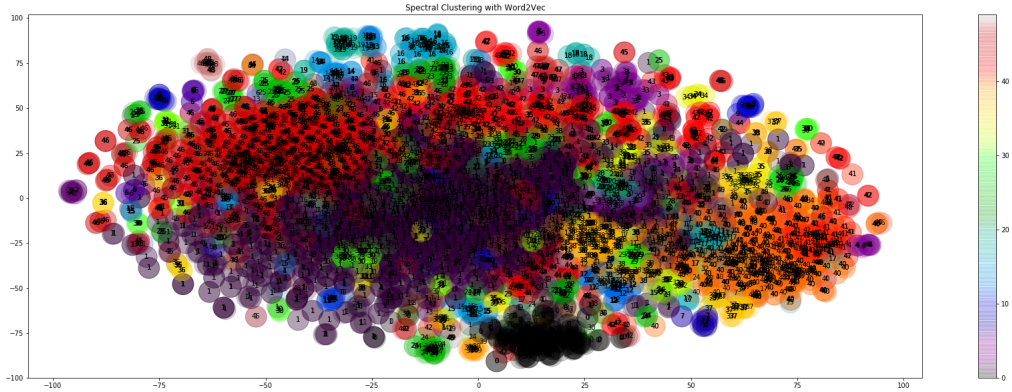
Şekil 6.11: fastText esaslı(based) DEC Kümelenme



Şekil 6.12: GloVe besaslı(based) DEC Kümelenme

Kümeleme Modeli	SSilhoutte Katsayısı	DB Indeksi	CH Skoru
Word2Vec	-0.0061	3.1233	31.6110
Doc2Vec	0.0059	2.8364	45.1595
fastText	-0.1091	2.9846	2.9845
GloVe	-0.0054	3.2436	35.1671

Tablo 6.6: Kalıplama Modelleri (Word2Vec,Doc2Vec,fastText, GloVe) ile Spectral



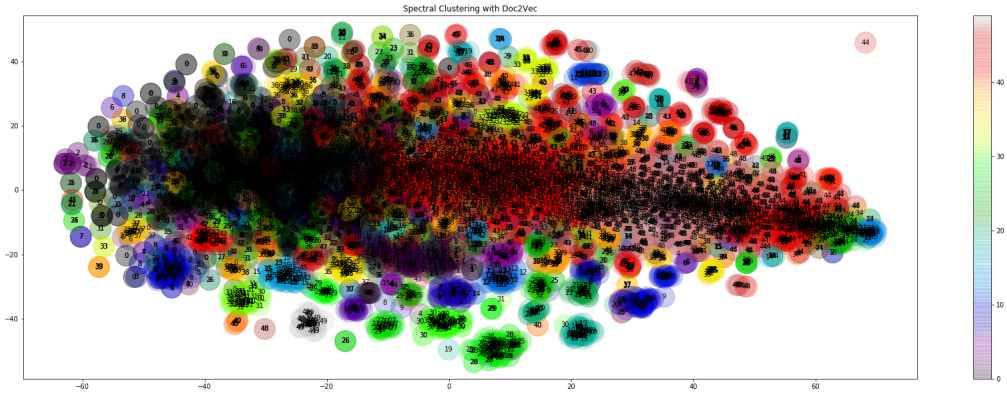
Şekil 6.13: Word2Vec esaslı(based) Spektral Kümelenme

6.1.3. Gömme Modelli Spektral (Spectral with Embedding Models)

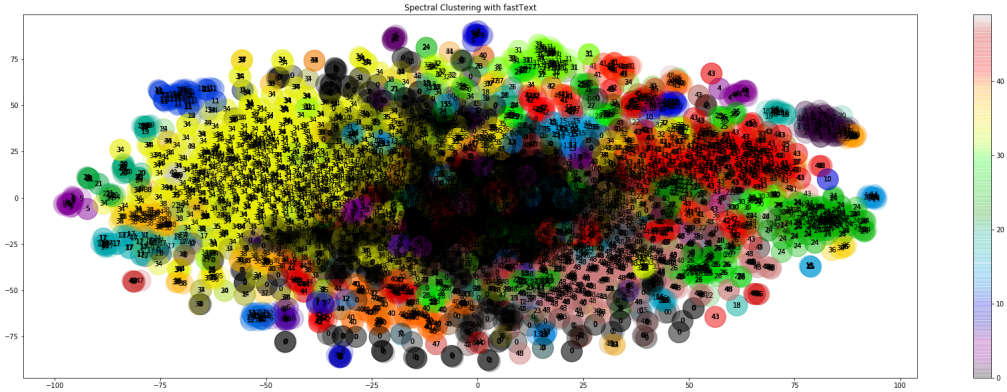
Doc2Vec Tabanlı Spektral kümeleme,diğer kalıplama esaslı Spektral modellerden çok umut verici sonuçlar göstermektedir. Doc2Vec Tabanlı Spektral kümeleme,**Tablo 6.6:**'da gösterildiği gibi yüksek silhoutte katsayısına,düşük DB endeksine ve daha yüksek CH puanına sahiptir; bu,Doc2Vec tabanlı spektral kümeleme modelinin iyi tanımlanmış ve iyice ayrılmış kümeler yarattığı anlamına gelir. Bu,Doc2Vec bazlı spektralın oldukça yoğun ve iyi tanımlanmış kümeler gerçekleştirdiğini gösteren 2B grafikten (**Şekil 6.13:**,**Şekil 6.14:**,**Şekil 6.15:** and **Şekil 6.16:**) kontrol edilebilir.

