T.C.

DOKUZ EYLÜL ÜNİVERSİTESİ

SOSYAL BİLİMLER ENSTİTÜSÜ

YÖNETİM BİLİŞİM SİSTEMLERİ ANABİLİM DALI

YÖNETİM BİLİŞİM SİSTEMLERİ PROGRAMI

YÜKSEK LİSANS TEZİ

ÇEVİRMENLER İÇİN TEKNİK ALANA YÖNELİK TÜRKÇE - JAPONCA KELİME DESTEK MODÜLÜ GELİŞTİRİLMESİ

Ali Aycan KOLUKISA

Danışman

Doç. Dr. Can AYDIN

İZMİR - 2023

TEZ ONAY SAYFASI

YEMİN METNİ

Yüksek Lisans Tezi olarak sunduğum “Çevirmenler İçin Teknik Alana Yönelik Türkçe - Japonca Kelime Destek Modülü Geliştirilmesi” adlı çalışmanın, tarafımdan, akademik kurallara ve etik değerlere uygun olarak yazıldığını ve yararlandığım eserlerin kaynakçada gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve bunu onurumla doğrularım.

Tarih 15/05/2023

Ali Aycan Kolukısa

ÖZET

YÜKSEK LİSANS TEZİ

Çevirmenler İçin Teknik Alana Yönelik

Türkçe - Japonca Kelime Destek Modülü Geliştirilmesi

Ali Aycan KOLUKISA

Dokuz Eylül Üniversitesi

Sosyal Bilimler Enstitüsü

Yönetim Bilişim Sistemleri Anabilim Dalı

Yönetim Bilişim Sistemleri Programı

Çeviri sorunlarının en başında gelen sıkıntılardan biri olan teknik alan yetkinliği problemi dünyanın çoğu yerinde olduğu gibi ülkemizde de sıklıkla karşılaşılan mesleki problemlerden biridir. Bu problem, çevirisi yapılacak yabancı dile yüksek derecede hâkim olma yeteneğinin yanı sıra, çevirisi istenilen alandaki teknik kelimelere sahip olma zorunluluğundan kaynaklanır. Günümüzde pek çok alanda olduğu gibi bilgi teknolojileri kurumların ve bireylerin yaşadığı sıkıntı ve problemlere oldukça etkili ve pratik çözümler üretebilmektedir. Çevirmenlerin meslekleri ile ilgili konularda yaşadıkları sıkıntılar da bu kapsamda değerlendirildiğinde, bilgi teknolojilerinin bir şekilde bu türdeki sorunlara da çözüm olabileceği düşünülmüştür.

Yapılan bu çalışmada çevirmenlerin yaşamış oldukları sıkıntılarından olan teknik alandaki kelime bilgisi yetkinliği sorununa bilgi ve beraberinde bilişim teknolojileri sayesinde çözüm üretilmeye çalışılmıştır.

WordNet ile ön eğitimli kelime vektör modellerine bağlı olarak elde edilen anlamsal benzerliklere göre hibrit şekilde çalışan bir modül olan, çevirmenler için teknik alana yönelik Türkçe-Japonca kelime destek modülü Python kodları ile geliştirilmiştir.

Anahtar Kelimeler: Anlamsal Benzerlik, Dağılımsal Hipotez, WordNet, Word2Vec, Python, Model_v.1

ABSTRACT

MASTER'S THESIS

Developing A Domain Specific

Turkish - Japanese Vocabulary Support Module For Translators

Ali Aycan KOLUKISA

Dokuz Eylül University

Graduate School of Social Sciences

Department of Management Information Systems

Management Information Systems of Program

The problem of technical field competence, which is one of the foremost problems of translation problems, is one of the professional problems that are frequently encountered in our country as well as in many parts of the world. This problem arises from the necessity of possessing a high level of skills in that foreign language, as well as being familiar with the technical terms in that field. Today, as it is in many fields, information technologies can produce very effective and practical solutions to the problems that individuals and organizations are facing. Considering the technical term difficulties experienced by translators in their profession, it is thought that information technologies can also be a solution to such problems.

In this study, it has been tried to produce a solution to the problem of vocabulary competence in the technical field, which is one of the problems experienced by translators, by means of information and information technologies.

Based on the semantic similarities of words obtained through WordNet and pre-trained word vector models, a domain specific Turkish-Japanese vocabulary support module for translators, which works as hybrid of the two, has been developed with Python codes.

Keywords: Semantic Similarity, Distributive Hypothesis, WordNet, Word2Vec, Python, Model_v.1

ÇEVİRMENLER İÇİN TEKNİK ALANA YÖNELİK TÜRKÇE - JAPONCA KELİME DESTEK MODÜLÜ GELİŞTİRİLMESİ

İÇİNDEKİLER

TEZ ONAY SAYFASI	ii
YEMİN METNİ	iii
ÖZET	iv
ABSTRACT	v
İÇİNDEKİLER	vi
KISALTMALAR	viii
TABLOLAR LİSTESİ	ix
ŞEKİLLER LİSTESİ	x
EKLER LİSTESİ	xiii
GİRİŞ	1

BİRİNCİ BÖLÜM
TEKNİK ALANDAKİ ÇEVİRİ SORUNLARI ve ÇEVİRMENLERİN YAŞADIĞI SIKINTILAR

1.1. PROBLEMİN TANIMI	5
1.2. ARAŞTIRMANIN AMACI	6
1.3. ARAŞTIRMANIN KISITLARI	7
1.4. İLGİLİ ÇALIŞMALAR	7

İKİNCİ BÖLÜM
DOĞAL DİL İŞLEME BAĞLAMINDAKİ KURAMSAL ÇERÇEVE

2.1. DOGAL DİL İŞLEMENİN KAPSAM VE KATEGORİLERİ	16

2.2. ANLAMSAL BENZERLİK VE DAĞILIMSAL HİPOTEZ 17
2.3. DAĞILIMSAL ANLAMBİLİM MODELLERİ 23
2.3.1. HAL 24
2.3.2. LSA 24
2.3.3. LDA (Topic Models) 26
2.3.4. BEAGLE 29
2.3.5. WORD2VEC 31
2.3.6. GLOVE 38
2.4. BİLEŞEN ANALİZİ VE ANLAM ÇÖZÜMLEMESİ 40
2.5. KELİME BENZERLİKLERİ İÇİN WORD2VEC KULLANIMI 43

ÜÇÜNCÜ BÖLÜM
KELİME DESTEK MODÜLÜ GELİŞTİRİMİ

3.1. MODEL_V.1: WORDNET + WORD2VEC 46
3.2. MODEL_V.1 İÇİN GEREKLİ KÜTÜPHANE VE KAYNAKLAR 46
3.3. MODEL_V.1 KURULUMU VE ÇALIŞTIRILMASI 47
3.4. MODEL_V.1 DEĞERLENDİRMESİ VE EDİNİLEN BULGULAR 62

DÖRDÜNCÜ BÖLÜM
KELİME DESTEK MODÜLÜ ÇIKTILARININ TEST EDİLMESİ

4.1. ETKİLİLİK DEĞERLENDİRMESİ 67
4.2. EVALUATOR_P MODÜLÜ İLE ETKİLİLİK DEĞERLENDİRMESİ 70
4.3. EVALUATOR_nP MODÜLÜ İLE ETKİLİLİK DEĞERLENDİRMESİ 89

SONUÇ 99
KAYNAKÇA 105
EKLER

KISALTMALAR

BD : Bilgisayarlı Dilbilim
CL : Computational Linguistics
DAM : Dağılımsal Anlambilim Modeli
DDA : Doğal Dil Anlama
DDİ : Doğal Dil İşleme
DDÜ : Doğal Dil Üretme
DSM : Distributional Semantic Model
HAL : Hiperuzay Analogdan Dile
LSA : Gizli Anlamsal Analiz
NLG : Natural Language Generation
NLP : Natural Language Processing
NLU : Natural Language Understanding
YBS : Yönetim Bilişim Sistemleri
YSA : Yapay Sinir Ağları
US : Uzman Sistemler
KDS : Karar Destek Sistemleri

TABLOLAR LİSTESİ

Tablo 1: WordNet Kelime İlişkileri (isimler) s. 11
Tablo 2: WordNet Kelime İlişkileri (fiiller) s. 12
Tablo 3: ConceptNet 5.5 s. 13
Tablo 4: Kelime Sıralamasının PMI Üzerine Etkisi s. 19
Tablo 5: Derlemdeki Kelimeler s. 20
Tablo 6: Dağılımsal Anlambilim Modelleri – DSM s. 23
Tablo 7: LDA Terim ve Parametre Açıklamaları s. 28
Tablo 8: Word2vec Modellerinin Karşılaştırması s. 31
Tablo 9: Pencere Genişliği Değeri s. 37
Tablo 10: Bileşen Analizi ile Anlam Çözümlemesi s. 41
Tablo 11: Word2vec'te Bileşen Analizi ile Anlam Çözümlemesi (1) s. 42
Tablo 12: Word2vec'te Bileşen Analizi ile Anlam Çözümlemesi (2) s. 43
Tablo 13: Test edilen Model_v.1 Kelime Dosyaları s. 79
Tablo 14: KDS Sınıf ve Özellikleri s. 104

ŞEKİLLER LİSTESİ

Şekil 1: Kavramsal Hiyerarşi s. 9
Şekil 2: Noktasal Karşılıklı Bilgi (PMI) Formülü s. 19
Şekil 3: Vektörel Benzerlik s. 22
Şekil 4: Gizli Anlamsal Analiz (LSA) s. 25
Şekil 5: İkili Dirichlet Dağılımı s. 27
Şekil 6: LDA Modeli İşleyişi s. 28
Şekil 7: Kelime Bazlı N-gram Yöntemi s. 30
Şekil 8: Dairesel Evrişim s. 30
Şekil 9: CBOW ve Skip-gram Modelleri s. 32
Şekil 10: CBOW Modeli İşleyişi s. 33
Şekil 11: Skip-gram Modeli İşleyişi s. 34
Şekil 12: Pencere Genişliği s. 36
Şekil 13: Word2vec Dağılımı s. 38
Şekil 14: Model_v.1 - 1. Adım s. 48
Şekil 15: Model_v.1 - Örnek 2 kelimelik girdi durumu s. 49
Şekil 16: Model_v.1 - Örnek 2 kelimelik girdi dönütü s. 50
Şekil 17: Model_v.1 - Ana Fonksiyon s. 51
Şekil 18: Model_v.1 - Çeviri Adımı s. 52
Şekil 19: Model_v.1 - Birinci Word2Vec yüklenmesi s. 54
Şekil 20: Model_v.1 - Birinci Word2Vec dönütü s. 55
Şekil 21: Model_v.1 - Birinci Word2Vec modelja1 kelime katkısı s. 55
Şekil 22: Model_v.1 - İkinci Word2Vec modelja2 kelime katkısı s. 56
Şekil 23: Model_v.1 - Üçüncü Word2Vec modelja3 kelime katkısı s. 58
Şekil 24: Model_v.1'de "dilbilim" anahtar kelimesi için kaynak dağılımı s. 58
Şekil 25: Model_v.1 - Sudachi Japonca Doğal Dil İşleme Kütüphanesi s. 59
Şekil 26: Model_v.1 - Romaji Japonca okunuşlarının eklenmesi s. 60
Şekil 27: Model_v.1'in son aşaması ve kısmi çıktı s. 61
Şekil 28: Model_v.1'in bilgisayar dosya çıktısı s. 61
Şekil 29: Sudachi kütüphanesi bölümleme modları s. 66
Şekil 30: Değerlendirme ile sonuç ilişkisi s. 67

Şekil 31: Terimsel Uygunluk s. 69
Şekil 32: Katkı Oranı s. 69
Şekil 33: "Evaluate_" başlıklı dosya oluşumu s. 70
Şekil 34: Klasör ve Dosya Oluşumu - 1 s. 71
Şekil 35: Klasör ve Dosya Oluşumu - 2 s. 72
Şekil 36: Evaluator_P Test Modülü - 1 s. 72
Şekil 37: Evaluator_P Test Modülü - 2 s. 73
Şekil 38: Evaluator_P Test Modülü - 3 s. 74
Şekil 39: Evaluator_P Test Modülü - 4 s. 75
Şekil 40: Evaluator_P "csv" Dosyası Çıktıları s. 76
Şekil 41: Evaluator_P Çıktı Dosyaları s. 76
Şekil 42: Evaluator_P "txt" Dosyası Çıktıları s. 77
Şekil 43: Testte Kullanılan Japonca Yazın Dosyaları s. 79
Şekil 44: Evaluator_P tarafından en alt seviyede üretilen analiz dosyaları s. 80
Şekil 45: Evaluator_P Analiz Sonuçları Dosyası (CSV) s. 81
Şekil 46: Model_v.1 Kelime Destek Modülü ile Japonca Doküman Kelimeleri s. 81
Şekil 47: Model_v.1 Kelime Destek Modülü Katkı Oranları (Evaluator_P) s. 82
Şekil 48: Model_v.1 Kelime Destek Modülü Uygunluk Değerleri (Evaluator_P) s. 83
Şekil 49: Alana Göre Toplam Japonca Yazın Kelime Sayısı s. 84
Şekil 50: Alandaki Terim Sayısı ve Uygunluk Derecesi (Evaluator_P) s. 85
Şekil 51: Katkı Derecesi (Evaluator_P) s. 86
Şekil 52: Heatmap Sonuçları (Evaluator_P) s. 87
Şekil 53: Evaluator_P'ye Göre Model_v.1'in Uygunluk ve Katkı Oranları s. 88
Şekil 54: Model_v.1 Destek Modülü Bölümleme Öncesi Kelime Sayıları s. 89
Şekil 55: Evaluator_nP tarafından yapılan ön bölümleme sayıları s. 90
Şekil 56: Evaluator_nP Analiz Sonuçları Dosyası (CSV) s. 91
Şekil 57: Model_v.1 Kelime Destek Modülü Katkı Oranları (Evaluator_nP) s. 92
Şekil 58: Model_v.1 Kelime Destek Modülü Uygunluk Değerleri (Evaluator_nP) s. 93
Şekil 59: Alana Göre Toplam Japonca Yazın Kelime Sayısı (Evaluator_nP) s. 94
Şekil 60: Maksimum ve Minimum Ön Bölümleme ile Uygunluk (Evaluator_nP) s. 95
Şekil 61: Heatmap Sonuçları (Evaluator_nP) s. 96
Şekil 62: Evaluator_nP'ye Göre Model_v.1'in Uygunluk ve Katkı Oranları s. 97

Şekil 63: Model_v.1'in Japonca Yazınlara Olan Genel Katkı Oranı s. 100
Şekil 64: Model_v.1 Destek Modülü Genel Test Sonuçları s. 101
Şekil 65: Bilgi Akışı ve Uzman Sistemler s. 102
Şekil 66: Bilgi Hiyerarşisi ve Uzman Sistemler s. 103

EKLER LİSTESİ

EK 1: Model_v.1 UML Diagramı ek s.1
EK 2: Model_v.1 Destek Modülü Kelime Çıktıları – “çeviri1.txt” ek s.2
EK 3: Model_v.1 Destek Modülü Kelime Çıktıları – “dilbilim9.txt” ek s.3
EK 4: Model_v.1 Destek Modülü Kelime Çıktıları – “anestezi1.txt” ek s.4
EK 5: Model_v.1 Destek Modülü Kelime Çıktıları – “güneşenerjisi1.txt” ek s.5
EK 6: Kullanılan Japonca Yazınlar ek s.7

GİRİŞ

Günümüzde pek çok alanda olduğu gibi bilgi teknolojileri gerek kurumların karşılaştığı sorunlarda, gerekse bireylerin gündelik ya da iş yaşantısındaki sıkıntı ve problemlerinde oldukça etkili ve pratik çözümler üretebilmektedir. Çevirmenlerin meslekleri ile ilgili konularda yaşadıkları sıkıntı ve problemler de bu kapsamda değerlendirildiğinde, bilgi teknolojilerinin bir şekilde bu türdeki sorunlara da çözüm olabileceği düşünülmüştür. Yapılan bu çalışmada özellikle çevirmenlerin yaşamış oldukları mesleki sıkıntılar mercek altına alınarak teknik alandaki kelime bilgisi yetkinliği sorunu irdelenmiş ve bilgi ve beraberinde bilişim teknolojilerinin özellikle Türkçe-Japonca alanındaki çevirmenlere ne şekilde katkı sağlayabileceği konusu ele alınmıştır.

Yapılan literatür çalışmalarına bağlı olarak, kelime benzerliklerini elde etme konusunda genellikle Bilgi Tabanlı ve Derlem Tabanlı olmak üzere iki tip yaklaşımın söz konusu olduğundan bahsetmek mümkündür. Çalışmada öncelikle, Bilgi Tabanlı tekniklerin neler olduğu ve ne türde sonuçlar elde edildiği konusu üzerinde durulmuştur. Daha sonra ise dağılım konusu üzerine odaklanılarak, Derlem Tabanlı teknikler ile bunların neler oldukları tartışılmıştır. Bu aşamada özellikle kelime vektörlerinden söz edilerek, Word2vec ya da GloVe gibi ön eğitimli modellerin nasıl çalıştıkları konusu üzerinde durulmuştur. Devamında ise, bu her iki teknik de sentezlenerek hibrit bir şekilde çalışan ve Türkçe - Japonca kelime destek modülü olarak tasarlanan Model_v.1 geliştirilmiştir.

Çalışmanın ilk bölümünde daha çok bilgi tabanlı teknikler konusu kelime benzerlikleri açısından ele alınarak kelime hiyerarşisi üzerinde durulmuştur. İkinci bölümde ise doğal dil işleme ile doğal dil işlemenin kapsam ve kategorilerinin neler olduğu konusu ele alınmış ve sonrasında ise anlamsal benzerlik ve dağılımsal hipotezinin yanı sıra dağılımsal anlambilim modelleri üzerinde durulmuştur. HAL, LSA, Topic models, BEAGLE, Word2vec ve Glove gibi modellerin çalışma şekilleri de yine bu bölümde ele alınmıştır.

Üçüncü bölümde ise, ilk bölümde ele alınan bilgi tabanlı teknikler ile ikinci bölümde ele alınan derlem tabanlı teknikler sentezlenerek hibrit şekilde çalışan bir

modül olan Model_v.1 kelime destek modülünün Python kodları ile geliştirilme aşamasına geçilmiştir.

Dördüncü ve son bölümde ise, çevirmenler için teknik alana yönelik Türkçe - Japonca kelime destek modülü olarak hazırlanan Model_v.1'in çıktılarının test edilmesi aşamasına geçilmiştir. Bu amaçla kuralcı ve kuralcı olmayan 2 farklı tipte test modülü oluşturularak, Model_v.1'in çıktıları KATKI ve Uygunluk değerleri açısından Japonca alan yazınları üzerinden *Etkililik (effectiveness)* testine tabii tutulmuştur.

Sonuç kısmında ise elde edilen veriler analiz edilerek görselleştirilmiş ve mevcut sonuçlar ile modülün çevirmenlere sağladığı katkılar yönetim bilişim sistemleri açısından da değerlendirilerek tartışılmıştır.

BİRİNCİ BÖLÜM
TEKNİK ALANDAKİ ÇEVİRİ SORUNLARI ve ÇEVİRMENLERİN YAŞADIĞI SIKINTILAR

Çeviri sorunlarının belki de en başında gelen sıkıntılardan biri olan teknik alan yetkinliği problemi dünyanın pek çok yerinde olduğu gibi ülkemizde de sıklıkla yaşanılan durumlardan biridir. Bu problem çevirisi yapılacak yabancı dile yüksek derecede hâkim olma yeteneğinin yanı sıra, çevirisi istenilen alanda yeterli derecede teknik bilgi birikimine sahip olma zorunluluğundan kaynaklanır. Bu durumda, örneğin Japonya'daki SunFlare Co., Ltd. (https://www.sunflare.com) firması ya da Amerika-İngiltere merkezli WGE Language Consultancy Ltd. (https://wgelang.co.uk/) çeviri firmasında olduğu gibi bu alanda faaliyet gösteren pek çok yabancı çeviri firmasının web siteleri incelendiğinde, çevirmen olarak işe alacağı tam ya da yarı zamanlı çalışanlarından hem alan yetkinliği hem de yabancı dil yetkinliğinin her ikisinin de arandığı görülmektedir. Dolayısıyla, bir çevirmenin belirli bir alanda mesleki eğitim almasının yanı sıra üst düzey yabancı dil eğitimi görmüş olması ya da o alanda çok uzun süreli çeviri tecrübesi edinmiş olması gerekmektedir. Ancak ülkemizde yabancı dil eğitimi ya da çeviri eğitimi veren kurumların mevcut koşulları incelendiğinde, bu tür bölümlerde özellikle yukarıdaki gibi belirli bir teknik alana yönelik çeviri eğitimi verebilen kurumların oldukça az sayıda olduğu görülür (Köksal, 2008: 69; Eruz, 2008: 170).

Çeviri türleri incelendiğinde, her ne kadar çalışmalar arasında bir takım farklılıklar görülse de genellikle uzmanlık gerektiren metinlerin ayrı bölümlendiği görülür. Örneğin, Doğan (2014: 137-138)'de metin türlerinin: 1) Bilgilendirici, 2) Anlatımcı, 3) İşlemsel, 4) İşitsel ya da çok araçlı şekilde bölümlenir ve bunlardan özellikle "Bilgilendirici metin türü"nün uzmanlık gerektiren nesnel durum, olay ve olguları betimleyen metin türlerinden olduğu bilgisi verilir. Burada "Anlatımcı metin türü" olarak ele alınan ise, daha çok edebi türdeki yazınsal metinler olarak özetlenebilir. Diğer bir yandan Doğan (2014: 138)'teki "İşlemsel metin türü"nün de uzmanlık gerektiren alana dâhil edilebileceği unutulmamalıdır. Neticede ilan panoları ve reklam metinlerini ele alan "İşlemsel metin türü" uzmanlık bilgisi gerektiren bir alan olarak değerlendirilebilir. Burada yine bilgilendirici ya da

bilimsel bir konferans söz konusu olduğu takdirde "İşitsel ya da çok araçlı metin türü"nün de kısmi olarak uzmanlık gerektiren metin türü olarak ele alınabileceğini söylemekte fayda vardır. Yine Göktürk (1994: 26-27) de, Reiss (1971, 1983)'ten faydalanarak, metin türlerini işlevlerine göre sınıflandırıp, 1) Betimleyici işlev, 2) Anlatım işlevi, 3) Seslenme işlevi, 4) İşitsellik işlevi şeklinde ayrılmaktadır. Ancak Göktürk (1994)'deki bölümlemede de öykü, roman, oyun, şiir gibi biçim ağırlıklı metinleri bünyesinde barındıran "Anlatım işlevine sahip metin türü" haricindekilerin yine uzmanlık gerektiren teknik alan bilgisine ihtiyaç duyulabilen metin türlerine tekabül edebilen metinler olduğundan söz etmek gerekir.

Tabernig (1964) ise metinleri, 1) Teknik ve bilimsel metinler, 2) Felsefi metinler ve 3) Edebi metinler olarak 3 temel sınıfa bölümlemektedir. Yine Brang (1963: 421) de, 1) Haberler, incelemeler, ticari ve resmi belgeler ve bilimsel metinler; 2) Organizasyonlara ilişkin ve siyasi belgeler; 3) Edebi metinler şeklinde bir ayrım yapmaktadır.

Aslında burada, Doğan (2014) ile Göktürk (1994)'teki Anlatımcı metin türü haricindeki 3 grubun, uzmanlık alan bilgisi gerektirebildiğini söylemek mümkündür. Yine, Tabernig (1964)'teki 2 ve 3'ün de teknik alan bilgisi ön planda olmadığından Doğan (2014) ile Göktürk (1994)'teki Anlatımcı metin türüne ve Tabernig'in 1'deki Teknik ve bilimsel metinler grubu ise Doğan ile Göktürk'teki diğer 3'lü gruba dahil edilebilir. Yine, Brang (1963)'te de durum çok farklı olmayıp, 1 ve 2 uzmanlık alan bilgisi gerektiren bir grup, 3 ise bu tür bir bilginin ön planda olmadığı Edebi metinler şeklinde ayrılabilir.

Dolayısıyla, yukarıdaki bilgilerde genellemeye gidilerek, aslında çeviri türlerini sınıflandırırken tıpkı Aksoy (1998: 72)'de belirtildiği gibi edebiyat dışındaki tüm ilim ve bilim alanlarını teknik çeviri olarak kabul edebilmemiz mümkündür. Dolayısıyla bu çalışmada bahsedilen teknik alan kavramı da bu şekilde olup, yazılı ya da sözlü edebi metinler ile günlük sıradan konuşmalar dışında kalan herhangi bir ilim ve bilim alanına yönelik tüm alanları kapsamaktadır. Bu bağlamda, her ne kadar oldukça geniş bir kapsam elde edilmiş olsa da, Anamur (1997)'de "teknik çeviri, sağlık bilimlerinden balıkçılığa, hukuktan arıcılığa kadar bir 'özel alan dili' kullanılan bilim, meslek ya da uzmanlık dallarında yazılmış olan çıkış betiklerinden yapılan çeviri türü" olarak tanımlanmakta ve teknik alanın

uzmanlık dili olan terimlerin her birinin "bir tek kavramı, bir tek nesneyi ya da bir tek süreci niteleyen anlambirimler" olarak nitelendirildiği görülmektedir (Anamur, 1997: 39). Dolayısıyla, teknik alan olarak kapsam her ne kadar geniş olsa da, kaynak dildeki bir kelimenin hedef dildeki eşdeğerliği o teknik alanda bire bir eşleşme sağladığından, edebi metinlerde olduğu gibi yan anlamlar konusunda endişelenmeye meydan vermemekte ve böylece belirli ölçüde kapsamsal bir sınırlama getirmektedir.