6.2. TARTIŞMA

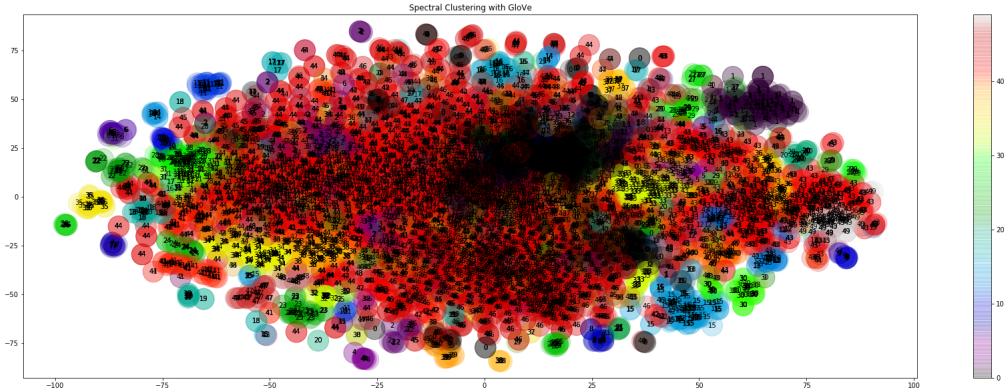
Bu çalışmada,kelime kalıplama işlemlerinin mevcut kümeleme teknikleri üzerindeki etkisini inceledik ve yeni 16 kalıplama tabanlı kümeleme modelini önerdik ve karşılaştırdık. Aşağıdaki 16 farklı gömme tabanlı modeli uyguladık:



Şekil 6.14: Doc2Vec esaslı(based) Spektral Kümelenme



Şekil 6.15: fastText esaslı(based) Spektral Kümelenme



Şekil 6.16: GloVe esaslı(based) Spektral Kümelenme

1. Word2Vec esaslı(based) K-Means.
2. Doc2Vec esaslı(based) K-Means.
3. fastText esaslı(based) K-Means.
4. GloVe esaslı(based) K-Means.
5. Word2Vec esaslı(based) BIRCH.
6. Doc2Vec esaslı(based) BIRCH.
7. fastText esaslı(based) BIRCH.
8. GloVe esaslı(based) BIRCH.
9. Word2Vec esaslı(based) DEC.
10. Doc2Vec esaslı(based) DEC.
11. fastText esaslı(based) DEC.
12. GloVe esaslı(based) DEC.
13. Word2Vec esaslı(based) Spectral Kümeleme.
14. Doc2Vec esaslı(based) Spectral Kümeleme.
15. fastText esaslı(based) Spectral Kümeleme.
16. GloVe esaslı(based) Spectral Kümeleme.

Önerilen tüm modeller arasında,Doc2Vec tabanlı kümeleme tekniklerinin,sonuçlarımıza dayanarak,verilerin çok iyi ve yoğun bir şekilde kümelenmesini sağladığı sonucuna varıyoruz. Doc2Vec tabanlı kümeleme modeli,diğer gömme tabanlı kümeleme modellerinden daha iyi performans gösteriyor. Bu,Doc2Vec tabanlı kümelemenin bağlamı (komşu bağlamı) dikkate alması ve Hadis belgesinin bağlamsal benzerliğine dayanması nedeniyle,önerilen diğerlerinden daha iyi kümelenme gerçekleştirmesi ve dolayısıyla ilgili Hadis belgelerini aynı kümeye gruplandırması nedeniyledir. Biz aynı zamanda,sadece demonstrasyon amaçlı,bir Hadisin programa girdi olarak verildiği zaman,bağlamsal benzerliğine dayalı olarak aynı kümeye giren ilgili Hadisleri ortaya çıkaran temel grafiksel kullanıcı arayüzünü (GUI) de uyguladık. Bu,İslam hukukunda bazı dini veya sosyal meselelere ilişkin kararları verirken karar destek sistemi olarak uygulanabilir.