Çeviri konusunda çevirmenlerin yaşadığı başlıca sıkıntı, yukarıdaki mevcut durum da göz önünde bulundurularak, edebi ve genel günlük konular dışındaki her alanın uzmanlık gerektiren bir alan olma potansiyelinin mevcut olmasından kaynaklanır. Bu çalışmada "teknik alan" olarak nitelendirdiğimiz bu tür bir alanda gelen bir çeviri metni, şüphesiz bir çevirmen için işin üstesinden gelip gelememe kaygısı ile kazanç sağlama arasındaki çelişkiyi ne derece tolere edebileceği ile alakalıdır. Daha önce uzmanlığı olmayan yahut hiç çevirisini yapmadığı bir alanda aniden gelen bir çeviri söz konusu olduğunda, genellikle çevirmenin elinde iki seçenek bulunur. Ya bu işi alacaktır, ya da geri çevirecektir. Bu durumda çevirmen öncelikle çevirinin yazılı olup olmadığı bilgisine başvurur. Eğer çeviri yazılı bir metin ise çoğunlukla internet ve sözlük kullanımına bağlı olarak fazladan bir miktar süre ve efor sarf ederek hem alan hem de kelime bilgisini arttırma yolunu seçer ve çeviriyi kabul eder fakat bazen de uzmanlık alanı dışında olduğu gerekçesiyle reddeder; ancak çeviri sözlü bir çeviri ise çoğu çevirmenin alanı dışındaki bir çeviriyi almak istemediği sıklıkla görülür[1]. İşte bu noktada bilgi teknolojileri, çevirmenlerin uzmanlık alanı dışında gelen yazılı ya da sözlü çevirilerdeki kaygılarını ortadan kaldırabilecek ya da yüklerini hafifletebilecek ne tür bir destek sağlayabilir sorusu çalışmamızın çıkış noktasını oluşturmuştur.

1.1. PROBLEMİN TANIMI

Çevirmen olarak çalışanların, aniden gelen uzmanlık alan dışı çeviriler konusunda hazırlıklı olabilmeleri amacıyla hedef dildeki alana yönelik kelimeleri kaynak dildeki anlamlarıyla birlikte derleyerek liste halinde bir çıktı döndürebilecek

[1] ALBAMU Çeviri Bilişim Danışmanlık ve Yayıncılık Ltd. Şti.'nin aynı zamanda yeminli tercümanı olan şirket müdürü ile yapılan görüşme ve fikir alışverişine dayanmaktadır.

bir modülün bilgi teknolojileri kullanılarak elde edilebilmesi mümkün mü? Bu tür bir modülün çevirmenlere yaptıkları iş hususunda herhangi bir katkısı mevcut mu? Mevcut ise ne tür bir katkısı bulunmaktadır?

Bu noktada yukarıda bahsi geçen problemin çözümüne yönelik olarak H0 ve H1 olarak iki adet hipotez üreterek bunlardan hangisinin gerçekleşeceği test edilebilir diye düşünülmektedir.

H0 hipotezimiz, çevirmen olarak çalışanların, aniden gelen çeviriler konusunda hazırlıklı olabilmeleri amacıyla, bilgi teknoloji ve imkânlarından faydalanarak hedef dildeki alana yönelik kelimeleri kaynak dildeki anlamlarıyla birlikte derleyerek liste halinde çıktıya dönüştüren bir modülün çevirmene sağlayacağı herhangi bir katkısı bulunmadığı yönündedir.

H1 hipotezimiz ise, çevirmen olarak çalışanların, aniden gelen çeviriler konusunda hazırlıklı olabilmeleri amacıyla, bilgi teknoloji ve imkânlarından faydalanarak hedef dildeki alana yönelik kelimeleri kaynak dildeki anlamlarıyla birlikte derleyerek liste halinde çıktıya dönüştüren bir modülün çevirmene katkısı bulunduğu yönündedir.

H1 hipotezimizin sağlandığı durumda, bu katkıların neler olabileceğinin ortaya konulması gerekmektedir. Ancak, H1 hipotezimizin sağlanmadığı durumda ise H0 hipotezimiz geçerli olacaktır.

1.2. ARAŞTIRMANIN AMACI

Bu araştırmanın amacı, çevirmen olarak çalışanların, uzmanlık alanları dışında gelen yazılı ya da sözlü çevirilerdeki kaygılarını ortadan kaldırabilme ya da yüklerini hafifletebilme ve böylece bu tür çeviriler konusunda hazırlıklı olabilmelerini sağlayabilme amacıyla, bilgi teknolojilerinden faydalanarak, kullanıcıdan o teknik alanda Türkçe olarak alınacak anahtar kelime vasıtasıyla yine o alandaki hedef dilde sıklıkla kullanılan diğer teknik kelimeleri derleyip, kaynak dildeki eşdeğerlikleri ile birlikte bir liste halinde çıkarabilecek ve temelleri *Bilgi Tabanlı* teknikler ile *Derlem Tabanlı* tekniklerin her ikisine de dayanarak bunları hibrit bir şekilde sentezleyebilen bir kelime destek modülü geliştirmek ve sonrasında

ise geliştirilen bu modülün çevirmene ne derece faydasının olduğunu yapılacak testler vasıtasıyla ortaya koymaktır.

Bu amaç doğrultusunda öncelikle kullanıcıdan çevirisini yapacağı teknik alanla ilgili Türkçe olarak birkaç anahtar sözcük alınmış, daha sonra bu anahtar sözcüklerden yola çıkılarak o alanda sıklıkla kullanılan diğer kelimelere ulaşılmış ve sonrasında ise bu kelimeler hedef dile aktarılarak kaynak dildeki eşdeğerlikleri de bunlara ilave edilmiştir.

Son aşamada ise, geliştirilen modül vasıtasıyla elde edilen çıktı bir kelime listesi haline getirilip çıktısı alınarak, modelin ne derece faydalı olduğunun testi gerçekleştirilmiş ve böylece gerekli optimizasyonların yapılması sağlanmıştır.

1.3. ARAŞTIRMANIN KISITLARI

Araştırmamızda kullanıcıdan alınan anahtar kelimeye bağlı olarak Doğal Dil İşleme (DDİ) teknikleriyle WordNet'ten döndürülen veriler, Word2vec ön eğitimli Japonca modelleri ile yine Doğal Dil İşleme (DDİ) teknikleri vasıtasıyla ilişkilendirilmiştir. Ancak yukarıda bahsi geçen uygulama ve modeller İngilizce ile Japonca dillerinde hazırlandıklarından kelimeler arasındaki bağlantıları bulabilmemiz için kullanıcı tarafından girilecek olan bağımsız değişkenin öncelikle İngilizce diline aktarılarak diğer işlemlerin gerçekleştirilmesi söz konusu olmuştur. Bu durum her ne kadar doğa bilimleri ya da mühendislik gibi teknik bir alana yönelik kelimeler hususunda herhangi bir sorun teşkil etmese de, İngilizce kullanılmayan ülkelerin kültürüne özgü kelimeler söz konusu olduğunda düzgün şekilde çalışmayacak olup bu da araştırma kısıtlarını belirlemektedir. Bu sebeple, araştırmamızda İngilizce kullanılmayan ülkelerin sadece kendi kültürüne özgü kelimeler (Ör. etnografik kelimeler, vb.) alana yönelik anahtar kelime seçimi kapsamı dışı bırakılmıştır.

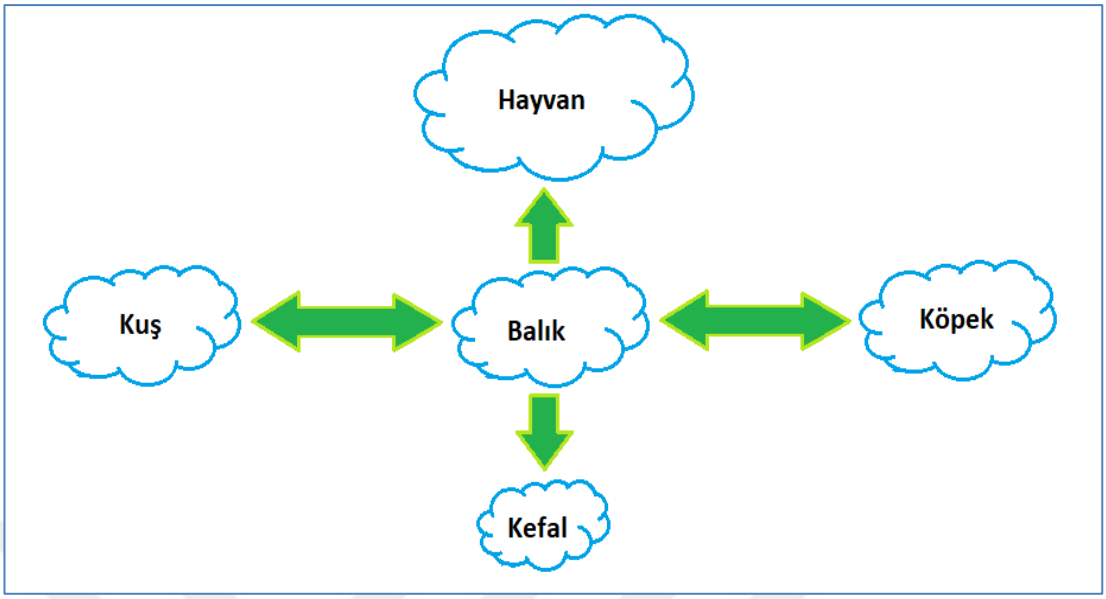
1.4. İLGİLİ ÇALIŞMALAR

Alana özgü kelimelerin (domain-specific words) tespitine yönelik daha önceki çalışmalara baktığımızda çıkış noktası olarak genellikle, kelimelerdeki

anlam bulanıklığını netleştirme (word sense disambiguity) ve varlık ismi tanıma (named entity recognition) gibi katmanlı yöntemlerden faydalanıldığı ve kelimelerin bağlamsal anlamlarının (word sense) alanı belirlemede oldukça önemli rol üstlendiği ve ayrıca alana özgülük kavramının kelimelerin bağlamsal anlamları (word sense) ile direk ilişkisi bulunduğundan, alana özgü bu kelimelerin içinde bulunduğu bağlamdaki anlamsal benzerliklerden faydalanmanın mümkün olduğu görülmüştür (Su Nam ve Lawrence, 2011; Rigutini ve diğerleri, 2005). Ancak tüm bu tür çalışmaların esas temel dayanak noktası öncesinde uzun yıllardır çalışılmakta olan "ontoloji" sayesinde ortaya çıkmıştır.

Ontoloji, kelimesi temel olarak "Varlık Bilimi" anlamına gelir. Bilinen tüm kavramları inceleyip onları sınıflandırmayı hedefler. Itoh (2000)'de ontoloji kavramının farklı bakış açılarından ve birden fazla kez tanımlandığı görülür. Örneğin, yapay zekâ araştırmaları açısından "kavramsallaştırmanın açık bir özelliği" olarak tanımlanır ve bu kavramsallaştırmanın, hedef dünyanın tüm kavramlarını ve bunların birbiri ile olan ilişkilerini ifade ettiğinden ve ayrıca bu kavramsallaştırmaya iş, oluş bildiren eylemlerin de dâhil edilmesi gerektiğinden söz eder (Itoh, 2000: 116-117). Haspolat (2018) ise, bu tür bir kavramlar dünyasını inşa ederken, ontoloji ile tanımlanan bilgiyi işlemek için kullanılan aracın mantık olduğundan bahseder. Dolayısıyla kavramların birbirleri ile olan ilişkilerinin belirlemede de belirli bir mantık çerçevesi oluşturulması gerekir. Özellikle, alana özgü kavramların yahut diğer bir deyişle terimlerin kavram hiyerarşilerinin, ilişkilerinin tanımlanması ve kurallarının öncesinde oluşturulan belirli bir mantık çerçevesinde açıklanması gerekir. Bu ilişkiler dikey, yatay ya da paralel olarak adlandırabileceğimiz şekilde olabilmektedir. Örneğin, balık ve hayvan kavramları arasında dikey bir ilişki mevcuttur. Yahut bir balık cinsi olan kefal ile balık arasında da yine dikey fakat birincisine göre ters yönde bir ilişki mevcuttur denilebilir. Köpek ile balık kavramları arasında ise yatay bir ilişki bulunmaktadır. Yine buna kuş kavramını da eklemek mümkündür.

Şekil 1: Kavramsal Hiyerarşi



Kaynak: Yazar tarafından derlenmiştir.

Benzer bir hiyerarşiden Lakoff (1987: 166-167), Markman (1991: 138), Taylor (1995: 47-48) ve Önal (2019: 664-667)'da da bahsedildiği görülür. Özelikle çocukluk döneminde nesnel dünyaya ait soyut ya da somut varlıkların zihinsel temsillerini oluşturma amacıyla yapmış olduğumuz bu tür bir hiyerarşik ayrımın, belleğimizdeki yükü hafiflettiğinden ve bunun sonucu olarak bilgi depolama ve geri çağırmada bizlere yardımcı olduğundan söz etmek mümkündür (Önal, 2019: 663-665; Markman, 1991: 11).

Bu durum aslında internet ve web'in mucidi Tim Berners Lee tarafından da farkına varılmıştı. 1990'larda ilk olarak geliştirdiği hipermetin (html) sistemi ile internet üzerinden bilgi paylaşımı fikrini ortaya atan ve geliştiren Tim Berners, web henüz emekleme aşamasında olmasına rağmen 2000'li yılların başında "Semantik Web" kavramını ortaya atmıştır. Web'de bulunan bilgilerin etiketlenerek ontolojilerin oluşturulabileceğini ve böylece çok daha dinamik ve verimli bir kullanım ortaya çıkarılabileceğinden bahsetmiş, adeta günümüzün internet ortamını vizyonunda geliştirerek temellerini atmıştır (Frauenfelder, 2001).

Diğer bir yandan Semantik Web (Anlamsal Web), kavramı henüz yeni yeni gelişmekte iken, web üzerindeki ontolojilerin bilgisayarların insanların dilini anlamada ne derece önemli olduğu konusu kavrandığından 2000 sonrasında bu konu

ile ilgili dünyanın pek çok yerinde yoğun araştırmaların yapıldığı görülmektedir. Amerika'da DARPA (Amerikan ileri savunma araştırmaları projeleri ajansı) tarafından desteklenen World Wide Web Konsorsiyumu başta gelse de, Network Inference, Hewlett-Packard ve Nokia gibi teknoloji devleri de Semantik Web kavramının önemini ilk etapta fark ederek bu konuda çalışmalar yapan özel sektör firmaları arasında yer aldığı görülür (Frauenfelder, 2001). Fakat bu konudaki çalışmalar elbette sadece Amerika ile sınırlı kalmamış ve 2005'te Avrupa Birliğinin de hibe desteğiyle NeOn-Projesi adlı bir proje geliştirilmiştir.

14 Avrupalı ortağı içeren ve Avrupa Komisyonu tarafından finanse edilen NeOn Projesi, Mart 2006'da başlamış ve amacı dağıtılmış organizasyonlarda büyük ölçekli semantik uygulamalar için ontolojileri kullanma konusunda en son teknolojiyi ilerletmek ve geliştirmek olmuştur [2]. Bu amaçla pek çok program ve API şeklinde uygulama da geliştirilmiştir. Bunlardan en yaygın ve tanınanları olarak Text2Onto, Oyster, OntoModel, OWLDoc, Evolva, Labeltranslator ve SPARQL gibileri sayılabilir ancak versiyonları ve kullanım amaçları farklı olsa da toplamda çok daha fazla sayıda API geliştirildiği görülür [3].

Bir diğer yandan tüm bu ontoloji araçları ve semantik web kavramının adeta bir dayanak noktası konumunda ise özellikle **WordNet**'ten bahsetmek gerekir. Kelime ilişkilendirme çalışmaları ile başlayarak bu amaçlar doğrultusunda karmaşık deneylerle devam eden psiko-dilbilimciler, 20.yy'lın ikinci yarısına gelindiğinde insan zihninde kavramları öğrenirken inşa edildiği varsayılan zihinsel sözlüğün, sözlükbilimde de yararlanılabilecek birçok özelliğini keşfetmiştir. Princeton Üniversitesi'ndeki bir grup psikolog ve dilbilimci ise 1985'te bu tür araştırmalardan yola çıkarak sözlüksel bir veritabanı geliştirmeye koyulmuş ve bunun sonucu olarak ise **WordNet** ortaya çıkmıştır (Miller, 1985). Başlangıçta sözlükteki girdileri yalnızca alfabetik olarak değil, kavramsal olarak da aramayı imkanlı kılacak bir ortam sağlama amacıyla yola çıkılıp geleneksel türden bir çevrimiçi sözlükle yakın bağlantılı olarak kullanılması hedeflenmiş olsa da, çalışmalar ilerledikçe, WordNet kendi ilke ve hedeflerini daha iddialı bir şekilde formüle edebilmiştir (Miller ve diğerleri, 1990).

[2] http://neon-project.org/nw/Welcome_to_the_NeOn_Project.html
[3] http://neon-toolkit.org/wiki/Neon_Plugins.html

Günümüze gelindiğinde ise, ontolojiler hususunda kavramların ve onları temsil eden kelimelerin birbirleri ile olan Şekil 1'de örnek olarak verdiğimiz türde ve bunun çok daha ötesindeki hiyerarşik bağlantılarından yola çıkarak, pek çok sınıflandırma ve bölümleme meydana getirilmiştir. Bilişim teknolojilerinin ve özellikle YSA'larının da bu alanda kullanımı sayesinde de ontolojiler oluşturabilir hale gelmiştir. Aslında tüm bu bağlantıların temelleri ise taksonomi (sınıflandırma) mantığına dayanmaktadır. Yukarıda verdiğimiz Şekil 1'de, sadece basit bir sınıflandırma görülmektedir. Örneğin, **Hayvan** kavramı **Balık**, **Kuş** ve **Köpek** kavramlarının üst sınıfını (hipernym) temsil etmektedir. Bu durumda **Balık**, **Kuş** ve **Köpek** ise **Hayvan** kavramının alt sınıfı (hyponym) konumundadır ve **Hayvan** kavramı ile **Balık**, **Kuş** ve **Köpek** kavramları dikey bir ilişki içindedir denilebilir. Yine **Balık** kavramına yatay ilişkide olduğu düşünülen **Kuş** ve **Köpek** kavramları da **Hayvan** kavramının alt sınıfı olduğunu söylemek mümkündür. Ancak kelime ilişkileri her zaman yukarıda bahsedildiği gibi basit değildir. Örneğin, **Wordnet**'te isimleri sınıflandırırken kelime ilişkilerinin en önemli dayanağı olan synsets (synonym sets) olmasının yanı sıra, Tablo 1'de görüldüğü üzere üst sınıf (hypernym), alt sınıf (hyponym), vb ilişkiler haricinde de pek çok anlamsal ilişkiden söz edildiği göze çarpar (Daniel ve James, 2018).

Tablo 1: WordNet Kelime İlişkileri (isimler)

Relation	Also Called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	*breakfast*^{1} → *meal*^{1}
Hyponym	Subordinate	From concepts to subtypes	*meal*^{1} → *lunch*^{1}
Instance Hypernym	Instance	From instances to their concepts	*Austen*^{1} → *author*^{1}
Instance Hyponym	Has-Instance	From concepts to concept instances	*composer*^{1} → *Bach*^{1}
Member Meronym	Has-Member	From groups to their members	*faculty*^{2} → *professor*^{1}
Member Holonym	Member-Of	From members to their groups	*copilot*^{1} → *crew*^{1}
Part Meronym	Has-Part	From wholes to parts	*table*^{2} → *leg*^{3}
Part Holonym	Part-Of	From parts to wholes	*course*^{7} → *meal*^{1}
Substance Meronym		From substances to their subparts	*water*^{1} → *oxygen*^{1}
Substance Holonym		From parts of substances to wholes	*gin*^{1} → *martini*^{1}
Antonym		Semantic opposition between lemmas	*leader*^{1} ⟺ *follower*^{1}
Derivationally Related Form		Lemmas w/same morphological root	*destruction*^{1} ⟺ *destroy*^{1}

(Daniel ve James, 2018: 6)

Kaynak: Daniel J. ve James H. M. (2018 : 6).

Yine aynı durum, fiillerin ilişkileri konusunda da söylenebilir (Tablo 2). WordNet'te synsets dışında da fiiller için oldukça detaylı ve isimlere kıyasla biraz farklı bir sınıflandırma şeklinde görülür.

Tablo 2: WordNet Kelime İlişkileri (fiiller)

Relation	Definition	Example
Hypernym	From events to superordinate events	*fly*[9] → *travel*[5]
Troponym	From events to subordinate event (often via specific manner)	*walk*[1] → *stroll*[1]
Entails	From verbs (events) to the verbs (events) they entail	*snore*[1] → *sleep*[1]
Antonym	Semantic opposition between lemmas	*increase*[1] ⟺ *decrease*[1]
Derivationally Related Form	Lemmas with same morphological root	*destroy*[1] ⟺ *destruction*[1]

(Daniel ve James, 2018: 6)

Kaynak: Daniel J. ve James H. M. (2018 : 6).

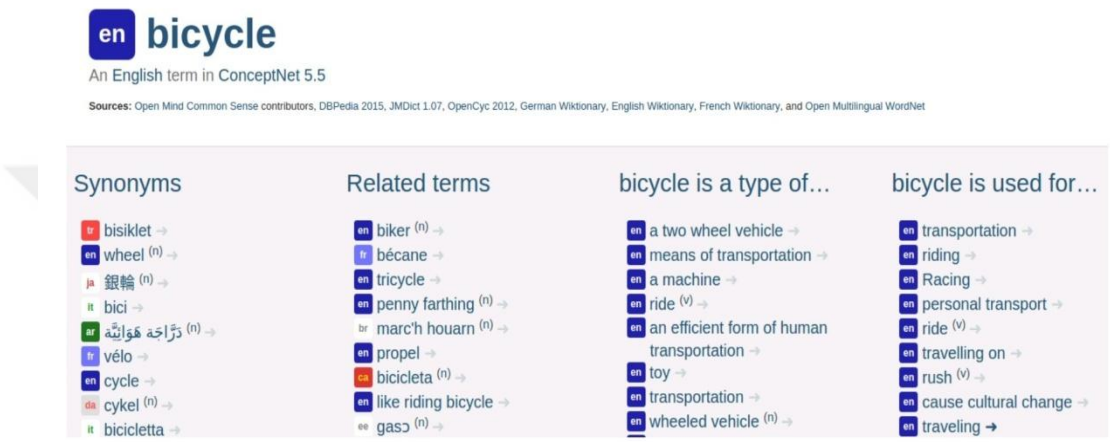
Ancak WordNet'in temel yapısı, yukarıda da bahsedildiği üzere esasen **synsets** (**synonym sets**) adı verilen anlamsal benzerliklere dayanmaktadır. Kelime formları arasındaki ilişkiyi yargılama yeteneği, anlamların sözcüksel bir matriste temsil edilmesi için bir ön koşul oluşturmakta ve bağlamdaki kelimelerinin bu matristeki eşdeğerlikler ile değiştirilmesi sonucunda, cümlenin gerçek değeri ve anlamı değişmiyor ise matristeki bu iki ifadenin birbiri ile eş anlamlı olduğu kabul edilmektedir (Miller ve diğerleri, 1990).

Diğer bir yandan, NeOn Projesi kapsamında geliştirilen uygulama ve API'lerde olduğu gibi ontoloji konusunda oluşturulan daha pek çok uygulamanın arka yüzünde de faydalanıldığı görülen WordNet, başlangıç noktasındaki ana amacının oldukça ilerisinde sonuçlar elde etmiştir. 1985'ten bu yana sözcük ilişkileri ve ontolojik bağlantılar başta olmak üzere, günümüzde de halen doğal dil işleme alanında geliştirilen pek çok uygulama ve platformda kullanımda olduğu görülmektedir.