Sonuçlarımıza dayanarak,fastText tabanlı kümelemenin önerilen modeller arasında en kötü sonuçları gösterdiği sonucuna varıyoruz.

Son olarak,veri kümelemesi için çok sayıda teknik önerilmiş olmasına rağmen hala açık bir soru olduğunu bildirmek istiyoruz. Veri kümelenmesi internetin sürekli büyümesi nedeniyle hala önemli ve en gerekli kısımdır.

7. SONUÇ VE GELECEK ÇALIŞMA

Çalışmamıza, İslam ictihadında karar destek sistemi olarak bir uygulama ile başladı. Toplumsal ve dini konularda fetva verilirken,İslam fakihleri, konu ile alakalı bütün temel,sahih metinleri göz önünde bulundurmak zorundadır. Şurası kesindir ki,bir hüküm eğer bağlamsal materyaller ile desteklenirse daha güçlü olur. Bu da ancak bağlamı algılayıp,gerekli materyalleri bulabilen bir sistem ile mümkündür. Tarihsel olarak,zayıf hükümlerin toplumsal problemlere ve ayrışmalara yol açtığı bilinmektedir. Bunun için geliştirdiğimiz çizimsel kullanıcı arayüzü, girdi olarak bir Hadis alıp,ona bağlamsal olarak benzer Hadisleri bulmaktadır. Bu karar destek sistemi olarak İslami kararlar alınırken,toplumsal yargılamalarda,şirketlerin disiplin komitelerinde,tavsiye sistemlerinde vb. oluşumlarda kullanılabilir.

Bu çalışmada önerilen modeller çok temel aşamadadır ve birçok farklı geliştirme yapılabilir. Yakın gelecekte,önerilen modellerimizi etiketli veri kümesi üzerinde kullanmayı ve performanslarını doğruluk,tamlık ve hatırlama açısından karşılaştırmalı bir şekilde analiz etmeyi planlıyoruz. Ayrıca ELMO,BERT,XLNet vb. gibi daha içeriğe yerleştirilmiş kalıplamaların kümeleme teknikleri üzerindeki etkilerini araştırmak ve etiketli ve etiketsiz veri setleri üzerindeki performanslarını karşılaştırmalı olarak analiz etmek istiyoruz.

Bu araştırma çalışmasında,birçok farklı derin kalıplama modelini ve bunların birçok kümeleme tekniğine etkisini ve Keras,scikit - learn,Gensim,GloVe gibi bazı derin öğrenme kütüphanelerini keşfetme ve öğrenme fırsatımız oldu. Bu araştırma çalışmasının uygulaması github sayfasında bulunmaktadır [9]

[9] https://github.com/shakeel608/TextClustering/blob/master/TextClustering/Embeddings.ipynb

KAYNAKLAR

[1]. Ahmad Al-Nasa'i. al-sunan al-kubra. *Hasan 'Abd al-Man 'am Shilbi*, pages 1–10, 2001.

[2]. Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[3]. Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[4]. Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[5]. Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

[6]. François Chollet et al. Keras. https://keras.io, 2015.

[7]. Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[8]. George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[9]. Scott Deerwester. Improving information retrieval with latent semantic indexing. 1988.

[10]. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:18 Attention is All You Need 10.04805*, 2018.

[11]. Abu Abd Allah Shams al Dhahabi. Tadhkirat al-huffaz, 1955.

[12]. Cicero Dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, 2014.

[13]. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[14]. Delbert Dueck and Brendan J Frey. Non-metric affinity propagation for unsupervised image categorization. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[15]. Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[16]. John Rupert Firth. *Selected papers of JR Firth, 1952-59*. Indiana University Press, 1968.

[17]. Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. Word embedding based generalized language model for information retrieval. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 795–798. ACM, 2015.

[18]. John Gantz and David Reinsel. Extracting value from chaos. *IDC iview*, 1142(2011):1–12, 2011.

[19]. H Paul Grice, Peter Cole, Jerry L Morgan, et al. Logic and conversation. *1975*, pages 41–58, 1975.

[20]. Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[21]. Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[22]. Michael I Jordan. Serial order: A parallel distributed processing approach. In *Advances in psychology*, volume 121, pages 471–495. Elsevier, 1997.

[23]. Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7):881–892, 2002.