Yine bu bağlamda bahsedilmesi gereken bir diğer yapı ise ConceptNet'tir. Web tabanlı bir platform olarak "https://conceptnet.io/" adresinden erişilebilen ConceptNet, WordNet başta olmak üzere Wikidata, Wiktionary, DBpedia, vd. gibi birden fazla platformun veritabanını kullanmakta ve böylece kelimelerin aynı kavram çerçevesinde eşanlamlılığı ve yakın anlamlılığı başta olmak üzere bağlantılı anlamlılık ve daha pek çok farklı çıkarım ve türevlerini farklı dillerde de verebilen oldukça geniş çaplı bir mimariye sahip olarak hizmet vermektedir. Robyn ve

Catherine (2012: 3681)'e göre ConceptNet 5 versiyonu, Nisan 2012 tarihi itibariyle, yaklaşık 12 milyon İngilizce kavramı bünyesinde barındırmaktadır. Ayrıca İngilizce'nin yanı sıra Çince, Portekizce, Japonca, Fransızca başta olmak üzere ve hatta 2023 yılı itibariyle Türkçe bile günümüzde temsil edilen diller arasında ilk giriş sayfasındaki arama kısmındaki seçeneklerde görülmektedir.

Tablo 3: ConceptNet 5.5



Kaynak: Robyn ve diğerleri (2017: 4445).

Yukarıda Tablo 4'te günümüzde de aktif versiyon olarak kullanımda olan ConceptNet 5.5 versiyonunda yapılan bir arama örneğinin sonucu görüntülenmektedir. İngilizce'de **bisiklet** anlamına gelen **bicycle** sözcüğü arandığında sonuç ekranından da anlaşılacağı üzere bu sözcük ile ilişkili olan ya da benzerliği bulunan diğer pek çok kelime de karşımıza gelir. Ayrıca aradığımız bu kelimenin ne tür bir varlık olduğu ve ne amacı karşıladığı gibi bilgiler ve daha fazlası hakkında da bilgiler Tablo 4'te görüldüğü gibi sonuç ekranında görüntülenir. Ancak en temel amacının kelime benzerlik bağlantılarından faydalanarak **bicycle** kavramının ne olduğunu kavratmak olduğu söylenebilir.

Aslında buraya kadar yukarıda açıkladığımız çalışmaların tamamı, ontoloji çalışmalarına dayanan ve "Bilgi Tabanlı Teknikler (Knowledge-based techniques)" kapsamında ele alınabilir çalışmalardır. Lofi (2015: 493)'te kelimeler arasındaki benzerlik ve ilişkiyi değerlendirmede 2 farklı tip teknikten söz edilmekte olup, bunlar belirli bir ontolojiye veya taksonomiye dayanan "Bilgi Tabanlı Teknikler

(Knowledge-based techniques)" ile büyük bir doğal dil metni külliyatı kullanan "Derlem Tabanlı Teknikler (Corpus-based techniques)" olarak ayrılmaktadır. Bunlardan ikincisi olan derlem tabanlı teknikler ise kelime vektörleri de dâhil[4], Dağılımsal Yaklaşımlar (Distributional Approaches)'a dayandırılmaktadır (Lofi, 2015: 493-495).

Bilgi Tabanlı tekniklere dayanan yaklaşımlar, mevcut bir ontoloji veya taksonomiden yararlanmakta ve bu tür ontolojiler ise gerek zaman gerek iş gücü açısından çok maliyetli ve genellikle manüel ya da yarı manüel olan işleme süreçlerinden oluşmaktadır. Bu nedenle, WordNet veya Freebase gibi çoğunlukla elle oluşturulmuş olanların, ya da DBpedia veya Yago gibi manüel kaynaklardan otomatik bilgi çıkarımına dayanan ontolojilerin yalnızca birkaç alanı, benzerlik ölçümü uygulanabilirliğine sahip olmaktadır (Lofi, 2015: 493). Bu durum ise elde edilmek istenen kelime benzerlikleri ya da ilişkilerine kapsam açısından düşük bir kalite ve sonuç ile yansıyacaktır.

Diğer bir yandan ontolojiler, her kavramı, aralarındaki ilişkileri temsil eden diğer kavramlara kenarlarla (edge) bağlanan bir düğüm (node) olarak temsil eder ve bilgiye dayalı benzerlik ölçütleri de taksonomik bilgiler ışığındaki kavram hiyerarşisine dayanır. Bir taksonomide iki kavram, mesafe olarak birbirine daha yakınsa daha benzerdir denilebilir ve benzerlik ölçütü ise kelimeleri ayıran düğüm veya kenar sayısına bağlı olarak basit bir ölçümle elde edilir. Ancak bu tür teknikler, taksonomik ilişkilerin homojen olmayan ayrıntı düzeyini göz ardı ettikleri için genellikle düşük sonuç kalitesi sağlarlar (örneğin, "hayvan" ile "canlı" arasındaki anlamsal mesafe, "kaniş" ile "köpek" arasındaki mesafeden önemli ölçüde daha büyüktür) (Lofi, 2015: 493).

İşte tüm bu yukarıdaki sebeplerle birlikte, hedeflenen çıktıya göre elde edilmek istenilen kelime ilişkilerinin ilerleyen süreçlerdeki güncelleme maliyetleri de göz önünde bulundurulduğunda, ontoloji ve taksonomiden yararlanan yöntemler haricinde farklı yöntem ve teknikler aranmaya başlanmış ve neticesinde bilgi tabanlı

[4] Lofi (2015)'te Derlem Tabanlı yaklaşımların tamamı her ne kadar Dağılımsal Hipotez'e dayandırılsa da, çalışmasının ilerleyen aşamasında bunları sadece kelimelerin bir arada bulunmaları sayımına dayandırdığı "Basit Dağılımsal Yaklaşımlar (Simple Distributional Approaches)" ve genellikle yapay sinir ağları ile boyutsal indirgeme (dimensionally reduction) tekniklerinden faydalanan "Yoğun Vektör Temsilleri (Dense Vector Representations)" olarak iki gruba daha ayırmaktadır. Ancak bu çalışmada bu tür bir ayrım gözetmeksizin, Derlem Tabanlı çalışmaların tamamının "Dağılımsal Yaklaşımlar (Distributional Approaches)" altında ele alınması uygun görülmüştür.

ontolojiler yerine derlem tabanlı Dağılımsal Anlambilim Modelleri'nden faydalanılma yoluna gidilmiştir.

Dağılımsal Anlambilim Modelleri, Amerikan yapısalcı dilbilimin öncülerinden olan Harris (1954)'in Dağılımsal Hipotez (Distributional Hypothesis)'ine dayanmaktadır. Harris'in kelime benzerliklerinin ve anlamlarının, kelimelerin kullanıldığı bağlamdaki diğer kelimelerden çıkarılabileceğini ortaya atmasıyla, bu alanda yepyeni bir yol açılmıştır (Harris, 1954: 158-159). Harris'in bu modeli ise günümüzde kullanılan Word2vec ya da GloVe gibi kelime vektörleri uzayı çalışmalarına da ışık tutmuştur.

İKİNCİ BÖLÜM
DOĞAL DİL İŞLEME BAĞLAMINDAKİ KURAMSAL ÇERÇEVE

İnsanoğlunun kullandığı dilin makineler tarafından anlaşılması konusundaki çalışmaların başlangıç noktası aslında oldukça eskilere dayanmaktadır. Bu konudaki çalışmaların temeli hem bilişsel bilimlere, hem matematik ve istatistiğe, hem de dilbilime dayanmaktadır. 1950'li yıllardan itibaren gelişmeye başlayan modern bilgisayar bilimlerinin de özellikle istatistik ve matematik ilminin insan dilini analiz etme sürecine dahil olmasıyla birlikte, tüm bu bilim alanlarının bir nevi kesişimi neticesinde ortaya çıkan yeni disiplinlerarası alan ise Bilgisayarlı Dilbilim – BD (Computational Linguistics – CL) olmuştur. Bu alandaki çalışmalara Makine Öğrenmesi, Derin Öğrenme ve özellikle Yapay Sinir Ağlarındaki gelişmeler ile birlikte Yapay Zeka çalışmaları ve Veri Biliminin de dahil olması neticesinde ise Doğal Dil İşleme – DDİ (Natural Language Processing – NLP) alanının geliştiği söylenebilir. Her ne kadar bazı durumlarda bağımsız disiplinlerarası bir çalışma alanı olarak karşımıza çıksa da, geniş bir perspektiften bakıldığında aslında DDİ'yi, BD'nin bir alt alanı olarak ele almak mümkündür (Tsujii, 2021: 709).

2.1. DOGAL DİL İŞLEMENİN KAPSAM VE KATEGORİLERİ

Adalı (2020: 19)'da DDİ'nin çalışma alanları kapsamında aşağıdaki konuların verildiği görülür:

- Yazım yardım araçlarının geliştirilmesi
- Metni anlama amacıyla gerçekleştirilen çalışmalar
- Bilgisayarla sesli etkileşim üzerine çalışmalar
- Bilgisayarlı çeviri alanındaki çalışmalar
- Yabancı dil okuma yardımcı araçları

Ancak, elbette bunları çoğaltmak mümkündür. Örneğin, bu tezin amacı düşünüldüğünde, yukarıdaki DDİ çalışma konularına "Çeviri yardımcı aracı geliştirme" şeklinde bir ekleme yapmak gerekir.

DDİ'nin alt kategorilerine bakıldığında ise, genellikle Doğal Dil Anlama - DDA (Natural Language Understanding – NLU) ve Doğal Dil Üretme – DDÜ

(Natural Language Generation – NLG) olarak ikiye ayrıldığı görülmektedir (Kavlakoğlu, 2020; Chethan Kumar, 2018). Bunlardan DDA, insanoğlunun konuştuğu dili anlamayı ve anlamlandırmayı amaç edinir ve bu amaçla konuşma ve söylemleri, yazılı ve sözlü metinleri, cümleleri ve kelimeleri ses, şekil, yapı, anlam, vb. açıdan analiz eder. Ayrıca ontolojiler ve kelime benzerlikleri başta olmak üzere, kelime ve kelime öbekleri arasındaki çeşitli ilişkiler de yine DDA'nın alanına girmektedir. DDÜ'de ise otomatik chatbot robotlarından, makale ya da roman yazma becerisi gösterebilen dil çıktıları üretebilen dilsel bir çıktının üretimi söz konusudur. Ancak elbette bu tür çıktıların üretimi için ya modelin önceden eğitilmiş olması, ya da mevcut veri girdisi ile belirli bir öğrenmenin sağlanması gereklidir. Dolayısıyla DDÜ'den söz edebilmemizin temelleri de aslında öncesinde hazırlanan DDA modelinin işletimine bağlı olmaktadır.

2.2. ANLAMSAL BENZERLİK VE DAĞILIMSAL HİPOTEZ

Yeni kelimeleri öğrenirken bu kelimelerin temelini oluşturan kavramları, kelimeye ihtiyaç duyulduğunda zihnimizden çağırıp tekrar doğru bir şekilde kullanabilmek amacıyla zihnimizde bulduğu varsayılan zihin sözlüğümüze (mental lexicon) kayıt ederiz. Ancak elbette ki bir kelimenin temelinde yatan bir kavramı algılamak o kadar da basit bir işlem değildir. Doğal dil algısı, çocukken çoğumuzun farkında olmadan geliştirdiği bir durum iken, ileri yaşlarda yeni bir kelimeyi öğrenmek bizler için oldukça zahmetli bir duruma dönüşmektedir. Ancak her iki durumda da, çevremizde bulunan somut nesneler başta olma üzere soyut kavramları da bir takım işlemler neticesinde anlamlandırdığımız bilinmektedir. İnsan zihninde anlamlanan bu yeni kavramları oluşturma sürecinde ise insanoğlu farklı obje ve olguların değişebilen ortak özelliklerine ilişkin bilgileri toplayarak meydana getirebilmektedir (Ülgen, 2004: 107). Diğer bir deyişle yeni bir kelimeyi zihin sözlüğüne yazarken bu kelimenin bağlı olduğu kavramı oluşturma aşamasında diğer kavramlardan farklı yönlerinin yanı sıra özellikle diğer kavramlarla benzer özelliklerinden faydalanarak onu anlamlandırır. Bu durum somut bir örnekle açıklanacak olursa, örneğin gözümüz bağlı bir durumda elimize bir nesne verildiğini ve bunun ne olduğunu algılamamız istenildiğini varsayalım. Öncelikle bu nesnenin

sert mi katı mı, metal mi plastik mi ya da köşeli mi yuvarlak mı gibi daha pek çok soruyu zihnimizde oluşturarak bu nesneye sorduğumuz görülecektir. Neticesinde ise, bu sorulara zihnimizde vereceğimiz yanıtlara göre nesnenin ne olabileceği ya da neye benzediği hususunda fikir sahibi oluruz. Aslında daha önce duymadığımız yeni bir kelimeyi öğrenme aşamasında da benzer bir yol izlenildiği düşünülmüştür. Bu hususta oldukça kabul gören hipotezlerden biri Amerikan yapısalcılığın dilbilimin öncülerinden biri olan Zellig S. Harris tarafından 1954'te ortaya atılmıştır. Dağılımsal Hipotez (Distributional Hypothesis) olarak adlandırılan bu hipotez ise aşağıdaki şekilde özetlenebilir:

> İki dilsel ifade A ve B arasındaki anlamsal benzerliğin derecesi, A ve B' nin görünebileceği dilsel bağlamların benzerliğinin bir fonksiyonudur. (Lenci, 2008: 3; Harris, 1954: 158-161)

Buradan hareketle, dağılımsal anlambilim modellerinin **anlamsal benzerlik** kavramı üzerine inşa edildiğinden bahsetmemiz mümkündür. Bu sebeple, Dağılımsal Hipoteze göre, sözcüksel ifadelerin anlamının en azından belirli yönlerinin, bu ifadelerin dağılım özelliklerine, yani gözlemlendikleri dilsel bağlamlara bağlı olduğu söylenebilir (Lenci, 2008: 3). Somut bir örnek olarak, anlam bakımından benzer olan **tekne** ve **gemi** gibi kelimelerin; **yelken**, **deniz** ve **denizci** gibi sözcüklerin bulunduğu bağlamlarda ortaya çıkması verilebilir (Mitchell ve Lapata, 2010: 1389).

Dağılımsal anlambilim analizlerinin ilk başlarda iki boyutlu olarak karşımıza çıktığı görülür. Genellikle belirli bir bağlamda bir kelimenin kaç kez çıktığının istatistiksel hesaplamasına dayanmaktadır. Örneğin, yukarıdaki örnekten yola çıkılacak olursak, **tekne** sözcüğünün sıklıkla kullanıldığı bir makale düşünün, denizcilik ile ilgili bir alanda yazılmış bir makale ise muhtemelen içerisinde **yelken**, **deniz** ya da **denizci** kelimelerine de yer verilecektir. Bu tür kelimeler ile tekne kelimesinin aynı bağlamda ortaya çıkma sıklığı genellikle aşağıda da görülen PMI (Pointwise Mutual Information) adı verilen Noktasal Karşılıklı Bilgi formülü ile hesaplanmaktadır. Ancak elbette kelimelerin birlikte görülme sıklığı hesaplanırken negatif değerler çıkmaması için PPMI (Positive Pointwise Mutual Information) adı verilen Pozitif Noktasal Karşılıklı Bilgi gibi farklı formüller de uygulanabilmektedir.

Şekil 2: Noktasal Karşılıklı Bilgi (PMI) Formülü

$$PMI(w_1, w_2) = log_2 \frac{p(w_1 \& w_2)}{p(w_1) * p(w_2)}$$

PMI (W_1,W_2) : W_1 ve W_2 kelimeleri için noktasal karşılık bilgi katsayısı.
P (W_1 & W_2) : W_1 ve W_2 kelimelerin birlikte geçme olasılıkları.
P (W_1) : W_1 kelimesinin dokümanlar içerisinde geçme olasılığı.
P (W_2) : W_2 kelimesinin dokümanlar içerisinde geçme olasılığı.

Kaynak: Wawer (2012) ile Özdoğan ve Turan (2019)' a dayanarak oluşturulmuştur.

Burada bahsedilen "Karşılıklı Bilgi (MI)" nedir sorusuna yanıt olarak Fano (1961: 28), W_1 ve W_2'yi birlikte gözlemleme olasılığının, W_1 ve W_2'yi bağımsız olarak gözlemleme olasılıklarıyla karşılaştırımı olarak yorumlar. Yine Church ve Hanks (1990: 77-78)'de, W_1 ve W_2 kelimelerinin bir arada bulunmaları ile ilgili ilginç bir noktaya dikkat çekilerek, bir derlemde W_1'den sonra W_2'nin gelmesi neticesinde elde edilecek PMI değeri ile W_2'den sonra W_1'nin gelmesi neticesinde elde edilecek PMI değerlerinin farklı olacağından bahsedilmektedir. Bu durum Church ve Hanks (1990: 78)'de aşağıda Şekil 3'teki gibi **doctor** (doktor) ve **nurse** (hemşire) kelimeleri üzerinden ele alınmıştır.

Tablo 4: Kelime Sıralamasının PMI Üzerine Etkisi

Some Interesting Associations with "Doctor" in the 1987 AP Corpus (N = 15 million)

I(x, y)	f(x, y)	f(x)	x	f(y)	y
11.3	12	111	honorary	621	doctor
11.3	8	1105	doctors	44	dentists
10.7	30	1105	doctors	241	nurses
9.4	8	1105	doctors	154	treating
9.0	6	275	examined	621	doctor
8.9	11	1105	doctors	317	treat
8.7	25	621	doctor	1407	bills
8.7	6	621	doctor	350	visits
8.6	19	1105	doctors	676	hospitals
8.4	6	241	nurses	1105	doctors

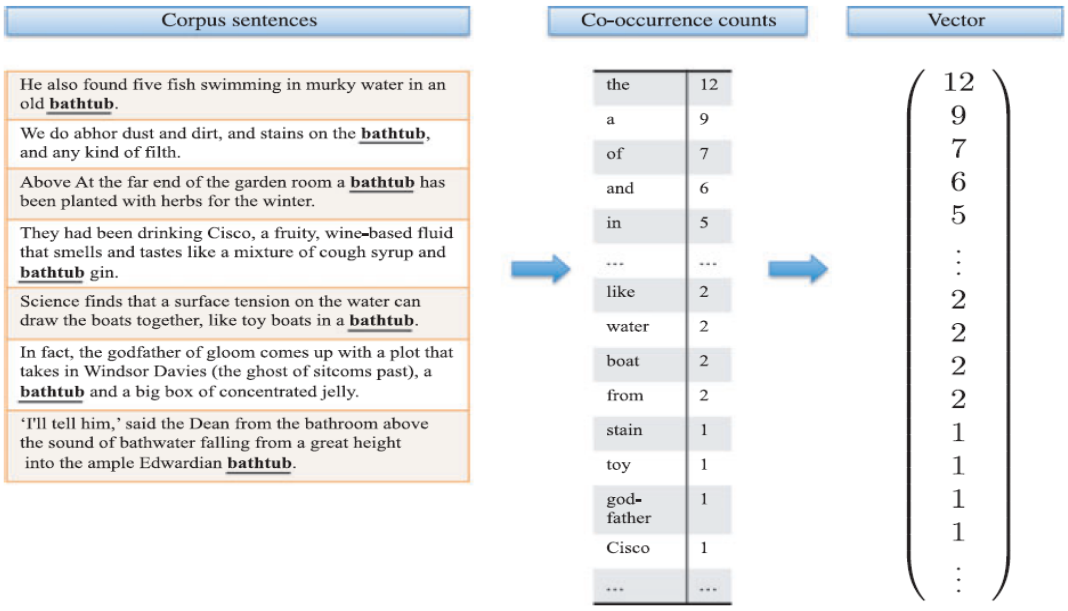
Burada; I(x,y) = PMI (W_1,W_2) ve f(x,y) = P (W_1 & W_2)'dir.

Kaynak: Church ve Hanks (1990: 78)

Burada 15 milyon kelimeden oluşan İngilizce bir derlem içerisinde **doktor** ve **hemşire** kelimelerinin bir arada görülme durumları incelenmiştir. Sonuç olarak **doktor** kelimesinin ardından **hemşire** kelimesinin aynı bağlamda görülmesinin, **hemşire** kelimesinin ardından **doktor** kelimesi gelmesinden daha yüksek bir olasılığa sahip olduğunu ve dolayısıyla da daha yüksek PMI değerine sahip olduğunu göstermiştir. Bunu daha basit bir şekilde açıklayacak olursak, bir derlemdeki cümlelerin içerisinde "Peynir" ve "Ekmek" sözcüklerinin sırasıyla bir arada bulunma olasılığı, "Ekmek" ve "Peynir" sözcüklerinin derlem içerisindeki cümlelerde sırasıyla bulunma olasılığından daha yüksek olması beklenen bir durum olup, bunun neticesinde PMI (Peynir, Ekmek) > PMI (Ekmek, Peynir) değeri şeklinde olacaktır. Bunun sebebi ise kelime kullanımlarımızda "peynir, ekmek alda gel" cümlesinde olduğu gibi "peynir" sözcüğünün "ekmek" sözcüğünden önce kullanımının daha yaygın ve kabul gören bir sıralama olmasından kaynaklanır.

Diğer bir yandan, geliştirmekte olduğumuz bu tez çalışmasında, kullanıcıdan alınacak anahtar kelimelerden yola çıkılarak kelime benzerliklerine, diğer bir deyişle verilen anahtar kelimeler ile bağlantılı olan diğer kelime gruplarına ulaşmak amaç edinildiğinden bu tür bir kelime sıralamasının şu aşamada çok da önemli olmadığı düşünülebilir. Ancak bu husus ilerleyen aşamalarda yeniden değerlendirilecektir.

Tablo 5: Derlemdeki Kelimeler

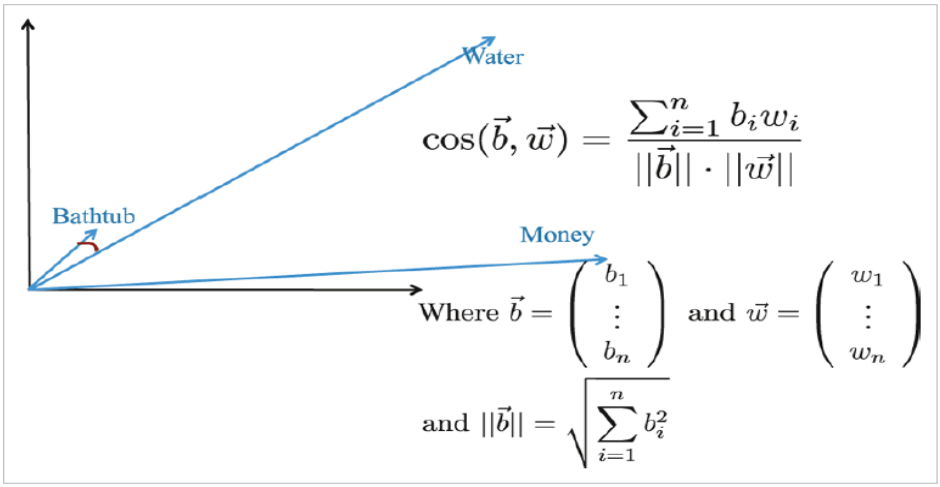


Kaynak: Erk (2012: 637)

Tablo 5'te, İngiliz Ulusal Derlemi'nden faydalanılarak gerçekleştirilen dağılımsal anlambilim analizlerinin vektörel büyüklüklere dönüştürülmesi aşamasında öncelikle derlemdeki kelime sayılarının hesaplaması yapılmaktadır. Sonrasında ise Erk (2012: 635), Harris (1954) ve Firth (1957)'den yola çıkarak, eğer iki kelime benzer bağlamlarda ortaya çıkma eğilimindeyse, anlamlarının benzer olduğunun varsayılabilineceğinden ve ayrıca dağılım modellerinin tipik olarak sözcük temsilinin yüksek boyutlu bir uzayda noktasal olarak vektör uzay modelleri ile gösterilebilineceğinden söz eder. Bu uzayda, bağlamda birlikte geçen sözcükler vektör boyutlarını ve koordinatlar ise birlikte bulunma sayılarını oluşturmaktadır. Dolayısıyla, hangi hesaplama tekniği ile elde edilirse edilsin, bağlamdaki kelime benzerlikleri vektör uzayına da benzer şekilde yansıyarak temsil ettikleri kelimeler arasında da görsel bir yakınlık oluşturmaktadır. Bu tür benzerlik hesaplamalarında ise sıklıkla kosinüs benzerliğinden faydalanıldığı görülür. Lowe (2001), bir vektör uzayını tanımlarken demet kavramını kullanır ve bunu ⟨A, B, S, M⟩ ile açıklar. Burada, B bağlam seti elementlerini diğer bir deyişle kelime boyutunu, S vektörler arası benzerliği; A noktasal ortak bilgideki gürültüyü azaltmak için ham birlikte oluşum sayımlarını ilişkilendirme ağırlıkları değiştiren parametresel formülü ve M de tüm vektör uzayının bir dönüşümünü diğer bir deyişle tipik bir boyut indirgemesini ifade etmektedir (Lowe, 2001; Erk, 2012: 636 - 637).

Aşağıda Şekil 3'te ise Tablo 5'in devamı olarak, derlemdeki bu kelimelerin sıklık hesabı yapıldıktan sonra nasıl vektörel büyüklüklere dönüştürüldüğü gösterilmiştir. "Bathtub (küvet)" ve "Water (su)" arasındaki benzerlik, "Bathtub (küvet)" ve "Water (su)" vektörleri arasındaki açının kosinüsü olarak hesaplanır ve buna "küvet" ile "su" kelimelerinin kosinüs benzerliği denir. Burada "su" kelimesi derlem bütününde "küvet" kelimesinden daha sık geçtiğinden dolayı daha büyük bir vektörel uzunluğuna sahip olsa da bu durum göz ardı edilerek benzerlik hesaplaması yapılmıştır (Erk, 2012: 637). Buna karşın "Water (su)" kelimesi, aynı derlem genelinde geçen "Money (para)" kelimesi ile kıyaslandığında, benzer bir sıklığa sahip olduğundan yakın bir vektörel büyüklüğe sahiptir. Ancak, ilişkisel açıdan bakıldığında, "küvet" sözcüğü ile "para" sözcüğünün aynı bağlamda görülme olasılığı daha düşük olduğundan, doğal olarak "Water (su)" ile "Money (para)" sözcüğünün benzerliği daha uzak olacaktır ve bu da vektör uzayına Şekil 3'te görüldüğü gibi yansıyacaktır.

Şekil 3: Vektörel Benzerlik



Kaynak: Erk (2012: 637)

Kelime benzerliklerini hesaplamada, vektörel büyüklükler değil, o vektörlerin yönelimleri arasındaki açıya göre yakınlığı bizlere benzer olup olmadıklarını, diğer bir deyişle aynı çevresel bağlamda noktasal olarak bulunup bulunmadıklarını bizlere verir. Bu sebeple, vektörel büyüklükleri ön plana çıkaran öklid mesafesi yerine, kosinüs benzerliğini hesaplamak oldukça sıklıkla faydalanılan bir yöntem olarak karşımıza çıkar. Şekil 3'te de bu sebeple iki kelime arasındaki benzerliği hesaplarken kosinüs benzerliğinden faydalanılmıştır. Kosinüs benzerliğinin öklid mesafesi ve türevlerinden (Karesel öklid, Karl Pearson, vb) en ayırt edici özelliği, kosinüs benzerliğinde frekansı düşük olan, diğer bir deyişle küçük vektörel büyüklüğe sahip nadir kelimelerin benzerliklerini de tespit etmenin mümkün olmasıdır[5]. Özellikle alana yönelik terimler söz konusu olduğunda bu durum bir kat daha öneme sahip olduğundan, bu çalışmada da kelime benzerlikleri konusunda kosinüs benzerliğine dayanan kelime benzerliği metriğinden faydalanılmıştır.

Dolayısıyla, Harris'in 1954'te ortaya attığı ve Firth (1957)'nin geliştirdiği *iki kelime benzer bağlamlarda ortaya çıkma eğilimindeyse, anlamlarının da benzer olduğunun varsayılabileceği ilkesi*ni temel alan dağılımsal hipotezden yola çıkarak, aynı bağlamda benzer kelimelerin kullanılabilirliği; diğer bir deyişle, tersine mühendislikle kelime bazlı düşünülecek olduğunda, örneğin ilk verdiğimiz **yelken**, **deniz** ve **denizci** gibi anahtar kelimeler vasıtasıyla bu kelimelerin mevcut olduğu

[5] https://www.quora.com/Why-is-similarity-between-words-usually-calculated-via-cosine-metric-not-Euclidean

bağlamda kullanılma olasılığı yüksek olan **tekne** yahut ona anlamsal yönden benzeyen **katamaran** ya da **monohull** gibi kelimelere ulaşabilirlik, bu çalışmanın da temel çıkış noktasını oluşturmaktadır.

2.3. DAĞILIMSAL ANLAMBİLİM MODELLERİ

Harris (1954)'in Dağılımsal Yapısalcı (Distributional Structure) modelinden sonra oldukça farklı yönlerde araştırmalar yapılmıştır. Bunlardan bir kısmı, Harris'in öğrencisi Chomski ile devam ettirilmiş ve günümüzde Evrensel Dilbilim adı altında incelenen alt alanı oluşturmuştur. Bir kısmı ise kelime anlamları ve bunların benzerlik analizleri konusuna odaklanarak devam ettirilmiş ve bu alanda pek çok çalışma yapılarak farklı dil modelleri geliştirilmiştir. Günther ve diğerleri (2019: 1009)'da ortaya çıkan aşağıdaki modellerin Harris'in ortaya attığı yapısalcılıktan yola çıkılarak geliştirilen aşağıdaki modellerin Dağılımsal Anlambilim Modeli – DAM (Distributional Semantic Model – DSM [6]) kapsamında ele alınabileceğinden bahsedilmektedir.

Tablo 6: Dağılımsal Anlambilim Modelleri – DSM

Model	References	Venue of publication	Description
HAL	Lund & Burgess (1996)	*Behavior Research Methods*	Creates a word-by-word matrix
LSA	Landauer & Dumais (1997)	*Psychological Review*	Creates a word-by-document matrix and applies dimensionality reduction via SVD
Topic models	Griffiths, Steyvers, & Tenenbaum (2007)	*Psychological Review*	Creates a word-by-document matrix and applies dimensionality reduction via LDA
BEAGLE	Jones & Mewhort (2007)	*Psychological Review*	Modifies initially random word vectors based on the other words in successively processed documents
word2vec	Mikolov, Chen, Corrado, & Dean (2013); Mikolov, Sutskever, Chen, Corrado, & Dean (2013)	*ICLR Workshop, Neural Information Processing Systems*	Trains word vectors as the hidden layer of a neural network that predicts words from the surrounding words or vice versa
GloVe	Pennington, Socher, & Manning (2014)	*Empirical Methods in Natural Language Processing*	Trains word vectors to optimally predict words' probability of co-occurrence

Note: More detailed descriptions are provided in the text. BEAGLE = bound encoding of the aggregate language environment; GloVe = global vectors; HAL = hyperspace analogue to language; LDA = latent Dirichlet allocation; LSA = latent semantic analysis; SVD = singular value decomposition.

Kaynak: Günther ve diğerleri (2019: 1007)

[6] Günther ve diğerleri (2019)'da aynı konu "Distributional Semantic Model – DSM (Dağılımsal Anlamsal Model)" olarak adlandırılmaktadır ancak Lenci (2008)'da "Distributional Semantics (Dağılımsal Anlambilim)" adı altında ve ayrıca Mitchell ve Lapata (2010)'da da "Distributional Models of Semantics (Anlambilimin Dağılımsal Modelleri)" şeklinde geçmekte olduğundan Türkçe'ye "Dağılımsal Anlamsal Model" yerine "Dağılımsal Anlambilim Modeli - DAM" şeklinde adlandırılarak aktarımı tercih edilmiştir.

Aşağıda bu modeller alt başlıklar altında ele alınarak temel işlevleri ve nasıl çalıştıkları açıklanmaya çalışılmıştır.

2.3.1. HAL

Hiperuzay Analogdan Dile (HAL) modeli 1996 senesinde Lund ve Burgess tarafından deneysel olarak ortaya atılmış bir yöntemdir. Şubat 1995'te haber gruplarından oluşan 160 milyon kelime metinli Usenet derlemi kullanılarak gerçekleştirilen çalışma, metin gövdelerinden yüksek boyutlu anlamsal alanların otomatik bir şekilde oluşturulabileceğine ilişkin semantik hafıza modeli prosedürü sunar ve bu alanların kavram benzerliğini nasıl modelleyebileceğini gösteren deneyler içerir (Lund ve Burgess, 1996). Bağlamı yalnızca belirli bir kelimeyi hemen çevreleyen kelimeler olarak kabul eden HAL, metin külliyatında artımlı olarak hareket eden 10 kelimelik bir pencere genişliği içerisindeki kelime birlikteliklerini Minkowski metriğini baz alarak hesaplar (Lund ve Burgess, 1996; Melli, 2021).

Ayrıca Lund ve Burgess'in yapmış olduğu bu deneysel çalışma, HAL'ın geliştirdiği temsiller sayesinde, kavramı çevreleyen semantik alanın, bir kelimenin diğer kelimelerle olan ilişkilerinde önemli rol oynayan "en yakın komşular" olarak gösterilebildiği ve vektörlerin, üst kategori örneklerini sınıflandırmada kullanılabileceği gibi mikroyapısal semantik için gerekli olabilecek bazı temel etkileri ortaya koymuştur.

2.3.2. LSA

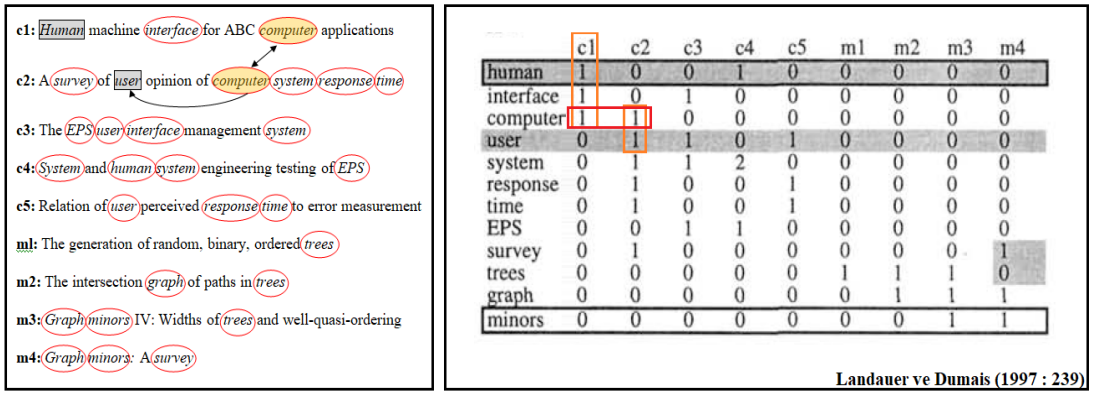
Gizli Anlamsal Analiz (LSA – Latent Semantic Analysis), temeli dağıtımsal anlambilime dayanan ve belgeler ve içerdikleri terimler arasındaki gizli ilişkileri analiz ederek ortaya çıkarabilen bir doğal dil işleme tekniğidir. Patenti, aslen 1988'de mucitleri olan Landauer, Deerwester, Dumais ve diğerleri tarafından *Computer information retrieval using latent semantic structure* [7] adı ile *Bell Communications Research, Inc.* firması adına alınmıştır. LSA'da bağlamda bulunan dokümanlar bir matrise dönüştürülerek tüm bağlam üzerinde bir dağılım matrisi oluşturulur ve daha

[7] https://patents.google.com/patent/US4839853A/en

sonra her dokümanın elemanlarının bu bağlam içerisindeki sıklıkları hesaplanarak SVD (Singular Value Decomposition) adı verilen Tekil Değer Ayrışımı'na tabii tutularak matris boyut indirgemesi yapılır. Hesaplama yöntemi olarak ise TF-IDF ve kosinüs benzerliğinden sıklıkla faydalanılır.

Yukarıdaki işlemler ve matris vasıtasıyla aslında 2 boyutlu bir haritalandırma yapılmaktadır. Bu duruma ilişkin, patent metni ve Landauer ve Dumais (1997)'de de ele alınan bir örnek aşağıda verilmiştir.

Şekil 4: Gizli Anlamsal Analiz (LSA)[8]



	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Landauer ve Dumais (1997 : 239)

Kaynak: Landauer ve Dumais (1997: 238-239) yazar tarafından modifiye edilmiştir.

Yaptığımız eklemeler sayesinde aslında Şekil 4, pratikte LSA'ın ne şekilde çalıştığı hususunda en kaba ve basit haliyle açıklayıcı bilgiler vermektedir. Burada **human** ve **user** kelimeleri aynı doküman içerisinde kullanılmamış olsa da gizli olarak bağlantılıdır. Bunu ortaya çıkarmak için ise, LSA'dan faydalanılır. Soldaki dokümanlar matris dönüşümünde transpoze edilip SVD uygulandığından bağlam bütününde en az iki kez karşılaşılan doküman içi terimler tabloya yansır ve bu sebeple doküman kelimelerinin tamamı kırmızı ile işaretlenmemiştir. Sağdaki tablo ise bu durumun özetidir. Burada LSA, mevcut elemanları için, $\mathbf{A} = \boldsymbol{U\Sigma V^T}$ şeklinde formüle edilen ve SVD olarak adlandırılan matematiksel bir hesaplama

[8] Şekil 4'teki örneğin sol kısmında, *human* kelimesinden *user* kelimesine ulaşmada, gösterim kolaylığı açısından yalnızca c1 ve c2 dokümanları baz alınarak *computer* kelimesi örneklendirilmiştir. Ancak *human* ve *user* kelimeleri aynı zamanda c1 ve c3 bağlamları ele alındığında da benzer bir şekilde fakat bu kez *interface* kelimesi üzerinden gizli anlamsal ilişki içerisindedir. Bu durum, aşağıda da ele alındığı gibi X, Y ve Z ' nin her birinin uzaydaki diğer tüm varlıklarla ilişkisine bağlı olmasından kaynaklanır ve Y bu durumda ele alınan bağlam ikilisine göre değişilik gösteren kelime olarak düşünülebilir.

kullanmaktadır. Pratikte ne tür bir yol izlendiği hususunda ise, Landuaer ve Dumais (1997: 217)'de, örneğin bir kelime olan X, ortak bağlamda sıklıkla bulunarak başka bir kelime olan Y ile ilişkilendirilmişse ve Y de Z ile ilişkilendirilmişse, o zaman X ve Z ' nin arasında da benzer bir ilişkinin bulunabileceğinden bahsedilir. Ayrıca burada, X-Z ilişkisinin gücünün, X-Y ve Y-Z arasındaki ilişkilerin güçlerinin bir kombinasyonundan çok daha fazlasına bağlı olduğundan, bunun nedeninin ise, X, Y ve Z ' nin her birinin uzaydaki diğer tüm varlıklarla ilişkisine bağlı olduğundan bahsedilir ve bu durum "X, Y'den iki birim ve Y, Z'den iki birim mesafede ise, X'ten Z'ye olan uzaklık hakkında bilinebilen tek şeyin bunun sıfır ile dört arasında olması gerektiğidir" şeklindeki bir ifadeyle somutlaştırır (Landuaer ve Dumais, 1997: 217).

2008 yılında patent aitliği sona erdiği bilinen[9] LSA, sözdizimsel ve dilbilgisel açıdan işlenmeye uygun hale getirilen her türlü dokümana uygulanabilmekte ve yukarıda da görüldüğü üzere, özellikle metinler arasındaki gizli bağlantıları ortaya çıkarmada ve metinleri sınıflandırmada, veyahut örneğin "sonbahar" kelimesi ile arama yaptığımızda "güz" kelimesinin de bulunduğu bağlamlara ulaşabilme gibi eş anlamlı sözcüklerin kavramsal aramalarında ve daha pek çok farklı alanlarda kullanılabilmektedir.

2.3.3. LDA (Topic Models)

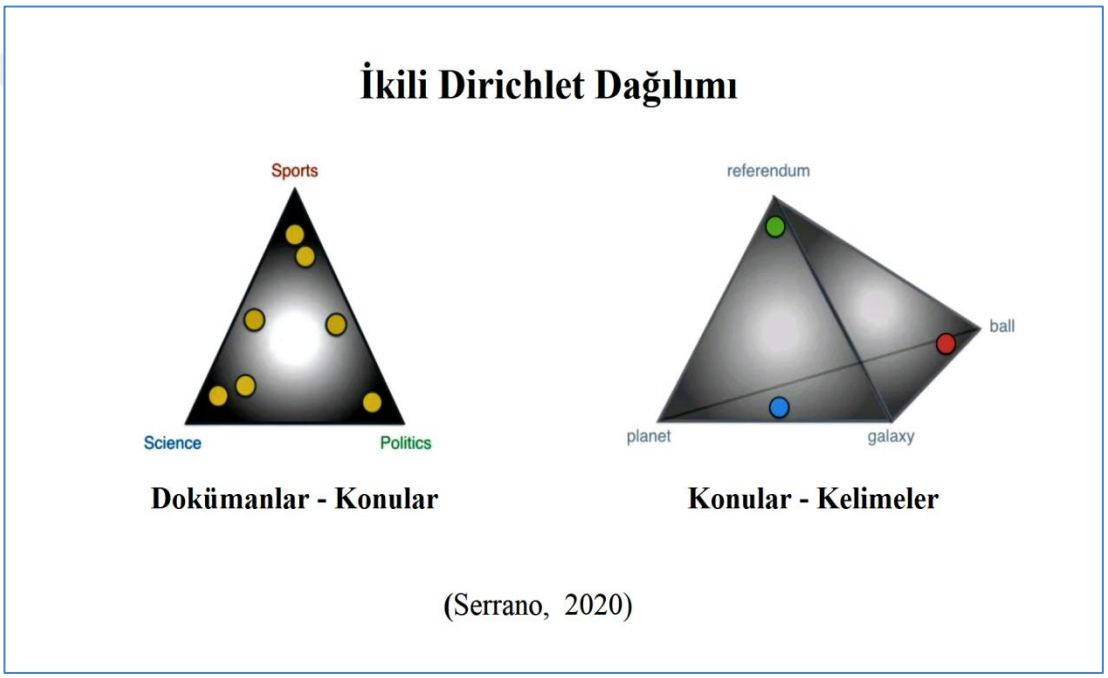
Gizli Dirichlet Ayırımı (LDA – Latent Dirichlet Allocation), metin verilerinden otomatik olarak konu çıkarımı yapabilen ve belirli bir koleksiyondaki gizli temaları keşfederek belgeleri bu temalara göre kümelemek ya da belgeleri özetleyerek sınıflandırmak gibi oldukça geniş bir kullanım alanı bulunan bir modeldir. LDA kısaca, LSA ve pLSA (Probabilistic Latent Semantic Analysis) modelleri yeterli performansı ve istenilen sonuçları verme konusunda yetersiz kaldığı için[10] geliştirilmiş bir üretici olasılık modeli olarak özetlenebilir. Bu modelin temeli, bir koleksiyonun her bir öğesinin, altta yatan bir dizi konu üzerinde sonlu bir karışım şeklinde modellendiği üç düzeyli hiyerarşik bir Bayes modeline dayanır (Blei ve

[9] https://www.ihs.com.tr/blog/lsi-nedir-ve-lsi-anahtar-kelimeler-nasil-kullanilmalidir/

[10] Blei ve diğerleri (2003).

diğerleri, 2003: 993). Diğer modellerde bir doküman bir konu ile temsil edilirken, bu modelde üretilen plakalar sayesinde içerdiği kelimelere göre değişebilen birden fazla konu temsili mümkündür. Rakamsal olarak hangi temsilin yüzdelik oranı büyükse ana konu olarak o gruba dahil edilebilir ya da ikinci bir yan konu gösterimine gidilebilir. Bunun sebebi ise, LDA'nın temelinde, konuların kelimeler üzerinde ve dokümanların da konular üzerinde dağılımı mevcut olduğundan ikili bir dağılımsal olasılık hesabı söz konusu olduğundan kaynaklanır.

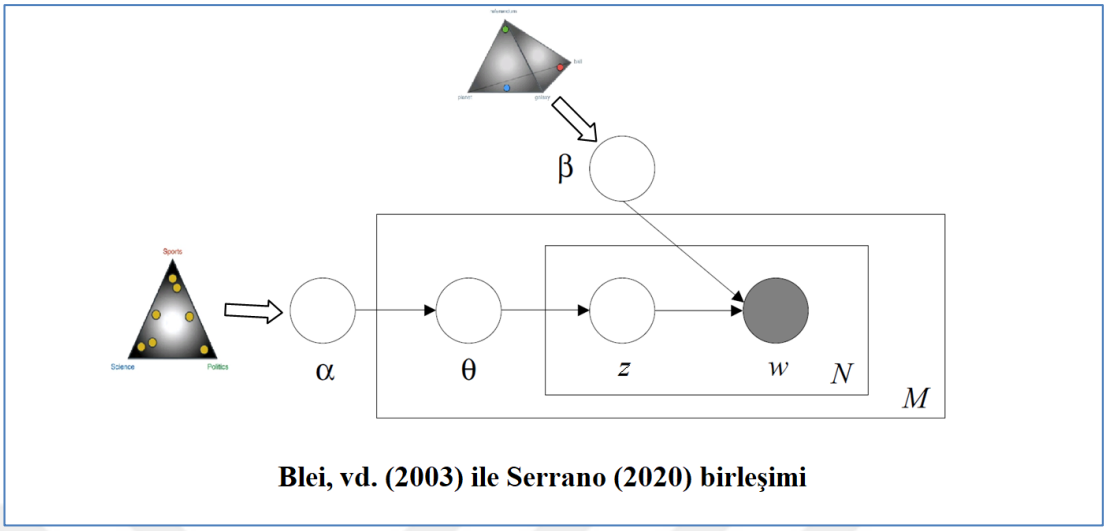
Şekil 5: İkili Dirichlet Dağılımı



Kaynak: Serrano (2020) (Yazar tarafından Türkçeleştirilmiştir).

Bunlar kısaca, dokümanlar içerisinde hale hazırda bulunan kelimeler ve bu kelimelerin olasılık hesaplamalarına dayanarak elde edilen konular olarak düşünülebilir. Teknik açıdan ise, her bir konuyu temsil eden en iyi kelimeler tespit edilirken bir yandan da her bir dokümanı en iyi temsil eden konular belirlenerek bir öğrenme gerçekleştirilir(Aydın ve Hallaç, 2021: 600).

Şekil 6: LDA Modeli İşleyişi



Kaynak: Blei ve diğerleri (2003) ile Serrano (2020) yazar tarafından birleştirilmiştir.

LDA'da $P(\ kelime \mid konu)$ ve $P(\ konu \mid doküman)$ olasılıkları üzerinden bir optimizasyon gerçekleştirilmesi amaçlanır ve bu yöntemde doküman kümelemeye benzer olarak, önceden belirlenen K sayısı kadar konu dokümanlarla rastgele olarak eşleştirilerek konu modellemesi yapılır (Aydın ve Hallaç, 2021: 600). Devamında ise bu ikisinin ne derece örtüştüğünün hesaplaması yapılarak gerekli güncellemeler yapıldıktan sonra sınıflandırılır. Aşağıdaki Tablo 7'de, Şekil 6'daki terimlerin yanı sıra, güncelleme denklemi de görülmektedir.

Tablo 7: LDA Terim ve Parametre Açıklamaları

Güncelleme Denklemi :

$$\frac{n_{ik}+a}{N_i-1+Ka}$$

Terim	Açıklama
K	Konu sayısı
n_{ik}	i. dokümanda bulunan k. konuya atanmış kelime sayısı
N_i	Dokümanda bulunan toplam kelime sayısı
a	Hangi konunun hangi dokümana atandığını gösteren parametre
β	Her bir konu için kelime atamalarını gösteren Dirichlet parametresi
M	Doküman sayısı
N	Her bir dokümandaki kelime sayısı
θ	θ dokümanı için konu dağılımı
z	Her bir kelime için atanan konular
w	Anlık incelenen kelime

Kaynak: Aydın ve Hallaç (2021: 600 – 601) yazar tarafından modifiye edilmiştir.

Diğer bir yandan, LDA'da belgeleri işlemeye başlamadan önce bir takım ön işlemlerden geçirildiği de görülür. Her belge sadece bir kelime koleksiyonu ya da bir "kelime çantası (bag of words)" olarak ele alınır [11] ve bu nedenle de kelimelerin sırası ya da dilbilgisel nitelikleri modelde dikkate alınmaz, ayrıca "konular" hakkında nitelikli bir bilgi taşımayan ve ayrıca çok sıklıkla tekrarlanan bazı kelimeler ön işleme esnasında genellikle belgelerden elenerek çıkarılır. Ancak bu durum belgedeki bilgilerin kaybına neden olmayıp, tam tersine hatalı sonuçlar vermesinin önüne geçilmesini sağlar.

2.3.4. BEAGLE

Yukarıda anlatılan modellerden özellikle LSA ve HAL, pek çok farklı kullanım alanı bularak günümüze değin farklı uygulama ve projelerde çeşitli başarılara imza atmışlardır ancak diğer bir yandan, ilk çıktıklarında özellikle Perfetti (1998: 376)'de görüldüğü gibi ya tam anlaşılamamalarından ya da Serafin ve Eugenio (2004: 692)'te de bahsedildiği üzere bir takım bilgileri bünyelerinde barındıramamalarından dolayı yoğun eleştirilere de maruz kalmışlardır. Bunun en büyük sebebi ise, örneğin HAL'de belirli bir pencere genişliğinden kelimelerin birbirleri ile olan mesafelerine dayalı bir bağlantı ölçümü gerçekleştirilir ancak kelime sıralaması ve dolayısıyla da dilbilgisel özellikleri dikkate alınmaz. Yine LSA için de durum farklı değildir, kelime çantası modelinden faydalanıldığından kelimelerin sırası ya da dilbilgisel nitelikleri modelin dışında bırakılır. Yapılan eleştirilere bakıldığında ise özellikle bu iki durumun eleştiri konusu olarak ön plana çıktığı görülmektedir (Perfetti, 1998; Serafin ve Eugenio, 2004; Jones ve Mewhort, 2007). Bu durum Jones ve Mewhort (2007: 2)'de de dile getirilmiş ve bu sebeple yeni bir model olan BEAGLE modeli önerilmiştir. Kelime bazlı n-gram (aslen n-gramlar) yönteminden faydalanan BEAGLE modelinde öğrenilen temsillerin, LSA ve HAL tarafından öğrenilenlerden en büyük farkı, BEAGLE'da bağlam bilgilerine ek olarak sıra bilgilerini de dikkate alabilen bir sistem geliştirilmiş olmasıdır (Jones ve Mewhort, 2007: 2, 30). N-gram yöntemi kelime ya da harf bazlı olabilmektedir (Topuz, 2017). Aşağıdaki şekilde kelime bazlı olan n-gramların işleyişi gösterilmiştir.