[24]. Yungchang Ku, Chaochang Chiu, Yulei Zhang, Hsinchun Chen, and Handsome Su. Text mining self-disclosing health information for public health service. *Journal of the Association for Information Science and Technology*, 65(5):928–947, 2014.

[25]. Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[26]. Hur-Li Lee. Organizing knowledge the chinese way. In *Proceedings of the 73rd ASIS&T Annual Meeting on Navigating Streams in an Information Ecosystem-Volume 47*, page 18. American Society for Information Science, 2010.

[27]. Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

[28]. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29]. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[30]. Marvin Minsky and Seymour Papert. Perceptrons cambridge. *MA: MIT Press. zbMATH*, 1969.

[31]. Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[32]. Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[33]. On The Signs conributors. Hadith and sunnah.

[34]. Esra Kahya Özyirmidokuz. Mining unstructured turkish economy news articles. *Procedia Economics and Finance*, 16:320–328, 2014.

[35]. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[36]. Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[37]. Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[38]. Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986.

[39]. Hinrich Schütze. Word space. In *Advances in neural information processing systems*, pages 895–902, 1993.

[40]. John R Searle. The background of meaning. In *Speech act theory and pragmatics*, pages 221–232. Springer, 1980.

[41]. John R Searle. *Expression and meaning: Studies in the theory of speech acts*. Cambridge University Press, 1985.

[42]. Khalid Mahmood Shaikh. *Hadith & hadith sciences*. Adam Publishers, 2006.

[43]. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[44]. Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[45]. Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[46]. Charles Travis. Saying and understanding: A generative theory of illocutions. 1977.

[47]. A. M. Turing. I.—Computing Machinery And Intelligence. *Mind*, LIX(236):433–460, 10 1950.

[48]. Mayuri Verma. Lexical analysis of religious texts using text mining and machine learning tools. *International Journal of Computer Applications*, 168(8):39–45, 2017.

[49]. Paul Werbos. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974.

[50]. John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*, 2015.

[51]. Wikipedia contributors. Davies–bouldin index — Wikipedia, the free encyclopedia, 2018. [Online; accessed 1-June-2019].

[52]. Wikipedia contributors. Birch — Wikipedia, the free encyclopedia, 2019. [Online; accessed 8-June-2019].

[53]. Wikipedia contributors. Cluster analysis — Wikipedia, the free encyclopedia, 2019. [Online; accessed 28-May-2019].

[54]. Wikipedia contributors. Dunn index — Wikipedia, the free encyclopedia, 2019. [Online; accessed 1-June-2019].

[55]. Wikipedia contributors. Self-organizing map — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Self-organizing$_m apoldid$ = 892805093, 2019. [*Online*; *accessed* 1 − *June* − 2019].

[56]. Wikipedia contributors. Silhouette (clustering) — Wikipedia, the free encyclopedia, 2019. [Online; accessed 1-June-2019].

[57]. Wikipedia contributors. Spectral clustering — Wikipedia, the free encyclopedia, 2019. [Online; accessed 10-June-2019].

[58]. Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[59]. Jiaming Xu, Wang Peng, Tian Guanhua, Xu Bo, Zhao Jun, Wang Fangyuan, Hao Hongwei, et al. Short text clustering via convolutional neural networks. 2015.

[60]. Mahmood Yousefi-Azar and Len Hamey. Text summarization using unsupervised deep learning. *Expert Systems with Applications*, 68:93–105, 2017.

[61]. Ruichang Zhang, Zhanzhan Cheng, Jihong Guan, and Shuigeng Zhou. Exploiting topic modeling to boost metagenomic reads binning. In *BMC bioinformatics*, volume 16, page S2. BioMed Central, 2015.

ÖZGEÇMİŞ

Kişisel Bilgiler	
Adı Soyadı	Shakeel Ahmad SHEIKH
Doğum Yeri	Wagoora Kashmir, India
Doğum Tarihi	05.03.1994
Uyruğu	[ ] T.C. [X] Diğer: India
Telefon	0551 657 31 53
E-Posta Adresi	shakeelzmail608@gmail.com
Web Adresi	https://github.com/shakeel608

Eğitim Bilgileri	
Lisans	
Üniversite	University of Kashmir
Fakülte	Fen Fakültesi
Bölümü	Bilgisayar Bilimi Mühendisliği
Mezuniyet Yılı	2015

Yüksek Lisans	
Üniversite	İstanbul Üniversitesi
Enstitü Adı	Fen Bilimleri
Anabilim Dalı	Enformatik Anabilim Dalı
Programı	Enformatik Programı
Mezuniyet Tarihi	2019

Makale ve Bildiriler

Makaleler

makale 1

makale 2

Bildiriler

Bildiri 1

Bildiri 2