[11] https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2

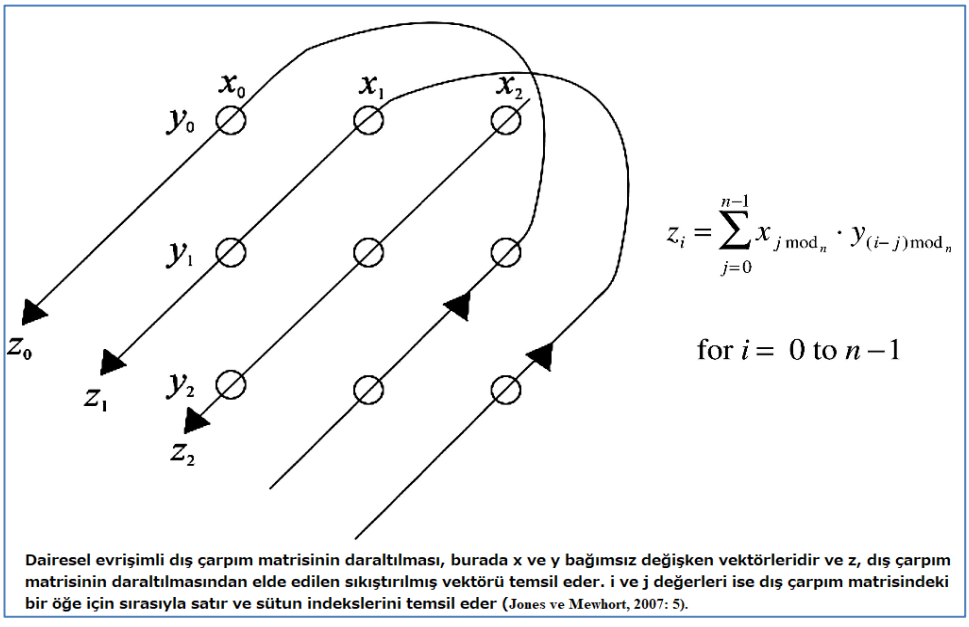
Şekil 7: Kelime Bazlı N-gram Yöntemi



Kaynak: Topuz (2017).

Önerilen BEAGLE modelinde hem kelime birlikteliklerini hem de sözdizimsel kelime sırası ile kelimelerin dilbilgisel özelliklerini tespit edebilmek için, görüntü ve sinyal işlemede yaygın olarak kullanılan bir teknik olan dairesel konvolüsyon modelinden faydalanılmış ve böylece özyinelemeli evrişim ile genişleyen boyutluluk problemine de çözüm üretilebilmiştir (Jones ve Mewhort, 2007: 4-5). Aşağıda "Toplu Dil Ortamının Bağlı Kodlaması (Bound Encoding of the Aggregate Language Environment – BEAGLE)" anlamına gelen BEAGLE modelinin dairesel evrişimi temsili olarak görülmektedir.

Şekil 8: Dairesel Evrişim



Dairesel evrişimli dış çarpım matrisinin daraltılması, burada x ve y bağımsız değişken vektörleridir ve z, dış çarpım matrisinin daraltılmasından elde edilen sıkıştırılmış vektörü temsil eder. i ve j değerleri ise dış çarpım matrisindeki bir öğe için sırasıyla satır ve sütun indekslerini temsil eder (Jones ve Mewhort, 2007: 5).

Kaynak: Jones ve Mewhort (2007: 5).

Aslında oldukça karmaşık bir çalışma şekline sahip olsa da, temel olarak BEAGLE modelinin, kelime uzayında aynı bağlamda geçen öğelerin bağlam ve sıra bilgilerinin vektörel olarak bir birleri üzerine birleştirilip, tek bir vektör ile temsil edilmesi prensibine dayandığı söylenebilir. Murdock (1982)'den esinlenerek ürettikleri bu model sayesinde, cümledeki kelimelerin ilişkisel konumlarını temsil etmek için evrişim bir sıra bağlama mekanizması olarak kullanılır ve bu sayede sıra bilgisi, kelimelerin cümlede birlikte bulundukları diğer kelimelere göre olan sözcüksel temsil konumlarından ya da hangi kelimelerle daha sıklıkla rezonansa girdikleri gibi bir kelime geçmişi bilgisinden yola çıkılarak elde edilir (Jones ve Mewhort, 2007: 12).

2.3.5. WORD2VEC

Yapılan çalışmalarda kullanılan teknikler sebebiyle özellikle bir takım sınırlılıkların var olduğu fark edilmiş ve bu durum Mikolov ve arkadaşları tarafından 2013'te yapmış oldukları çalışmada dile getirilmiştir. Bu sınırlılıklar ise, 2013 öncesinde var olan tekniklerin eğitilebilen kelime sayısını genellikle birkaç milyon kelimeden ve maksimum 50-100 vektör boyutundan daha öteye götüremediği şeklinde dile getirilmiştir (Mikolov ve diğerleri, 2013a: 1-2). Bu sebeple, yapmış oldukları çalışmada hem milyarlarca kelime vektörünü oldukça kısa bir sürede işleyebilen hem de vektörlerin 300 ile 1000 boyutlu olarak işlenebilmesine olanak sağlayan 2 tip yeni model tanıtmışlardır. Sürekli Kelime Çantası (CBOW) ile Sürekli Atlamalı Gram (Skip-gram) modelleri olarak bilinen bu modeller sayesinde 6 milyar kelimeyi 2-2,5 gün gibi kısa bir sürede işleyebilir hale gelmişlerdir. Aşağıda bu iki modelin diğer tekniklerle donanımsal ve zamansal açıdan kıyaslaması verilmiştir.

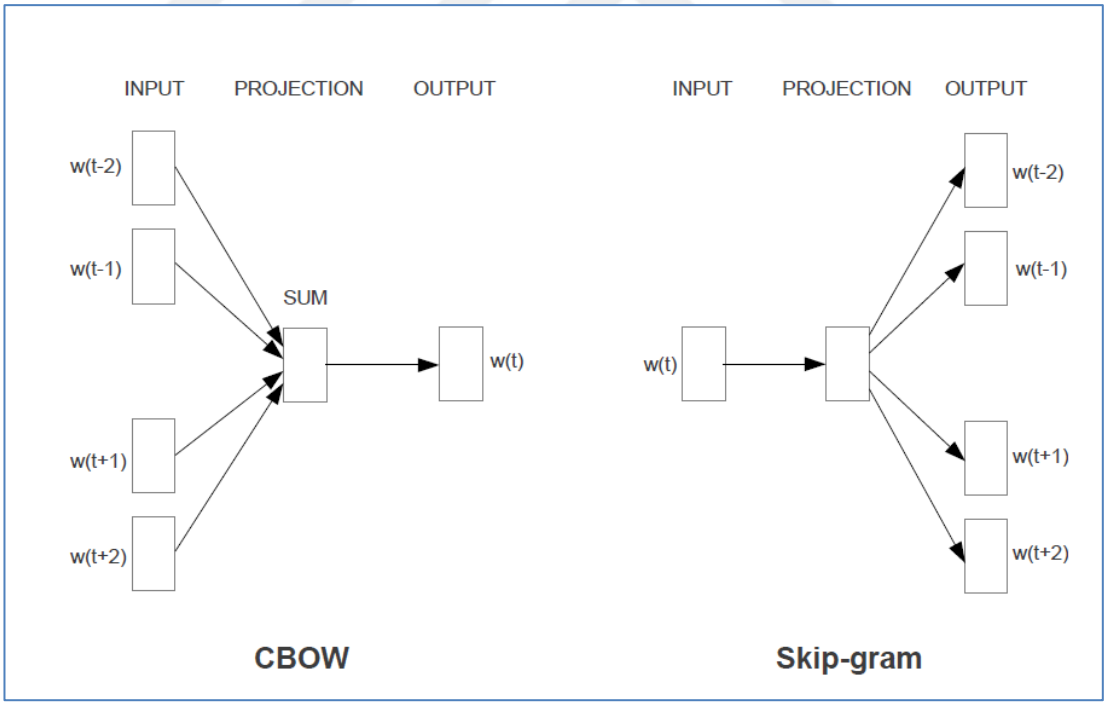
Tablo 8: Word2vec Modellerinin Karşılaştırması

Model	Vektör Boyutu	Eğitilen Kelime Sayısı (B = Milyar)	Doğruluk Oranı [%]			Eğitim Süresi [gün x CPU cores]
			Anlamsal	Sentaktik	Toplam	
NNLM	100	6B	34.2	64.5	50.8	14 x 180
CBOW	1000	6B	57.3	68.9	63.7	2 x 140
Skip-gram	1000	6B	66.1	65.1	65.6	2.5 x 125

Kaynak: Mikolov ve diğerleri (2013a: 9) (Yazar tarafından Türkçeleştirilmiştir).

Burada görüldüğü üzere CBOW ve Skip-gram modellerinin her ikisinin anlamsal ve sentaktik(sözdizimsel) başarısı, İleri Beslemeli Sinir Ağı (NNLM)'na göre hem doğruluk oranları açısından daha yüksek hem de eğitim süresi ve donanımsal gereksinimler açısından adeta çığır açar niteliktedir. Aynı zamanda bu iki model, bir birinin simetrisi olma özelliği de taşımaktadır. CBOW modelinde bir bağlamda örneğin w(t) adını verdiğimiz kelimenin bu kelimeden önce gelen w(t-2), w(t-1) ile sonra gelen w(t+1) w(t+2) kelimelerine bağlı olarak tahmin edilmesi söz konusu iken; Skip-gram modelinde ise bunun tam tersi olarak ile verilen bir w(t) kelimesinden yola çıkılarak aynı bağlamda bulunan w(t-2), w(t-1) ve w(t+1), w(t+2) kelimelerinin tahmini söz konusu olmaktadır. Dolayısıyla CBOW modelinde output kelimesi bağlamda önce ve sonra gelen input kelimelerine göre tahmin edilirken, Skip-gram modelinde input olarak verilen kelimeden bağlamdaki diğer kelimelerin birliktelikleri output olarak bulunmaktadır.

Şekil 9: CBOW ve Skip-gram Modelleri

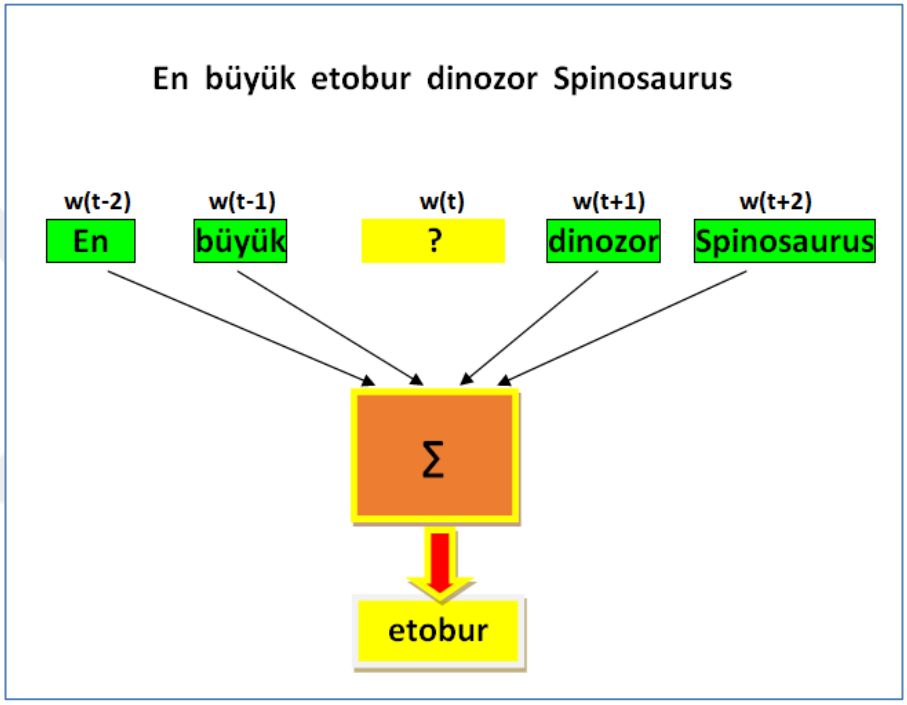


Kaynak: Mikolov ve diğerleri (2013a: 5).

Yukarıda da görüldüğü üzere, CBOW modelinde w(t) kelimesinden önce gelen 2 kelime ile w(t) kelimesinden sonra gelen 2 kelime input(girdi) olarak modele

verilir ve bağlamda bulunan bu 4 kelime vasıtasıyla istatistiksel olarak sıklıkla bir arada bulunma eğiliminde olan orta kelime w(t) nin ne olabileceği tahminlenir. CBOW [12] modelinde, input adı verilen bir girdi katmanı ile bu girdileri toparlayarak vektör üreten bir projeksiyon katmanı ve ayrıca output adı verilen ve input girdilerine göre orta kelime olan w(t)'yi tahmin eden bir çıktı katmanı bulunmaktadır. Aşağıda CBOW'un işleyişini temsili olarak gösteren için basit bir örnek verilmiştir.

Şekil 10: CBOW Modeli İşleyişi

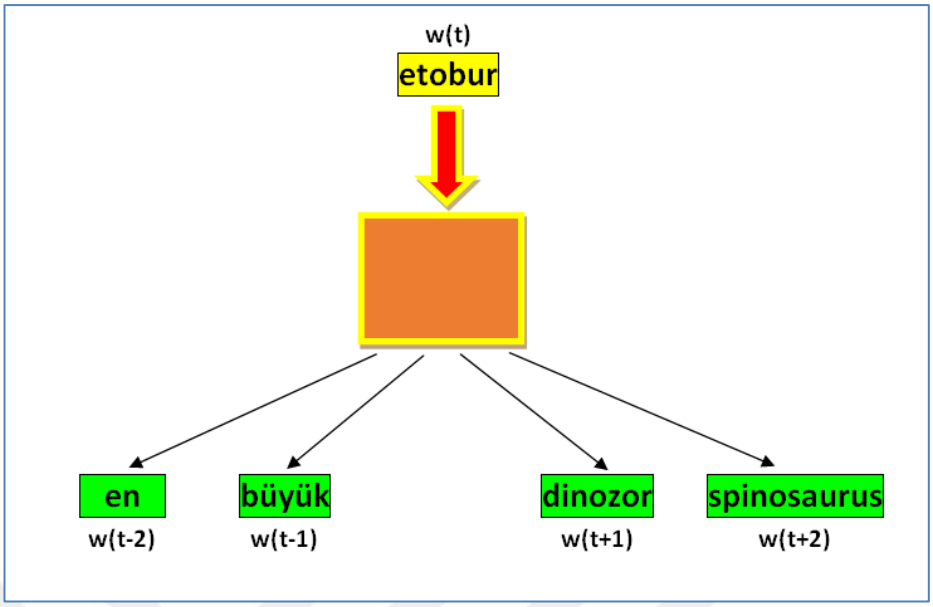


Kaynak: Yazar tarafından derlenmiştir.

Skip-gram modelinde ise, yukarıdaki CBOW'un tam tersi yönde bir işlem gerçekleşmektedir. Skip-gram'da input olarak verilen w(t) kelimesinden yola çıkılarak w(t-2), w(t-1), w(t+1), w(t+2) kelimeleri, diğer bir deyişle bağlamda birlikte bulunma olasılıkları yüksek olan bağlantılı kelimelerin output yani çıktı olarak elde edilmesi hedeflenmektedir. Bu durumu yukarıda verdiğimiz örneği kullanarak açıklayacak olursak aşağıdaki gibi olur.

[12] Mikolov ve diğerleri (2013a)'te CBOW mimarisi karmaşıklık hesabı için "Q = N x D + D x log2(V)" formülü verilmekte olup, burada Q model mimarisi karmaşıklığını, N kodlanmış kelime girişlerini ve D Projeksiyon matrisinin boyut sayısını ve V ise sözcük dağarcığı miktarını temsil etmektedir.

Şekil 11: Skip-gram Modeli İşleyişi



Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü gibi Skip-gram modelinde bu kez input olarak verilen "etobur" kelimesinden yola çıkarak bağlamda bu kelimeye yakın olan diğer ilişkili kelimeler elde edilebilmektedir. Skip-gram modelinde başlangıçta Softmax algoritmasından faydalanma denenmiş ancak bunun tüm derleme uygulanmasının hesaplama maliyeti çok yüksek olduğundan farklı çözüm yolları aranmıştır. Bu amaçla ilk etapta *Hiyerarşik Softmax* [13] adı verilen ve Softmax'a ikili ağaç modelinin adapte edilmiş bir şekli olan algoritma kullanılmış ve bu sayede bir takım kayıplar yaşansa da hesaplama sürelerinin oldukça kısaldığı tespit edilmiştir. Ancak mevcut kayıpların doğruluk oranlarını olumsuz yönde etkilemesi durumu söz konusu olduğundan farklı bir çözüm yolu da yine aranmaya devam edilmiştir. Bunun sonucunda ise, Gutmann ve Hyvärinen (2012) tarafından aslında resim işlemede kullanılmak üzere tasarlanan Gürültü Kontrast Tahmini (Noise Contrastive Estimation – NCE) modelinin, Mnih ve Teh (2012) tarafından dil modellemeye uygun hale getirilerek uyarlanması neticesinde, *Negatif Örnekleme (Negatif Sampling - NEG)* algoritması elde edilmiş ve Hiyerarşik Softmax'a alternatif bir olarak kullanılmaya başlanmıştır (Mikolov ve diğerleri, 2013b: 4).

[13] Bu algoritma Morin ve Bengio (2005) tarafından dil modellemelerindeki öğrenme süresini kısaltma amacıyla önerilen modelin Mnih ve Hinton (2008)' e göre modifikasyonu neticesinde elde edilmiştir (Sübay, 2019: 31-32; Mikolov ve diğerleri, 2013b: 4).

Negatif Örnekleme (Negatif Sampling - NEG) algoritması, temel olarak w(t)'ye göre bağlamda yakın mesafede bulunan kelimelerin (pozitif örneklem kümesi) değerlerini, ikili kıyaslama metodu kullanarak, bağlamdan tamamen uzak mesafedeki kelimeler (negatif örneklem kümesi) vasıtasıyla örnekler ve böylece pozitif örnekler içeren küme elemanlarının olasılığını lojistik regresyon ile eğitir [14]. Burada negatif örnekler belirli bir sayıda, diğer bir deyişle "k" adet seçilebilir. Bu sayede tüm derlemi kapsayacak şekilde büyük bir dağılım ve hesaplama işlemi yapılmayıp sadece k kadar (genellikle 5 - 20 kelimelik [15]) büyüklüğü olan bir ön tanımlı küme üzerindeki dağılım sonuçlarına göre işlem yapılır. Ayrıca burada, belli bir frekansın üzerindeki kelimeler ile belirli bir frekansın altında kalan ancak önemli bilgiler içeren kelime birlikteliklerine sahip olan kelimeler arasında bir denge kurulabilmesi amacıyla, belirli bir eşik değeri verilerek bir alt kümeleme (subsampling of frequent words) yapıldığı görülmektedir (Mikolov ve diğerleri, 2013b: 4-5) . Böylece hem zamansal ve donanımsal gereksiz maliyetlerin önüne geçilebilmekte hem de sıklık frekansı çok yüksek olan kelimelerin diğer kelimelerden elde edilebilecek bilgileri engellemesi önlenebilmektedir.

Bu model bir derleme uygulandığında ise, denetimsiz öğrenme meydana gelerek aynı tür ve ortak özelliklere sahip olan kelimeler gruplanabilmektedir. Örneğin "Salonu ****** renge boyadı." şeklinde bir cümlemiz olsun burada yıldızla işaretli boşluğa gelebilecek kadar örneğin "Salonu **kırmızı** renge boyadı.", "Salonu **mavi** renge boyadı.", "Salonu **pembe** renge boyadı.", "Salonu **mor** renge boyadı.", "Salonu **yeşil** renge boyadı.", "Salonu **sarı** renge boyadı." şeklinde çeşitli renklere

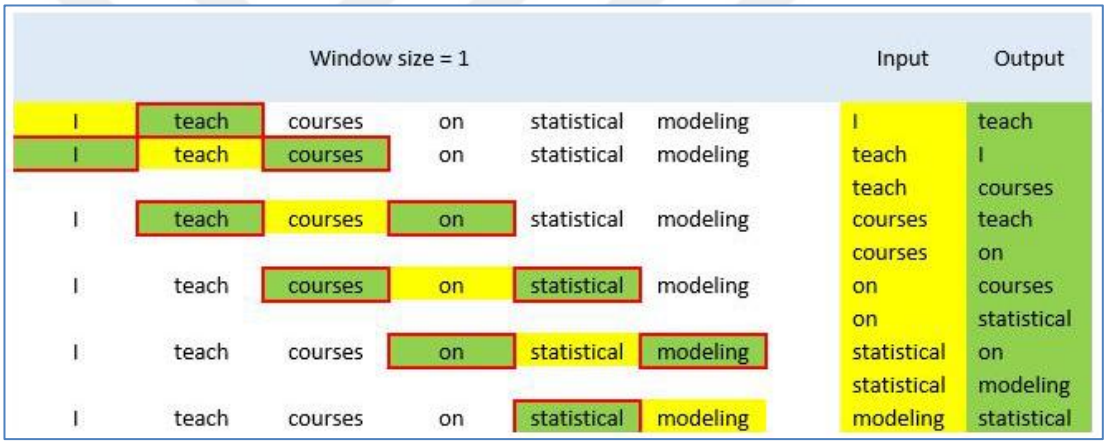
[14] Aslen bir sonraki kelimeyi tahmin etmek, çok sınıflı sınıflandırıcı ile bir sınıf tahmininde bulunmak gibidir ve çıktı katmanında sınıf sayısı kadar nöronun bulunmasını gerektirmektedir. Dolayısıyla, her bir sınıf gerçek kelimelerle temsil edildiğinden, derlemin bütünü düşünüldüğünde bu durum çok fazla sayıda nöronun çıktı katmanında bulunması demektir. Temel sinir ağlarında olasılık dağılımının temsil edilebilmesi için çıkıştaki nöron değerleri genellikle *çapraz entropi (cross entropy)* fonksiyonu ile işleme tabi tutulduğundan, ağ tarafından her bir sınıf için hesaplanan çıkış değerlerinin normalleştirilmesi ve dönüştürülmesi gerekmektedir ve bu normalleştirme adımında ise, Softmax'tan faydalanılmaktadır (StackExchange, 2016). Bu sebeple, çıktı katmanında ne kadar çok kelime olursa o kadar fazla nöron olacağından, Softmax normalleştirme için uygulandığında çok büyük bir hesaplama maliyeti söz konusu olacaktır. İşte bu noktada Negatif Örnekleme devreye girmektedir. Negatif Örnekleme'deki temel fikir ise, sonraki kelimeyi tahmin etme vasıtasıyla oluşan çok terimli bir sınıflandırma probleminde Softmax kullanmak yerine, bu problemi ikili bir sınıflandırma problemi haline dönüştürerek, çıktı kelimesinin gerçek olasılık dağılımını tahmin etmede daha az maliyetli olan *ikili lojistik regresyon* sınıflandırmasını kullanmaktır (StackExchange, 2016).

[15] Mikolov ve diğerleri (2013b: 4)'te, "k" sayısının genellikle 5-20 olarak belirlenmesi tavsiye edilmekte olup, ancak büyük derlemlerde ise muhtemelen hız ve hesaplama maliyetlerini düşürme amacıyla 2-5 olarak daha düşük sayıda da belirlenebileceğinden bahsedilmektedir.

boyama yapıldığına ilişkin cümlelerin de bağlamda bulunduğunu var sayalım. Bu durumda, renklere ilişkin kelimeler olan w(t)'ler bağlamdaki diğer kelimeler olan w(t-1), w(t+1) ve w(t+2) kelimelerine benzer bir mesafeye sahip olacağından aynı küme altına toplanarak etiketlenecektir. Neticesinde ise, renk kavramlarına ilişkin bu kelimelerden her hangi biri verildiğinde kümedeki diğer benzer renk kelimelerine ulaşmak word2vec işlemesi yapılmış bir modelde mümkün hale gelecektir.

Word2vec'te pencere genişliği (window size)[16] denilen bir parametre daha bulunmaktadır. Yukarıdaki örneklerin hepsinde bu pencere genişliği 2 olarak verilmiştir ancak bu rakam daha yüksek ya da daha düşük olarak da verilebilir. Aşağıda pencere genişliği 1 olduğu duruma bir örnek gösterilmiştir.

Şekil 12: Pencere Genişliği



Kaynak: Kilmen ve Bulut (2022).

Skip-gram ya da CBOW İşleyiş modellerinde olduğu gibi pencere genişliği 2 ya da daha yüksek de verilebilir, ancak bu değer yükseldikçe örneğin pencere genişliğinin "n" ile temsil edildiğini varsayalım, pencerenin her sağa hareketinde w(t-n)…..w(t-1), w(t) ve w(t+1)…..w(t+n) şeklinde daha uzun hesaplama maliyetlerinde neden olacaktır. Düşük verilmesi durumunda ise her ne kadar bu maliyetler azalmış olsa da, merkezdeki kelimenin daha az bağlam öğesi ile temas ederek hesaplanmasına sebep verecektir. Dolayısıyla bu durum, kelime birliktelikleri

[16] "Window size" kelimesi Türkçe kaynaklarda çoğunlukla "Pencere Boyutu" olarak geçmektedir ancak pencere hücre sayılarının sadece sağa ve sola doğru, yatay düzlemde genişlemesi söz konusu olduğundan ve ayrıca "boyut" kelimesinin İngilizcedeki karşılığı olan "dimension" kelimesi ile karışmaması adına da, "window size" kelimesinin karşılığı olarak "Pencere Genişliği" tabirinin kullanılması tercih edilmiştir.

hakkında elde edilecek bilgi yoğunluğu ile odak durumundaki kelime hususunda elde edilecek bilgi değişikliği faktörü olarak karşımıza çıkacaktır. Levy ve Goldberg (2014: 304), büyük bir konu içeriğini yakalamak için genellikle pencere genişliği 5 olarak belirlenirken, daha küçük pencere değerlerinin kullanmanın hedef kelime hakkında daha fazla odaklanmış bilgi içereceğinden bahseder. Yine benzer bir tespit Bansal ve diğerleri (2014: 811)'te de görülerek, pencere genişliğinin 1 olması durumunda merkezdeki kelimeye odaklanılıp daha çok sözdizimsel (sentaktik) özellikler ve dolayısıyla genelde aynı türdeki kelime ve bilgiler elde edilmekte iken, pencere genişliği 10 verildiğinde ise belirli bir konu hususunda merkez kelimesine bağlı olarak daha uzak mesafede de olsa o konu içerisinde genellikle geçmesi olası kelimeler ve buna bağlı bilgiler elde edilmektedir. Aşağıda, pencere genişliği ve elde edilen özellikler hususunda Bansal ve diğerleri (2014)'te verilen tablo görülmektedir.

Tablo 9: Pencere Genişliği Değeri

w	Example clusters
1	[Mr., Mrs., Ms., Prof., ...], [Jeffrey, Dan, Robert, Peter, ...], [Johnson, Collins, Schmidt, Freedman, ...], [Portugal, Iran, Cuba, Ecuador, ...], [CST, 4:30, 9-10:30, CDT, ...], [his, your, her, its, ...], [truly, wildly, politically, financially, ...]
10	[takeoff, altitude, airport, carry-on, airplane, flown, landings, ...], [health-insurance, clinic, physician, doctor, medical, health-care, ...], [financing, equity, investors, firms, stock, fund, market, ...]

Pencere genişliği (w) = 1 (sözdizimsel) ile w = 10 (topik) olan SKIP yerleştirmeleri için örnek kümeler.

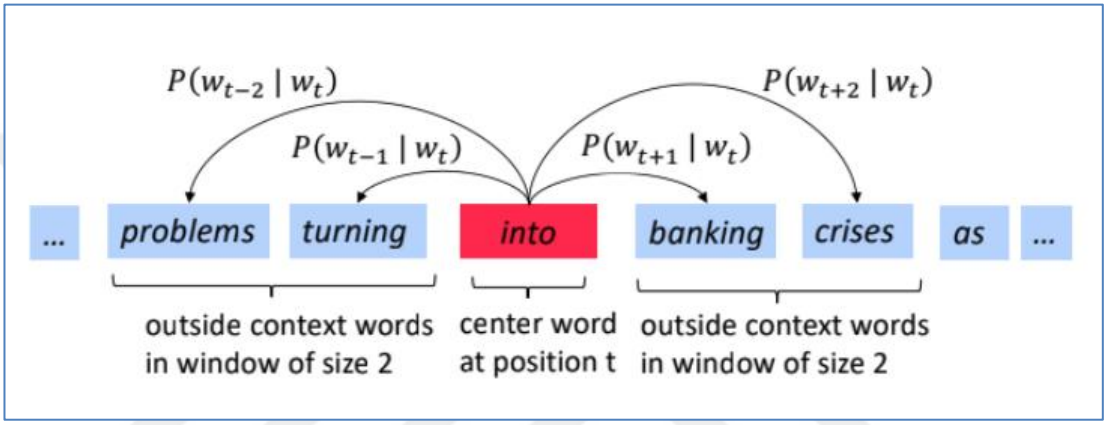
Kaynak: Bansal ve diğerleri (2014: 811).

Mikolov ve diğerleri (2013b: 7)'de 33 milyar kelimelik bir derlem üzerinde tüm cümle kullanılarak bir eğitim yapıldığında %72 doğruluk oranına sahip bir model elde edildiğinden, ancak derlemi 6 milyar kelime gibi daha düşük olarak belirlediklerinde bunun doğruluk oranına olumsuz yönde etkilerinin olduğundan bahsedilmektedir. Dolayısıyla derlem büyüklüğünün, nitelikli sonuçlar almada direk etkisi olduğu söylenebilir ancak burada özellikle dengeyi yakalamak da önemlidir.

Eğer sadece teknik alana yönelik dokümanlardan oluşan bir derlem kullanılıyorsa bu durumda günlük konuşma ya da yaşantıda karşılaşılan (örneğin argo, ya da z-kuşağı konuşma dili) kelimelere ilişkin yüksek bir doğruluk oranı beklemek yersiz olacaktır.

Diğer bir yandan Word2vec modelinde hesaplama yapılırken burada pencerenin her hareketinde merkezdeki kelimeye göre tüm olasılık dağılımının yeniden hesaplanması gerekmektedir.

Şekil 13: Word2vec Dağılımı



Kaynak: Manning, C. (2019).

Burada tüm bağlam boyunca pencerenin her hareketinde $P(w_{t\pm n} \mid w_t)$ değerleri merkez konumunda olan kelimeye göre her seferinde yeniden hesaplanmaktadır. Bu durumda ise derlem ne kadar büyükse belirli oranda hesaplama maliyeti olarak sonuca yansıyacaktır.

2.3.6. GLOVE

LSA ve HAL gibi modeller hem kelime dağılımları hususunda istatistiksel verimliliklere hem de hızlı eğitim sürelerine sahip iken, özellikle Word2vec'in bu konuda bazı açılardan yetersiz kaldığı görüşüne sahip Standford Üniversitsesi'nden Pennington, J., Socher, R. ve Manning C. ve yapmış oldukları çalışma neticesinde Word2vec'e belirli açılardan benzer ancak özellikle dağılım hesaplaması konusunda farklı bir teknik kullanan Global Vektör (GloVe) adını verdikleri yeni bir modeli 2014 yılında tanıtmışlardır (Pennington ve diğerleri, 2014: 1532, 1541). Teknik

açıdan, matris çarpanlarına ayrım metodu kullanan GloVe, sığ katmanlı ve ileri beslemeli bir sinir ağına göre eğitim yapan[17] Word2vec'e göre rakip niteliğindedir. Word2vec modelinde kelime birliktelikleri üzerindeki dağılım hesabının pencerenin her hareketinde tekrar hesaplanmasını eleştiren Socher ve Manning (2017), bu durumun GloVe tekniği ile çözülerek, GloVe'da tüm derlem üzerindeki kelime birlikteliklerinin sayılarak istatistik bilgilerinin sonuca yansıtılabildiğinden bahsetmektedir. Diğer bir yandan, Word2vec'te pencerenin her hareketinde kelime dağılım hesaplamasının yapılması eğitim süresi açısından belirli oranda bir dezavantaj olarak modele yansımaktadır ancak diğer bir yandan derlemdeki tüm kelime dağılımları yerine sadece daha dar kapsamlı olan bağlamda bulunan kelime dağılımlarına göre hesap yapılmasının da özellikle teknik alan gibi dar kapsamlı bir alan söz konusu olduğunda tam tersine bir avantaj olarak görülebileceği de düşünülmektedir.

Aslen sayma tabanlı olan ve aynı zamanda doğrudan tahminleme yöntemi de kullanan ve ayrıca log-bilineer bir model olan GloVe, kelimelerin birbiriyle ne sıklıkta göründüğünü hesaplar ve yalnızca yerel bağlam pencereleri olan ve bu sebeple kelimelerden gelen yerel bilgilere dayanan Word2vec'in aksine, kelime birliktelik oluşum bilgilerini genel istatistik bilgileri ile birleştirerek derlemdeki kelimeler arası anlamsal ilişkileri elde eder (Dharma ve diğerleri, 2022: 351). Bu özelliği itibariyle genellikle sayım tabanlı bir model olarak nitelendirilen GloVe'da yerel bir pencere mantığı yerine matris çarpanlarına ayırma yönteminden faydalanıldığından Word2vec'teki gibi bağlam üzerinde hareket ederek hesaplama yapan bir pencere bulunmaz ve bu sebeple de Word2vec'e göre daha hızlı sonuçlar üretir.

Ancak diğer bir yandan yukarıda 2.3.5.'te derinlemesine ele aldığımız Word2vec modeli öncelikli olmak üzere, 2.3.6.'da ele aldığımız GloVe modelinde de, vektörler üzerinde "king – man + woman = queen" şeklinde basit hesaplama işlemleri yapmamız mümkündür. Özellikle Word2vec üzerinde devam edilecek olan bu çalışmada, dilbilim'de "*Bileşen Analizi (Componentional Analysis)* olarak adlandırılan bu yöntemden sıklıkla faydalanılacaktır.

[17] MLNerds (2019)'e göre iki model arasındaki en büyük fark, Word2vec gömmeleri, sığ beslemeli ileri bir sinir ağının eğitimine dayanırken, GloVe gömmeleri matris faktörizasyon tekniklerine dayanarak öğrenmesidir.

2.4. BİLEŞEN ANALİZİ ve ANLAM ÇÖZÜMLEMESİ

Amerika'da, L. Bloomfield ile başlayan yapısalcı dilbilim, Bloomfield'in öğrenci olan Z. S. Harris sayesinde doruk noktasına ulaşmıştır. Özellikle Harris'in yapmış olduğu çalışmalar sayesinde insanoğlunun konuştuğu doğal dil, matematik ve istatistik alanına yakınlaştırılarak daha rahat uyarlanabilir hale gelmiştir. Elbette bu serüvende Firth (1957: 11)'in "*You shall know a word by the company it keeps!*" şeklinde özetlediği bir kelimenin anlamının ona eşlik eden kelimelerden elde edilebilir olduğu görüşü de büyük katkılar sağlamış ve günümüzdeki DDİ (NLP) uygulamalarının temelleri atılmıştır. Diğer bir yandan, Harris'in doktora öğrencisi olan N. Chomsky tarafından yapısalcı dilbilim çalışmaları bir adım daha öteye taşınmış ve devamında ise Chomsky'nin öğrencileri olan Katz ve Fodor'un da katkılarıyla, yapısalcı dilbilimde *Bileşen Analizi (Componentional Analysis)* adı verilen yeni bir yöntem uygulanmaya başlanmıştır. Bu yöntem, aslen sesbilim (phonology)'de kullanılan Ayırt Edici Nitelik (Distinctive Features) çıkarımının anlambilim alanına uyarlanması neticesinde elde edilmiş olup, bağlamda kullanılan kelimelerin anlamlarının matematiksel olarak toplama ve çıkarma yöntemi ile çözümlenmesini amaçlamıştır.

Yapısalcı düşüncede, kelime anlamı kendi başına diğer kelimelerden ayrı olarak var olamadığından, diğer bir deyişle kelimelerin anlamı sadece birbirleriyle ilişkili vaziyette genel bir sistemin parçası olarak var olabildiğinden, bir ifadenin anlamı diğer tüm kelimelerle olası ilişkilerinin toplamı olarak ele alınmakta ve bu nedenle de anlamsal analiz, ilgili kelimelerin içinde bulundukları anlam alanı içerisinde sistematik olarak anlamsal nitelikleri karşılaştırılarak zıt ve benzer yönleri en ekonomik şekilde özetlenmektedir (Goddard, 1998: 43). İşte Bileşen Analizi adı verilen yöntem de tam bu noktada devreye girmektedir.

Farklı bir açıdan bakıldığında, bir anlamda öznitelik çıkarımı olarak da görülebilen Bileşen Analizinde, anlamı olan en küçük birimler olan *anlambirimler*, onları oluşturan ve aslında öznitelik olarak da değerlendirilebilmesinin mümkün olduğu düşünülen *anlambirimcik* adı verilen parçalardan oluşmaktadır. İmer ve diğerleri (2011: 29)'de, "*belli bir anlam alanında işgören, anlamı ayırıcı en küçük anlamsal özellik* " olarak tanımlanan anlambirimcik için, "*örneğin masa anlambirimi*

'ayaklı', 'üzeri düz', kare, dikdörtgen', vb. anlambirimciklerden oluşur " şeklinde bir açıklama görülmektedir. Dolayısıyla anlambirim aslında bir kelimenin bağlamda mesaj olarak iletmekle yükümlü olduğu özniteliklerinin, diğer bir deyişle anlambirimciklerinin toplamından ibarettir denilebilir. Aşağıda bileşen analizi kullanılarak yapılan bir anlam çözümlemesi görülmektedir.

Tablo 10: Bileşen Analizi ile Anlam Çözümlemesi

	Erkek	Dişi	Yetişin	İnsan	Hayvan
Adam	+	-	+	+	-
Kadın	-	+	+	+	-
Oğul	+	-	-	+	-
Kız	-	+	-	+	-
At	±	∓	+	-	+
Tay	±	∓	-	-	+
Kısrak	-	+	+	-	+
Aygır	+	-	+	-	+

Kaynak: Yazar tarafından derlenmiştir.

Burada, mavi ile gösterilen **Adam**, **Kadın**, **Oğul**, **Kız**, **At**, **Tay**, **Kısrak** ve **Aygır** kelimelerini İmer ve diğerleri (2011)'deki açıklamadan yola çıkarak anlambirimler olarak düşünmek mümkündür. Diğer bir yandan yeşil ile işaretli olan **Erkek**, **Dişi**, **Yetişkin**, **İnsan** ve **Hayvan** kavramları ise bu anlambirimlerin taşıdığı öznitelikler yani anlambirimciklerdir. Dolayısıyla her bir her bir kelimenin bağlamda üstlendiği anlambirim, bağlama göre pozitif ya da negatif olarak analiz edilen bu anlambirimcikler toplamından elde edilir. Örneğin, **Adam** kelimesi burada şu şekilde tanımlanabilir: Adam = [+Erkek] + [−Dişi] + [+Yetişkin] + [+İnsan] + [−Hayvan].

Guiraud (1999: 112)' de yukarıdaki gibi bir analizin genişçe bir metne uygulanması durumunda milyarlarca bağlantının söz konusu olacağından ve bu tür bir işin üstesinden gelebilecek bilgisayarların daha uzun bir süre mevcut olmayacağından bahsedildiği görülür. Ancak Mikolov ve arkadaşları tarafından keşfedilen **Word2vec** ile Pennington ve arkadaşları tarafından geliştirilen **GloVe** sayesinde bu düş günümüzde mümkün hale gelmiştir.

Mikolov ve diğerleri (2013a, 2013b)'de de benzer şekilde Bileşen Analizi ile Anlam Çözümlemesi yapılmakta olup, vektör haline dönüşmüş kelime değerleri üzerinden aşağıdaki şekilde bir hesaplama yapıldığı görülür.

Tablo 11: Word2vec'te Bileşen Analizi ile Anlam Çözümlemesi (1)

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Kaynak: Mikolov ve diğerleri (2013a: 10).

Yukarıdaki tabloda en sol taraftaki sütunda örneğin **Paris** kelimesinin ilişkili özniteliklerinden biri olan **France** kelimesi negatif yönde ilave edilmiş ve bu işlem neticesinde elde edilen değere **Italy** eklenerek **Rome** vektörel değerine ulaşılmıştır. Bu durum, matematiksel olarak Mikolov ve diğerleri (2013a: 9)'da, "çıkarma" işlemi olarak ele alınarak **Paris - France + Italy = Rome** olarak gösterilmiştir. Aslında burada Fransa'dan Paris çıkarılacakmış gibi ilk etapta düşünülse de, bu tür bir işlem yapıldığında geriye Fransa'yı Fransa yapan diğer öznitelikler yine kalacağından bunun İtalya ile toplanması hatalı sonuçlar üretecektir. Dolayısıyla, Paris'i Fransa'nın başkenti yapabilme değerler bütününe sahip olan Fransa'yı eğer Paris'ten çıkaracak olursak geriye sadece tarafsız bir "başkent" ya da "o ülkenin en önemli kenti" ya da "o ülkenin sembol kenti" olma gibi değerler kalacaktır. Bu değeri herhangi bir başka ülke ile topladığımızda ise toplanan ülkenin "başkenti", "o ülkenin en önemli kenti" ya da "o ülkenin sembol kenti" değerine ulaşılmış ve sonuç olarak Roma kelimesi elde edilmiştir. Bunu en iyi şekilde kanıtlar ilişki aslında Example 3'te görülmektedir. **Florida** bir ülke olamasa da, bu eyaletin baş şehri olan **Talahassee** sonucuna aynı şekilde ulaşılmıştır. Burada eğer işlemimiz farklı şekilde, **Fransa – Paris + Rome** olmuş olsaydı, işte bu durumda **İtalya** sonucunu elde ederdik. Önemli olan burada çıkarma işlemi yapıldığında, çıkarılan yerine toplama olarak gelecek değerin, kendi anlam alanı içerisinde benzer düzlemde olmasıdır. Diğer bir deyişle, "Paris"ten "ülke" kavramı çıkarılıyor ise yine başka bir "ülke"

kavramı ile toplanmalı ki, doğru ve istenilen sonuçlar elde edilebilsin. Yukarıdaki tablonun devamında bu durum ilk örnekte açıkladığımızla benzer şekilde diğerleri için de hesaplanmış ve elde edilen sonuçlar Example 1, Example 2 ve Example 3 sütunlarında satırlara göre verilmiştir.

Yine benzer şekilde ancak bu kez o ülkelerdeki diğer özelliklerden faydalanarak elde edilen farklı bir uygulama örneği Mikolov ve diğerleri (2013b: 7)'de de aşağıdaki şekilde ele alınmıştır.

Tablo 12: Word2vec'te Bileşen Analizi ile Anlam Çözümlemesi (2)

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Kaynak: Mikolov ve diğerleri (2013b: 7).

Yukarıdaki tabloda, ülkeler ile parabirimi (currency), başkent (capital), havayolları (airlines), nehir (river) ve aktris (actress) gibi değerler toplanmış ve neticesinde elde edilen sonuçlar tabloda gösterilmiştir. Tabii ki belirli bir değer çıkarılmadan direk iki ayrı vektörün toplama işlemi gerçekleştirildiğinden sonuçların doğruluk oranı bir önceki tabloya kıyasla değişkenlik göstermekte olup, modelin en yakın olarak tahminde bulunabildiği ilk dört sonuç yukarıda verilmiştir.

2.5. KELİME BENZERLİKLERİ İÇİN WORD2VEC KULLANIMI

Derlem tabanlı teknikler kullanılarak elde edilen kelime vektörleri genellikle Word2vec ya da Glove tekniklerine göre oluşturulmaktadır. Yine Word2vec'in devamı niteliğinde olan ve Facebook'u kullanan FastText gibi versiyonlar da bulunmaktadır. Bu çalışmanın bir sonraki bölümü olan uygulama aşamasında Python kodları ve Gensim kütüphanesinden faydalanılmıştır. Gensim kütüphanesinde sırasıyla ['fasttext-wiki-news-subwords-300', 'conceptnet-numberbatch-17-06-300', 'word2vec-ruscorpora-300', 'word2vec-google-news-300', 'glove-wiki-gigaword-50', 'glove-wiki-gigaword-100', 'glove-wiki-gigaword-200', 'glove-wiki-gigaword-300', 'glove-twitter-25', 'glove-twitter-50', 'glove-twitter-100', 'glove-twitter-200',

'__testing_word2vec-matrix-synopsis'] şeklinde ön eğitimli modeller bulunmaktadır. Ancak çalışmada Gensim kütüphanesinden model yükleme aşamasında yararlanılsa da, ön eğitimli modeller hususunda bu yukarıdaki modellerle sınırlı kalınmamış ve çalışmanın gereksinimlerine göre farklı ön eğitimli kelime vektör modellerinden faydalanılmıştır.

Bununla birlikte, GoogleNews'un geliştirilmiş derinlikle daha yaygın bir kelime dağarcığını kapsayarak oldukça iyi sonuçlar verdiğine ve Wikipedia'nın büyük bölümlerinin çoğunun günlük kelimeleri kapsadığı için özel konular içerdiğinden kelimeler arasındaki benzerliği belirlemede pek yardımcı olamadığına ve ayrıca GoogleNews'un GLOVE gibi modern tekniklere kıyasla daha iyi referanslı veri setlerini bünyesinde bulundurduğuna ilişkin bilgiler bulunmaktadır (Lofi, 2015: 499). Ancak, kullanılacak tekniklerden hangisinin çalışma için daha uygun olacağı ya da hangi ön eğitimli kelime vektör modellerinden faydalanılacağı konusu, bir sonraki bölüm olan kelime destek modülü geliştirme aşamasına bırakılmıştır. Word2vec tekniğinde genel itibariyle hareket eden pencere genişliği mantığı kullanması belirli bir bağlamdaki kelimeler arası ilişkilere öncelik tanıması açısından önemlidir. Fakat diğer bir yandan derlemin genişliği ve aranılan kelimenin o derlem içerisinde bulunması da önem taşımakta olduğundan, çalışmamızda kullanılacak kaynaklar sadece tek bir teknikle sınırlandırılmamıştır. Dolayısıyla, gerek GloVe tekniğine göre olsun, gerekse Word2vec tekniğine göre elde edilmiş olsunlar, derlem tabanlı olarak hazırlanmış kelime vektörlerinin tamamının *kelime vektörü* kapsamında ele alınabilmesinin mümkün olduğu düşünülmektedir. Örneğin, bu çalışmada her ne kadar tekniklerinde farklılıkların mevcut olduğu bilinse de, genel bir çerçeveden bakılarak Word2vec ve FastText'in her ikisinden de faydalanılmış ve bu ikisi aynı temellere dayandığından çalışmada bu ikisinin temsili adı "Word2Vec" olarak, "Word2"den sonraki "V" harfini büyük şekilde yazarak bu ikisini temsil edecek şekilde adlandırılmıştır.

Dolayısıyla çalışmada oluşturulan modelde "Word2Vec" adı altında Word2vec ve onun daha gelişmiş versiyonu olan FastText[18] ön eğitimli kelime vektörü modelleri gereksinim duyulan dillere göre kullanılmıştır.

[18] Word2vec'e kıyasla, aslen n-gram olarak sözcükten daha küçük birimleri alarak işler.

ÜÇÜNCÜ BÖLÜM
KELİME DESTEK MODÜLÜ GELİŞTİRİMİ

Birinci bölümde de bahsedildiği üzere kelime benzerliklerini elde etmede Bilgi Tabanlı (Knowledge-based) ve Derlem Tabanlı (Corpus-based) olmak üzere genellikle iki farklı teknikten yararlanılmaktadır. Lofi (2015: 499)'te bu iki tekniğin hibritlenmesi neticesinde, diğer bir deyişle bilgi tabanlı teknikler ile derlem tabanlı tekniklerin birleştirilmesiyle kelimelerin benzerlik ve ilişki sorunlarıyla başa çıkabilen güçlü algoritmaların elde edilebileceğinden bahsedilmektedir. Yine benzer bir şekilde, Suárez ve Palomar (2002: 7)'de de, kelime ilişkileri ve kelimelerin birbirlerine olan bağımlılıkları ile özellik seçimi hususunda yeni bir strateji geliştirebilmek için yapmış oldukları çalışmada bu iki teknik arasında işbirliğine dayalı yeni bir yöntem geliştirme çabası içerisinde olduklarından bahsedilir. Dolayısıyla bu durum, yukarıdaki çalışmalarda da bahsedildiği üzere, bilgi tabanlı ve derlem tabanlı tekniklerin hibritlenerek birleştirilmesi neticesinde, kelime ilişkileri, sorunları ve benzerlikleri hususunda yeni yöntem ve metotların elde edilebilmesinin mümkün olduğuna işaret etmektedir.

Bu çalışmada da bu iki yöntemin hibirtlenerek model oluşturulması yoluna gidilerek, Bilgi Tabanlı (Knowledge-based) ve Derlem Tabanlı (Corpus-based) tekniklerin her ikisinden de faydalanılmıştır. Bilgi Tabanlı (Knowledge-based) tekniğe dayalı olarak WordNet'ten ve Derlem Tabanlı (Corpus-based) tekniğe bağlı olarak ise 3 farklı Word2Vec ön eğitimli kelime vektörü modelinden faydalanılmıştır. Neticesinde ise model çıktısı değerlendirilerek modelin güçlü ve zayıf yönleri tartışılarak ortaya konulmuştur. Oluşturulan modele Model_v.1 adı verilmiş ve kullanım kaynakları sırasıyla aşağıdaki şekilde özetlenmiştir.

Model_v.1: WordNet + Word2Vec (modelja1 + modelja2 + modelja3)

Hibrit bir model olarak tasarlanan Model_v.1 neticesinde aranan anahtar kelimeye ve bu kelime neticesinde WordNet'ten dönen kelime sayısına göre modelin hibritlik oranı yüzdelik anlamda değişkenlik gösterse de, sonuç olarak hibrit bir model olma kısmı değişmemektedir.

3.1. MODEL_V.1: WORDNET + WORD2VEC

Model_v.1 temel olarak, bilgi tabanlı bir sistem olarak oluşturulan WordNet ile derlem tabanlı bir sistem olan Word2Vec üzerine kurulan ve yarı yarıya hibrit olarak nitelendirebileceğimiz bir modeldir. Kullanıcıdan alınan input girdi kelimesi temel alınarak bu kelimeye ilişkin bilgiler öncelikle WordNet üzerinden çekilir. Devamında ise elde edilen bu bilgiler, derlem tabanlı olarak eğitilmiş olan Word2Vec'e gönderiler ve WordNet'en dönen kelime sayısı dikkate alınarak her bir kelimenin benzer kelimelerine ulaşılarak elde etmek istediğimiz alana ait külliyat oluşturulur.

Model_v.1'in tamamında, Python kodları kullanılmış ve model "Anaconda3-2020.11-Windows-x86_64" versiyonu üzerinde yüklü olarak gelen "Jupiter" üzerinde gerçekleştirilmiştir. Çalışmada kullanılan donanım mevcut imkanlar dahilinde "Merkezi İşlemci: Intel(R) Celeron(R) N4020 CPU @ 1.10GHz " ve "Hafıza: 4,00 GB(3,65 GB usable) RAM" olup, donanım üzerinde koşan sistemin özelliği ise 64 bitlik mimariye sahip Windows 10 Home Single Language şeklindedir.

3.2. MODEL_V.1 İÇİN GEREKLİ KÜTÜPHANE ve KAYNAKLAR

Model_v.1 için gerekli kütüphane ve modüller ise, WordNet verilerine ulaşabilmek için "NLTK" kütüphanesi ve bileşenleri; ön eğitimli Word2Vec kelime vektörü modellerini çalışmada kullanabilmek için "Gensim" kütüphanesi ve bileşenleri; WordNet'ten dönen kelimeler üzerinde gerekli temizleme işlemlerini yapabilmemiz için "re (Regular Expression)" modülü; kullanıcıdan alınacak Türkçe input girdisi başta olmak üzere diğer kaynaklardan dönen kelimeleri de hedef dillere çevirebilmek için "googletrans" kütüphanesinden "Translator" modülü; Japonca olarak hazırlanacak kelime çıktılarının okunuş bilgisini de kullanıcıya verebilmek amacıyla "pykakasi" kütüphanesi ile kullandığımız Anaconda versiyonu ve dolayısıyla Python versiyonu 2020'ye ait olduğundan dolayı zaman zaman aldığımız "DeprecationWarning" adlı kategorisel uyarıları kapatmak amacıyla "warnings" modülü şeklindedir.

Model_v.1'de faydalanılan Word2Vec ön eğitimli modeli olarak ise 3 farklı derlem kullanılmıştır. İlki https://github.com/singletongue/WikiEntVec/releases adresinde bz2 sıkıştırma formatıyla indirmeye hazır bir şekilde karşımıza gelen ön eğitimli Japonca kelime vektörü modelleri listesinde bulunan ve Japonca Wikipedia verileri üzerinde 20 Mayıs 2019 tarihinde eğitilmiş olan "jawiki.all_vectors.200d.txt" dosyasıdır. İkincisi ise, Japonya – Tohoku Üniversitesi, Enformatik Enstitüsü, Sistem Bilimleri Bölümünde görevli Masatoshi Suzuki tarafından web sitesinde yayınlanan ve 1 Şubat 2017 tarihli "ja-entity_vectors.txt" isimli dosyadır ve http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/ web adresi üzerinden indirilmiştir. Üçüncüsü ise FastText sitesi üzerinde pek çok dil seçenekleriyle sunulan Word2Vec dosyaları arasından Japonca dilinde ön eğitimli model olan "cc.ja.300.vec" dosyasıdır ve https://fasttext.cc/docs/en/crawl-vectors.html adresinden indirilmiştir. Bu ön eğitimli model dosyalarından ilki olan "jawiki.all_vectors.200d.txt" dosyası 200 boyutlu olup, 1,511,783 adet Japonca kelime vektörü içermektedir. İkincisi olan "ja-entity_vectors.txt" de yine 200 boyutludur ancak bu dosya 1,015,475 adet Japonca kelime vektöründen oluşmuştur ve wikipedia makale ve varlıklarına dayanmaktadır. Üçüncüsü olan "cc.ja.300.vec" ise 2,000,000 Japonca kelimeden oluşmaktadır ve Facebook verisi üzerinden eğitilmiştir. Bu vektör dosyalarının üçünden de faydalanılmasındaki en büyük sebep, her birinden dönen benzer kelimelerin farklı olmasından kaynaklanmaktadır.

3.3. MODEL_V.1 KURULUMU ve ÇALIŞTIRILMASI

Model_v.1'de temel olarak iki aşama söz konusudur. Öncelikle, kullanıcıdan alınan anahtar girdisinin WordNet üzerinde aranıp bulunduktan sonra bu anahtar kelimeye ait alt alan bilgisinin döndürülmesi aşaması ilk adımı oluşturmaktadır. Bunun için öncelikle gerekli olan kütüphaneler yüklenir ve kullanıcıdan aşağıda görüldüğü şekilde Türkçe olarak alana ilişkin bir anahtar kelime istenir. Burada tamlama şeklinde olabileceği de düşünülerek anahtar kelime sayısı maksimum 2 kelime olabilecek şekilde model tasarlanmıştır. Kullanıcıdan Türkçe olarak alınan girdi kelimesi bilgi tabanlı ontolojik veriye dayanan WordNet'te aranabilmesi için google translate'in "Translator" modülü vasıtasıyla İngilizceye çevrilir. Bu çeviriden

dönen çevrilmiş girdi daha sonra bir "senskey" adında bir liste haline dönüştürülür ve her bir indise bir kelime gelecek şekilde değeri atanır. Burada liste uzunluğu eğer 1 kelimeden oluşuyorsa devamındaki işlemler 1 kelime üzerinden sürdürülür. Eğer 2 kelimeden oluşuyor ise, bu durumda kelimelerin bir arada bulunma durumu, ilk kelimenin bulunma durumu ve ikinci kelimenin bulunma durumu olmak üzere 3 kombinasyonun değerlendirilmesi yapılır.

Şekil 14: Model_v.1 - 1. Adım

```
######################## Model Kurulumu ##########################.

[6]: from googletrans import Translator
     cevirmen = Translator()
     keytr = input("Anahtar:").lower()
     keying = cevirmen.translate(keytr, dest='en').text
     senskey = []
     sn = keying.split(" ")
     for i in sn:
         senskey.append(i)
     print("\nİngilizce eşdeğerliği:", keying)
     print(senskey)

     Anahtar:dilbilim

     İngilizce eşdeğerliği: linguistics
     ['linguistics']

[7]: from nltk.corpus import wordnet as wn

     if len(senskey) == 2:
         xlist = []
         sozcuk = senskey[0]+"_"+senskey[1]
         listem1 = wn.synsets(sozcuk)
         if listem1 != 0:
             yenilist1= []
             for i in listem1:
                 yenilist1.append(i)
             for i in yenilist1:
                 xlist.append(i.name())
         if listem1 != 1:
             sozcuk = senskey[0]
             listem1 = wn.synsets(sozcuk)
             if listem1 != 0:
                 yenilist1= []
                 for i in listem1:
                     yenilist1.append(i)
             sozcuk = senskey[1]
             listem1 = wn.synsets(sozcuk)
             if listem1 != 0:
                 for i in listem1:
                     yenilist1.append(i)
             for i in yenilist1:
                 xlist.append(i.name())
         print(xlist)
         print(len(xlist))
     else:
         sozcuk = senskey[0]
         liste = wn.synsets(sozcuk)
         ylist= []
         for i in liste:
             ylist.append(i)
         xlist = []
         for i in ylist:
             xlist.append(i.name())
         print(xlist)

     ['linguistics.n.01', 'linguistics.n.02']
```

Kaynak: Yazar tarafından derlenmiştir.

Yukarıdaki kaynak kodunda da görüldüğü gibi tek kelimelik bir input girdisi olan "dilbilim" anahtar kelimesi kullanıcıdan alındığı bir durum senaryosu bu

çalışmada değerlendirilmiştir. Ancak bu aşamada eğer iki kelimelik bir anahtar girdisi alınsa idi ne şekilde olurdu sorusuna yanıt olarak aşağıda "araba tekeri" şeklinde iki kelimelik bir input girdisinin kullanıcıdan alındığı durum da örnek olarak gösterilmiştir.

Şekil 15: Model_v.1 - Örnek 2 kelimelik girdi durumu

```
for i in sn:
    senskey.append(i)
print("\nİngilizce eşdeğerliği:", keying)
print(senskey)
```

```
Anahtar:araba tekeri

İngilizce eşdeğerliği: car tire
['car', 'tire']
```

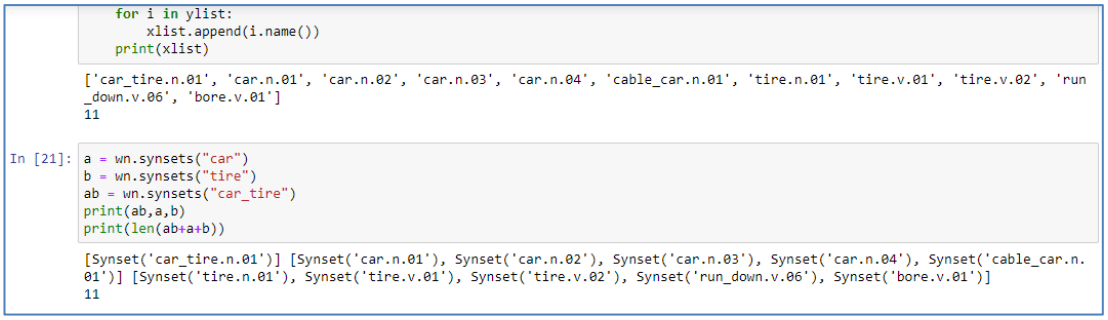
```
In [7]: from nltk.corpus import wordnet as wn

if len(senskey) == 2:
    xlist = []
    sozcuk = senskey[0]+"_"+senskey[1]
    listem1 = wn.synsets(sozcuk)
    if listem1 != 0:
        yenilist1= []
        for i in listem1:
            yenilist1.append(i)
        for i in yenilist1:
            xlist.append(i.name())
    if listem1 != 1:
        sozcuk = senskey[0]
        listem1 = wn.synsets(sozcuk)
        if listem1 != 0:
            yenilist1= []
            for i in listem1:
                yenilist1.append(i)
        sozcuk = senskey[1]
        listem1 = wn.synsets(sozcuk)
        if listem1 != 0:
            for i in listem1:
                yenilist1.append(i)
        for i in yenilist1:
            xlist.append(i.name())
    print(xlist)
    print(len(xlist))
else:
    sozcuk = senskey[0]
    liste = wn.synsets(sozcuk)
    ylist= []
    for i in liste:
        ylist.append(i)
    xlist = []
    for i in ylist:
        xlist.append(i.name())
    print(xlist)
```

```
['car_tire.n.01', 'car.n.01', 'car.n.02', 'car.n.03', 'car.n.04', 'cable_car.n.01', 'tire.n.01', 'tire.v.01', 'tire.v.02', 'run
_down.v.06', 'bore.v.01']
11
```

Kaynak: Yazar tarafından derlenmiştir.

Burada dikkat edilecek olursa "araba tekeri" girdisinin İngilizce eşdeğerliği olan "car tire" anahtarının 3 kombinasyonu da hazırladığımız kod sayesinde değerlendirilmeye alınmaktadır. Bunu aşağıdaki şekilde görüntülenen ekstra kod ile de teyit etmek mümkündür.

Şekil 16: Model_v.1 - Örnek 2 kelimelik girdi dönütü



```
    for i in ylist:
        xlist.append(i.name())
    print(xlist)

['car_tire.n.01', 'car.n.01', 'car.n.02', 'car.n.03', 'car.n.04', 'cable_car.n.01', 'tire.n.01', 'tire.v.01', 'tire.v.02', 'run
_down.v.06', 'bore.v.01']
11

In [21]: a = wn.synsets("car")
         b = wn.synsets("tire")
         ab = wn.synsets("car_tire")
         print(ab,a,b)
         print(len(ab+a+b))

         [Synset('car_tire.n.01')] [Synset('car.n.01'), Synset('car.n.02'), Synset('car.n.03'), Synset('car.n.04'), Synset('cable_car.n.
         01')] [Synset('tire.n.01'), Synset('tire.v.01'), Synset('tire.v.02'), Synset('run_down.v.06'), Synset('bore.v.01')]
         11
```

Kaynak: Yazar tarafından derlenmiştir.

Tek input girdisi olan "dilbilim" kelimesi ile Model_v.1'e devam ettiğimiz durum senaryosuna geri dönecek olursak, öncelikle bu kelimeye ilişkin WordNet'teki alt kavramlara ulaşabilmek için bir "hyponim" adında bir fonksiyon tanımlanmıştır. Ancak burada dikkat edilmesi gereken bir durum daha söz konusudur. WordNet yapısı gereği iç içe geçmiş kümeleme mantığına sahip bir hiyerarşiye sahiptir. Diğer bir deyişle, ilk aşamada dönen ve sonu ".n.01" ya da ".n.02" … gibi dönen veriler aslında bir küme adı olup bunların içerisinde de başka alt kavram kelimeleri ve o kelimeleri altında da başka alt kavram kelimeleri mevcut olabilmektedir. Daha sonraki adımda ise, hazırlanan bu fonksiyon "altkavram" adı altında oluşturulan başka bir fonksiyonun içerisinde kullanılmış ve gerektiği takdirde de 10 kat derinliğe kadar inebilen bir yapıda tasarlanmıştır. Böylece her seferinde eğer bir alt kümenin içi boş değilse bir iç döngüye girerek alt kümelerdeki kavramları da karşımıza getirebilmektedir. Burada bu derinliğin 10 kat olarak seçilmesinin ana sebebi, Sarkar ve Law (2015)'te de bahsedildiği üzere WordNet hiyerarşisinde, isim türündeki kelimeler için maksimum 25 ve fiil türündeki kelimeler için ise maksimum 15 katmana sahip bir tasarımın söz konusu olmasından kaynaklanır. Bu sebeple, bu derinliğin bu iki rakamın en büyük ortak böleninin iki katı olarak belirlenmesi uygun görülmüştür. Bu değer daha sığ olarak, örneğin 5 şeklinde alınması durumunda özellikle isim türündeki kelimelerin sadece 5 alt kavramının, diğer bir deyişle maksimum küme sayısının[19] sadece 5'te 1'ine ulaşılacağından oldukça az sayıda veri elde edilecektir. Fakat diğer bir yandan daha büyük alınacak olduğu takdirde ise, bu

[19] Her ne kadar pratikte bu tür bir duruma pek rastlanmasa da, en üst seviyede bir kelimenin örneğin "varlık" gibi hiyerarşide en üst düzeyde bulunabilecek bir kelimenin input olarak girilmesi durumunda.

durum hem çeviri kısıtları açısından olası problemlere, hem de alt alta çok fazla sayıda kelime hiyerarşisi söz konusu olacağından, başlangıç kelimesine göre kavramsal açıdan göze çarpan sapmalara ve git gide uzaklaşan kavramların elde edilmesine sebebiyet verecektir. Aşağıda hazırlanan bu fonksiyonlar görüntülenmektedir. Daha sonra tanımlanan bu fonksiyon anahtar kelime ile WordNet'ten döndürdüğümüz bu ilk küme kelime listesi olan ve ['linguistics.n.01', 'linguistics.n.02'] şeklinde iki indislik veri barındıran "xlist" adlı listeye "altkavram(xlist)" şeklinde uygulanır ve dönen veri bir değişkene atanarak alt veri kümelerinde aynı ismi taşıyan verilerin olup olmadığının sorgulanarak dönen verinin tekrar edilmemesi sağlanır.

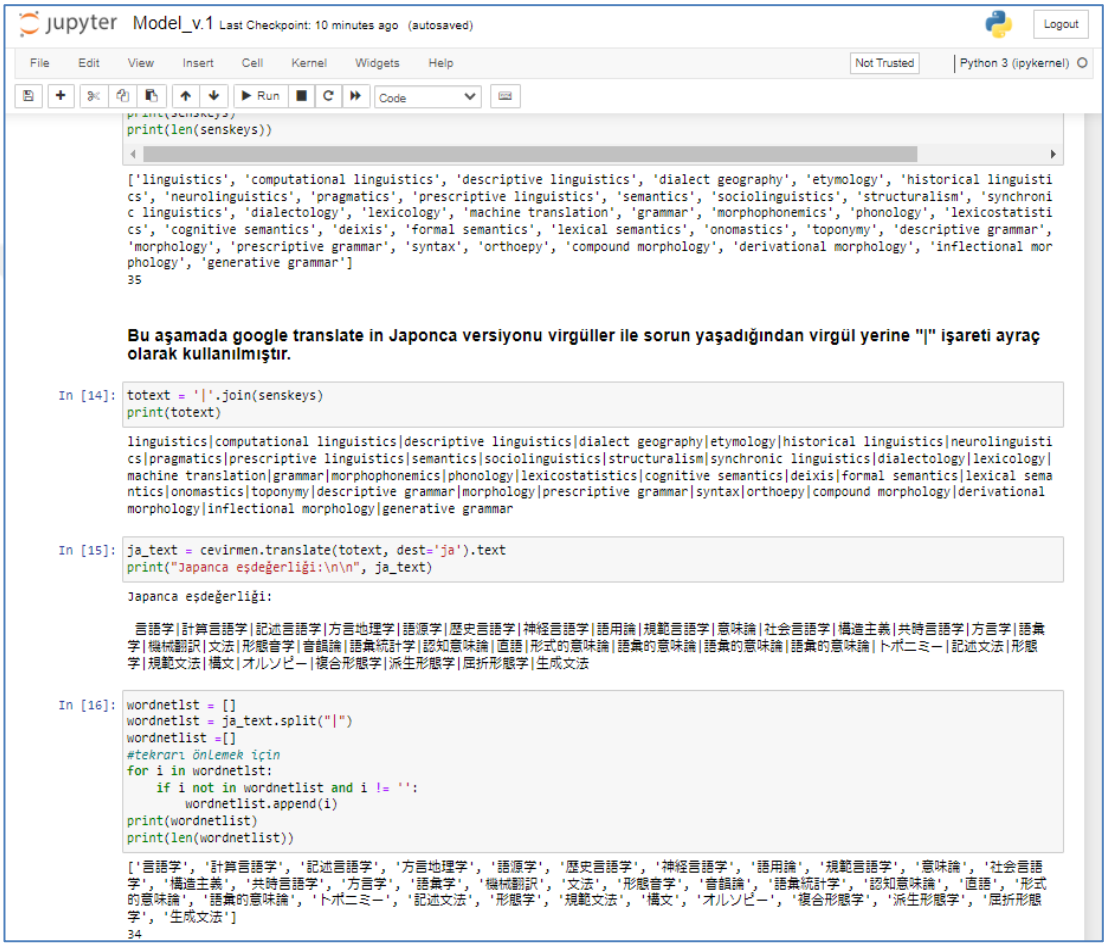
Şekil 17: Model_v.1 - Ana Fonksiyon

```
In [9]: def hyponim (liste):
            df = []
            for i in range (0,len(liste)):
                x = wn.synset(liste[i])
                z = x.hyponyms()
                df.append(z)

            dflist= []
            for i in df:
                for k in i:
                    dflist.append(k.name())
            return(dflist)

In [10]: def altkavram(x):

            limit = 100
            det = hyponim(x)
            listeler = []
            if det != 0:
                for i in det:
                    listeler.append(i)
                det = hyponim(hyponim(x))
                if det != 0 and len(listeler) < limit:
                    for i in det:
                        listeler.append(i)
                    det = hyponim(hyponim(hyponim(x)))
                    if det != 0 and len(listeler) < limit:
                        for i in det:
                            listeler.append(i)
                        det = hyponim(hyponim(hyponim(hyponim(x))))
                        if det != 0 and len(listeler) < limit:
                            for i in det:
                                listeler.append(i)
                            det = hyponim(hyponim(hyponim(hyponim(hyponim(x)))))
                            if det != 0 and len(listeler) < limit:
                                for i in det:
                                    listeler.append(i)
                                det = hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(x))))))
                                if det != 0 and len(listeler) < limit:
                                    for i in det:
                                        listeler.append(i)
                                    det = hyponim(hyponim(hyponim(hyponim(hyponim((hyponim(x)))))))
                                    if det != 0 and len(listeler) < limit:
                                        for i in det:
                                            listeler.append(i)
                                        det = hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(x)))))))
                                        if det != 0 and len(listeler) < limit:
                                            for i in det:
                                                listeler.append(i)
                                            det = hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim((hyponim(x))))))))))
                                            if det != 0 and len(listeler) < limit:
                                                for i in det:
                                                    listeler.append(i)
                                                det = hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(hyponim(x)
                                                if det != 0 and len(listeler) < limit:
                                                    for i in det:
                                                        listeler.append(i)

            return (listeler)
```

Kaynak: Yazar tarafından derlenmiştir.

Bir sonraki adımda ise gelen liste string formatına dönüştürülerek verinin “.n.01”, “.n.02”, vb. uzantılardan temizlenmesi sağlanır ve sonrasında anahtar sözcük ilk liste elemanı olacak şekilde bu veri ile birleştirilir. Sonraki aşamada ise google translate modülü vasıtasıyla çeviri işlemine geçilir.

Şekil 18: Model_v.1 - Çeviri Adımı



Kaynak: Yazar tarafından derlenmiştir.

Burada dikkat edilmesi gereken bir husus, google translate Python modülünün sadece string (text) tarzında veri kabul ettiği unutulmamalıdır. Ayrıca bir sonraki kısımda ele alındığı üzere google translate Python modülü ücretsiz sürümünün bir takım sınırlamaları da mevcuttur. Yukarıdaki adımda İngilizce olarak WordNet’ten dönen kelimeler Japoncaya çevrilir ve tekrar bir liste haline getirilerek ilk, Model_v.1’in ilk aşaması tamamlanmış olur.

Model_v.1'in ikinci aşamasında WordNet'ten gelen bu kelimelerin derlem tabanlı bilgiye dayalı ön eğitime sahip Word2Vec modeli içerisinde taranarak var olan kelimelerin benzerlerine erişim aşamasından oluşur. Bu aşamada öncelikle gensim modülü import edilerek "modelja1" adındaki bir değişkene "gensim.models.keyedvectors.KeyedVectors.load_word2vec_format('jawiki.all_vectors.200d.txt', binary = False)" kodu atanır. Bu sayede "jawiki.all_vectors.200d" adlı ön eğitimli kelime vektörü modeline erişimimiz "modelja1" ile sağlanmış olur. Ancak bu modelin yüklenmesi mevcutta bulunan donanıma göre biraz vakit almaktadır. Çalışmada kullanılan donanımda yaklaşık 3-4 dk tutmuştur ancak bu süre yüklenen dosya büyüklüğüne göre de değişim gösterebilir. Özellikle ön eğitimli derlem modellerinin her birinin yüklenmesi çok yüksek olmayan donanımlarda 3 ila 7 dk gibi süreler alabilmektedir. Bu sebeple, eğer birden fazla anahtar kelime aranacaksa, her seferinde bu kadar süre beklenmemesi amacıyla Jupiter arayüzü kullanımında ön eğitimli model tanımlama kodlarının ayrı bir satırda yazılarak sadece bir kez çalıştırılması yeterli olacağından kullanıcıya hız kazandıracaktır. Bir diğer husus da iki tip ön eğitimli modelin söz konusu olmasıdır, uzantısı ".bin" olan gurup ile ".txt" ya da ".vec" olan tipte ön eğitimli kelime vektör dosyası bulunabilmektedir. Bunlarda "bin" dosyaları modele ilişkin alt parametreleri de bünyesinde barındırır ve çoğunlukla farklı model eğitimi, optimizasyonu gibi işlemler için ihtiyaç duyulur. Ancak sadece bir kelimeye ilişkin yakınlık, benzerlik gibi bilgilerin hesaplanması ya da istenilmesi gibi durumlar için ".vec" ve ".txt" uzantılı kelime vektör bilgisinin bulunduğu dosya yeterlidir ve sadece bu amaçla kullanılacaksa ".bin" dosyasına göre daha iyi performans verir. Kodda bu durum "binary = False" parametresi ile kontrol edilmektedir. Ön eğitimli bir ".vec" ya da ".txt" dosyasının içine bakılacak olursa kelime boyutuna göre, bir kelimenin karşısında boyut sayısına göre hesaplanmış ondalıklı rakamlarla karşılaşılır. İşte vektörel yakınlık ve uzaklıklar da bu rakamlara göre hesaplanmakta olup, bu sebeple de çalışmada hem amaca uygunluğu hem de mevcut donanımla dosya büyüklüğü açısından daha uyumlu olduğundan ".vec" ve ".txt" uzantılı ön eğitimli kelime vektörü model dosyaları kullanılmıştır.

Şekil 19: Model_v.1 - Birinci Word2Vec yüklenmesi

```
In [18]: import gensim
         modelja1 = gensim.models.keyedvectors.KeyedVectors.load_word2vec_format('jawiki.all_vectors.200d.txt',
                                                                                 binary = False)

In [19]: if len(wordnetlist) != 0:
             jp1_sayi = 0
             jp1list=[]
             for i in range(0, len(wordnetlist)):
                 word = wordnetlist[i]
                 if word in modelja1:
                     jp1list.append(i)
                     print("Modelja1 Listeden modelde var olan list no",i, word)
                     jp1_sayi += 1
             print("\njawiki.all_vectors Word2Vec Listeden Modelde Var Olan Kelime Sayısı: ", jp1_sayi)

         Modelja1 Listeden modelde var olan list no 0 言語学
         Modelja1 Listeden modelde var olan list no 4 語源学
         Modelja1 Listeden modelde var olan list no 5 歴史言語学
         Modelja1 Listeden modelde var olan list no 7 語用論
         Modelja1 Listeden modelde var olan list no 9 意味論
         Modelja1 Listeden modelde var olan list no 10 社会言語学
         Modelja1 Listeden modelde var olan list no 11 構造主義
         Modelja1 Listeden modelde var olan list no 13 方言学
         Modelja1 Listeden modelde var olan list no 15 機械翻訳
         Modelja1 Listeden modelde var olan list no 16 文法
         Modelja1 Listeden modelde var olan list no 18 音韻論
         Modelja1 Listeden modelde var olan list no 20 認知意味論
         Modelja1 Listeden modelde var olan list no 25 記述文法
         Modelja1 Listeden modelde var olan list no 26 形態学
         Modelja1 Listeden modelde var olan list no 28 構文
         Modelja1 Listeden modelde var olan list no 33 生成文法

         jawiki.all_vectors Word2Vec Listeden Modelde Var Olan Kelime Sayısı:  16
```

Kaynak: Yazar tarafından derlenmiştir.

Model_v.1'de ilk derlem tabanlı ön eğitimli Word2Vec modeline ulaşıldıktan sonra WordNet'ten dönen kelimelerden hangisinin bu model içinde bulunduğu yukarıda da görüldüğü gibi indis numaraları ile birlikte taranarak belirlenir. Örneğin ilk Word2Vec modelimizde sırasıyla "dilbilim(言語学), etimoloji(語源学), tarihsel dilbilim(歴史言語学), edimbilim(語用論), anlambilim(意味論), toplumdilbilim(社会言語学), yapısalcılık(構造主義), lehçeler bilimi(方言学), makine çevirisi(機械翻訳), gramer(文法), fonoloji(音韻論), bilişsel anlambilim(認知意味論), tanımlayıcı dilbilgisi(記述文法), morfoloji(形態学), sözdizimi(構文), üretken dilbilgisi(生成文法)" kelimelerinin ilk Word2Vec eğitimli model olan modelja1 içerisinde mevcut olduğu tespit edilmiştir. Bir sonraki adım ise dönen bu kelimelerin sayı miktarına, diğer bir deyişle modelja1 içerisinde bulunma frekansına göre bir kontrol mekanizması kurularak modelja1'de tespit edilen kelimelerin her birinin "tps" adı verdiğimiz bu değere göre benzer kelimelerinin tespit edilmesinden oluşmaktadır.

Şekil 20: Model_v.1 - Birinci Word2Vec dönütü



```
In [20]: if jp1_sayi <= 5:
             tps = 10
         elif jp1_sayi <= 10:
             tps = 5
         elif jp1_sayi <= 25:
             tps = 2
         else:
             tps = 1

         word2vec1=[]
         for i in range(0, len(wordnetlist)):
             word = wordnetlist[i]
             if word in modelja1:
                 result = modelja1.most_similar_cosmul(word, topn=tps)
                 for i in result:
                     word2vec1.append((i)[0])
         print(word2vec1)
         print (len(word2vec1))

['##言語学##', '比較言語学', '##哲学的言語##', '中世ラテン語', '##神山孝夫##', '比較言語学', '発話行為', '##語用論##', '##意味論##',
'統語論', '##西村義樹##', '語彙論', '##ハーバーマス##', 'ポスト構造主義', '##佐藤亮一_(言語学者)##', '##日野資純##', '##機械翻訳##',
'##SYSTRAN##', '##文法##', '語彙', '音韻', '##音韻論##', '##高見健一##', '##山梨正明##', '##寺村秀夫##', '語彙論', '発生学', '分類
学', '##予約語##', '##シンタックス##', '##山梨正明##', '##仁田義雄##']
32
```

Kaynak: Yazar tarafından derlenmiştir.

WordNet'ten gelen kelime listesine göre modelja1'de tespit edilen kelimelerin sayısı 16 olduğundan "tps" değeri yukarıda da görüldüğü üzere 2 olarak belirlenmiştir. Buradaki frekans değerlerinin daha yüksek alınmamasının sebebi, ana fonksiyonda WordNet kelimeleri limit değeri 100 olarak belirlendiğinden[20] model üzerindeki kelime sayısının bilgi tabanlı ve derlem tabanlı olarak dengeli ve homojen bir dağılım oluşturmasının sağlanması hedeflendiğinden kaynaklanır. Dolayısıyla yukarıda da göründüğü üzere 32 adet kelime modelja1 üzerinden döndürülmüştür.

Şekil 21: Model_v.1 - Birinci Word2Vec modelja1 kelime katkısı



```
sem1 = ",".join(word2vec1)
s1 = re.sub("##","",sem1)
s2 = re.sub("#","",s1)
s3 = list(s2.split(","))
sonlst1 =[]
#tekrarı önlemek için

for i in s3:
    if i not in sonlst1 and i != '':
        y = i.replace('[', '').replace(']', '')
        if y not in wordnetlist and y not in sonlst1:
            sonlst1.append(y.replace('[', '').replace(']', ''))

print (sonlst1)
print("\nmodelja1'den Model 1' e sağlanan kelime katkısı:", len(sonlst1))

wordn_ja1 = wordnetlist + sonlst1
#print("\n",wordn_ja1)
print("\n\033[1mWordnet listemize modelja1'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı: \033[0;34m\033[1m{}(Wor
      .format(len(wordnetlist), len(sonlst1), len(wordn_ja1)))

['比較言語学', '哲学的言語', '中世ラテン語', '神山孝夫', '発話行為', '統語論', '西村義樹', '語彙論', 'ハーバーマス', 'ポスト構造主義',
'佐藤亮一_(言語学者)', '日野資純', 'SYSTRAN', '語彙', '音韻', '高見健一', '山梨正明', '寺村秀夫', '発生学', '分類学', '予約語', 'シンタ
ックス', '仁田義雄']

modelja1'den Model 1' e sağlanan kelime katkısı: 23

Wordnet listemize modelja1'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı: 34(WordNet) + 23(modelja1) = 57
```

Kaynak: Yazar tarafından derlenmiştir.

[20] Bu değerin 100 olarak alınma sebebi bir sonraki kısım olan "3.4."te detaylı olarak ele alınmıştır.

Ancak yukarıda da görüldüğü üzere bu yeni kelimeler hamdır ve temizlenip WordNet'ten gelen kelime listesi içerisinde de aynı kelimenin bulunup bulunmadığının kontrol edilerek listeye ilave edilmesi gerekmektedir. Yukarıda hazırlanan kod ile tüm bu kontroller yapıldıktan sonra modelja1'den dönen kelime sayısı 32'den 23'e düşmüştür. Bunun sebebi ise benzer kelimelere bir kelimeden fazla kelime vasıtasıyla da derlem içerisinden ulaşılabilmesinden ya da fazladan köşeli parantez ya da parantez gibi bir takım karakterlerin mevcut olmasından kaynaklanır ki, bu durum bir kelimenin iki farklı kelime olarak makine tarafından değerlendirilmesine sebep olur. Tüm bu istenmeyen durumları ortadan kaldırmak için döndürülen verinin en saf hale gelene kadar temizlenmesini gerektirir. Bu tip ön işlemeye tabi tutulduktan sonra modelja1'den dönen ve Model_v.1'e katkı sağlayan yeni kelime sayısı 23 olmuştur.

İlk derlem tabanlı model olan modelja1'den dönen kelimeler belirli ölçüde katkı sağladıktan sonra, ikinci derlem tabanlı model olan "ja-entity_vectors.txt" dosyası gensim modülü vasıtasıyla "modelja2" adında bir değişkene tanımlanarak kullanıma hazır hale getirilir. Bu aşamada yukarıda modelja1'deki anlattığımız işlemler tekrarlanarak modelja2 derleminde içinde tespit edilen kelimelerin benzerleri modelja2'den istenir.

Şekil 22: Model_v.1 - İkinci Word2Vec modelja2 kelime katkısı

```
wordn_ja1_ja2 = wordn_ja1 + sonlst2
#print("\n",wordn_ja1_ja2)
print("\n\033[1mWordnet listemize modelja2'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı:\n\033[0;34m\033[1m{}(Wo
      .format(len(wordnetlist), len(sonlst1), len(sonlst2), len(wordn_ja1_ja2)))
```

```
Modelja2 Listeden modelde var olan list no 0 言語学
Modelja2 Listeden modelde var olan list no 5 歴史言語学
Modelja2 Listeden modelde var olan list no 7 語用論
Modelja2 Listeden modelde var olan list no 9 意味論
Modelja2 Listeden modelde var olan list no 10 社会言語学
Modelja2 Listeden modelde var olan list no 11 構造主義
Modelja2 Listeden modelde var olan list no 13 方言学
Modelja2 Listeden modelde var olan list no 15 機械翻訳
Modelja2 Listeden modelde var olan list no 16 文法
Modelja2 Listeden modelde var olan list no 18 音韻論
Modelja2 Listeden modelde var olan list no 26 形態学
Modelja2 Listeden modelde var olan list no 27 規範文法
Modelja2 Listeden modelde var olan list no 28 構文
ja-entity_vectors Word2Vec Listeden Modelde Var Olan Kelime Sayısı:  13

Modelja2'den gelen kelimeler:
 ['[言語学]', '[比較言語学]', '応用言語学', '英語学', '事大主義', '[数理]', '[意味論]', '統語論', '[民族誌]', '[言語類型論]', '実証主
義', '[記号論]', '[国語学]', '[日本語学]', '[国際化と地域化]', '[形態素解析]', '[文法]', '[語彙]', '形態論', '[言語類型論]', '発生学',
'[形態学]', '解釈学', '中国の歴史', '記法', '[正規表現]']
Modelja2'den gelen kelimelerin sayısı: 26


Tekrasız ve ayıklanmış sonlst2 liste:
 ['応用言語学', '英語学', '事大主義', '数理', '民族誌', '言語類型論', '実証主義', '記号論', '国語学', '日本語学', '国際化と地域化', '形
態素解析', '形態論', '解釈学', '中国の歴史', '記法', '正規表現']

modelja2'den Model 1' e sağlanan kelime katkısı: 17

Wordnet listemize modelja2'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı:
34(WordNet) + 23(modelja1) + 17(modelja2) = 74
```

Kaynak: Yazar tarafından derlenmiştir.

Modelja2 farklı bir derlem olduğundan WordNet'ten gelen kelimelerden modelja2 içerisinde tespit edilen kelime sayısı yukarıda da görüldüğü üzere bu kez 13'tür. Burada WordNet'ten dönen kelimelerden modelja1 ile modelja2'nin her ikisinde de aynı kelimelerin bulunması ve işlemlere bu şekilde devam edilmesi bir anlığına hatalı bir tutum olarak değerlendirilebilir, ancak değildir. Aynı kelimelere müsaade edilmesinin en büyük sebebi derlemler farklı tarihlerde ve dolayısıyla birebir aynı olmayan veriler üzerinden eğitildiğinden, aynı kelimeleri girdi olarak versek dahi benzer olarak tayin edilen kelimeler farklı olabilmektedir. Bu sebeple, derlem tabanlı ön eğitimli modellerin tümünden veri döndükten sonra aynı kelimelerin ayıklama işlemine geçilmiş ve benzer eğitim kaynaklarına sahip ön eğitimli modeller olsa dahi bu modellerin eğitim yılları bile farklı olsa bu sayede farklı yıllarda trend olabilen kelime benzerliklerine bile erişim imkanı elde edilebilmiş olmaktadır. Burada modelja1 içinde bulunan kelimeler ile modelja2 içinde bulunan kelimeler sadece 1 kelime farklı olup, bu da WordNet listesinde 27. indiste bulunan "kuralcı dilbilgisi(規範文法)" dir. Diğer bir deyişle modelja2'de tespit edilen 12 adet kelime modelja1'de de öncesinde tespit edilmiş ve bu kelimelere bağlı olan benzer kelimeler modelja1'den döndürülmüştür. Ancak oluşturduğumuz modelde her iki derlem modelinde bulunan kelime sayısına göre elde edilen tps değerleri de aynı olduğu halde, diğer bir deyişle sadece 1 adet kelimeye karşın 2 adet benzer kelime bulmak üzere programlanmış olmasına rağmen, modelja2'den dönen kelimelerden 17 adeti modelja1'den dönen kelimeler ile WordNet'ten dönen kelimeler arasında bulunmamaktadır. Eğer tamamen aynı olmuş olsa idi bu rakamın sadece 2 adet olması gerekirdi ki, aslen sonuç 17'dir. Buradan da anlaşılacağı üzere, aynı kelime olmasına rağmen her derlemin eğitim verileri benzer kaynakları kullanmış olsa dahi birebir, yani satırı satırına ve byte'ı byte'ına aynı olmadığı takdirde hesaplanan vektörel değerler farklılık göstereceğinden elde edilecek kelime benzerlilikleri de belirli ölçüde değişecektir. Bu sayede modelja2 sayesinde modelimize derlem tabanlı 17 adet yeni kelime daha eklenmiştir.

Bir sonraki adımda ise, Facebook verisine dayalı olarak eğitilen Word2Vec ön eğitimli derlem modeli olan "cc.ja.300.vec" gensim modülü yardımıyla kullanılmak üzere "modelja3" adlı değişkene değer olarak atanmıştır. Devamındaki adımlar modelja1 ve modelja2'deki ile aynıdır. Ancak, modelja3 içerisinde tespit edilen

kelimeler sadece 2 adet olduğundan hazırladığımız kod gereği tps frekansı değeri daha yüksektir ve neticesinde tespit edilen her bir kelime için 10 adet benzer kelime modelja3'ten döndürülmüştür. Bu kelimelerden yine WordNet'te önceden bulunan ve diğer derlem modelleri olan modelja1 ve modelja2 vasıtasıyla sağlanan yeni kelimeler içerisinde mevcut olmayanlar ayıklandığında ise geriye 15 adet yeni kelime kalmış ve modelja3 vasıtasıyla bu kelimeler Model_v.1'in çıktı listesine eklenmiştir.

Şekil 23: Model_v.1 - Üçüncü Word2Vec modelja3 kelime katkısı

```
#tekrarı önlemek için
for i in s3:
    if i not in sonlst3 and i != '':
        y = i.replace('[', '').replace(']', '')
        if y not in wordn_ja1_ja2 and y not in sonlst3 :
            sonlst3.append(y.replace('[', '').replace(']', ''))

print ("\n\nTekrasız ve ayıklanmış sonlst3 liste:\n",sonlst3)
print("Tekrasız ve ayıklanmış sonlst3 liste uzunluk:", len(sonlst3))


wordn_ja1_ja2_ja3 = wordn_ja1_ja2 + sonlst3
#print("\n",wordn_ja1_ja2_ja3)
print("\n\033[1mWordnet listemize modelja3'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı:\n\033[0;34m\033[1m{}(Wo
      .format(len(wordnetlist), len(sonlst1), len(sonlst2), len(sonlst3), len(wordn_ja1_ja2_ja3)))
```

```
Modelja3 Listeden modelde var olan list no 16 文法
Modelja3 Listeden modelde var olan list no 28 構文
cc.ja.300.vec Word2Vec Listeden Modelde Var Olan Kelime Sayısı:  2

Modelja3'den gelen kelimeler:
 ['構文', '文型', '語順', '語彙', '語法', '統語', '単語', '国文法', '文語文', '語い', '文法', '文型', '統語', '語順', '複文', 'ＳＶＯ
Ｃ', 'パーザ', '形態素', 'EBNF', '統辞']
Modelja3'den gelen kelimelerin sayısı: 20


Tekrasız ve ayıklanmış sonlst3 liste:
 ['文型', '語順', '語法', '統語', '単語', '国文法', '文語文', '語い', '複文', 'ＳＶＯＣ', 'パーザ', '形態素', 'EBNF', '統辞']
Tekrasız ve ayıklanmış sonlst3 liste uzunluk: 14

Wordnet listemize modelja3'den gelen yeni kelimeler eklenlendikten sonraki kelime sayısı:
34(WordNet) + 23(modelja1) + 17(modelja2) + 14(modelja3) = 88
```

Kaynak: Yazar tarafından derlenmiştir.

Buraya kadar olan adımlar ile listemiz tamamlanmış ve Model_v.1'e "dilbilim" anahtar kelimesi girildiğinde elde edilen katkı payları da aşağıdaki şekilde hesaplanmıştır.

Şekil 24: Model_v.1'de "dilbilim" anahtar kelimesi için kaynak dağılımı

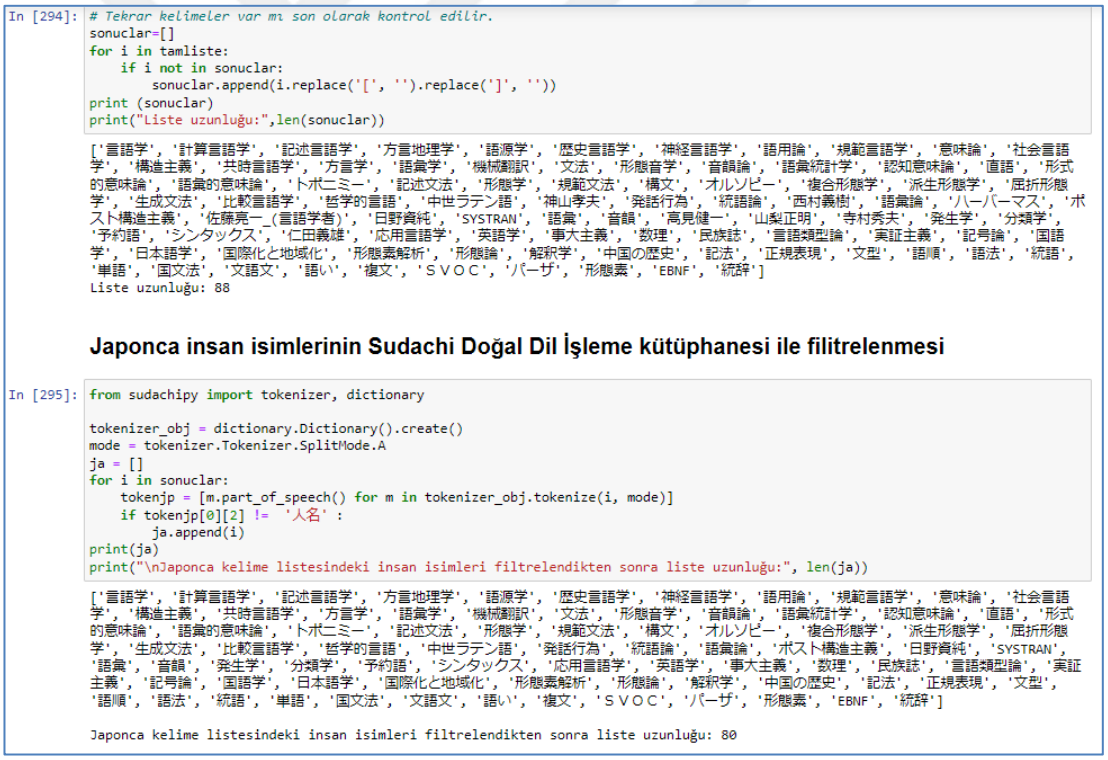
```
derlemlist=(sonlst1+sonlst2+sonlst3)
print("\033[0;30m\033[1mModel 1'deki Bilgi Tabanlı (Knowled-based) kelime sayısı: {} \nModel 1'deki Bilgi Tabanlı (Knowled-based)
print("\n\033[0;30m\033[1mModel 1'deki Derlem tabanlı (Corpus-based) kelime sayısı: {} \nModel 1'deki Derlem Tabanlı (Corpus-base
print("\n\033[0;30m\033[1mSudachi öncesi kelime listeleri toplam uzunluğu:",len(derlemlist)+len(wordnetlist))
```

```
Model 1'deki Bilgi Tabanlı (Knowled-based) kelime sayısı: 34
Model 1'deki Bilgi Tabanlı (Knowled-based) yüzdelik payı: %38.64

Model 1'deki Derlem tabanlı (Corpus-based) kelime sayısı: 54
Model 1'deki Derlem Tabanlı (Corpus-based) yüzdelik payı: %61.36

Sudachi öncesi kelime listeleri toplam uzunluğu: 88
```

Kaynak: Yazar tarafından derlenmiştir.

Bir sonraki adımda ise elde ettiğimiz “tamliste” içerisindeki kelimeler herhangi bir tekrar olmaması adına son bir kez daha tekrar eden olup olmadığı filtre edilir. Bu işlem sonrasında ise elde edilen liste içerisinde herhangi bir insan isimi olup olmadığının elemesinin yapılabilmesi için Japonca dil işleme kütüphanelerinden biri olan Sudachi vasıtası ile post-tag aramasına girilir. Sudachi, WAP Tokushima Yapay Zekâ ve Doğal Dil İşleme Laboratuvarı (WAP Tokushima Laboratory of AI and NLP - Japan) tarafından geliştirilmiş ve Python uygulamaları için “https://pypi.org/project/SudachiPy/0.2.1/” adresinden indirilebilen bir kütüphaneye sahiptir. Çalışmada bu kütüphaneden faydalanılarak, özellikle derlem tabanlı kaynaklardan gelen bilgilerde sıklıkla rastlanılan insan isimleri aşağıdaki şekilde filtre edilmiştir.

Şekil 25: Model_v.1 - Sudachi Japonca Doğal Dil İşleme Kütüphanesi



In [294]:
```
# Tekrar kelimeler var mı son olarak kontrol edilir.
sonuclar=[]
for i in tamliste:
    if i not in sonuclar:
        sonuclar.append(i.replace('[', '').replace(']', ''))
print (sonuclar)
print("Liste uzunluğu:",len(sonuclar))
```

```
['言語学', '計算言語学', '記述言語学', '方言地理学', '語源学', '歴史言語学', '神経言語学', '語用論', '規範言語学', '意味論', '社会言語学', '構造主義', '共時言語学', '方言学', '語彙学', '機械翻訳', '文法', '形態音学', '音韻論', '語彙統計学', '認知意味論', '直語', '形式的意味論', '語彙的意味論', 'トポニミー', '記述文法', '形態学', '規範文法', '構文', 'オルソピー', '複合形態学', '派生形態学', '屈折形態学', '生成文法', '比較言語学', '哲学的言語', '中世ラテン語', '神山孝夫', '発話行為', '統語論', '西村義樹', '語彙論', 'ハーバーマス', 'ポスト構造主義', '佐藤亮一_(言語学者)', '日野資純', 'SYSTRAN', '語彙', '音韻', '高見健一', '山梨正明', '寺村秀夫', '発生学', '分類学', '予約語', 'シンタックス', '仁田義雄', '応用言語学', '英語学', '事大主義', '数理', '民族誌', '言語類型論', '実証主義', '記号論', '国語学', '日本語学', '国際化と地域化', '形態素解析', '形態論', '解釈学', '中国の歴史', '記法', '正規表現', '文型', '語順', '語法', '統語', '単語', '国文法', '文語文', '語い', '複文', 'ＳＶＯＣ', 'パーザ', '形態素', 'EBNF', '統辞']
Liste uzunluğu: 88
```

Japonca insan isimlerinin Sudachi Doğal Dil İşleme kütüphanesi ile filitrelenmesi

In [295]:
```
from sudachipy import tokenizer, dictionary

tokenizer_obj = dictionary.Dictionary().create()
mode = tokenizer.Tokenizer.SplitMode.A
ja = []
for i in sonuclar:
    tokenjp = [m.part_of_speech() for m in tokenizer_obj.tokenize(i, mode)]
    if tokenjp[0][2] != '人名' :
        ja.append(i)
print(ja)
print("\nJaponca kelime listesindeki insan isimleri filtrelendikten sonra liste uzunluğu:", len(ja))
```

```
['言語学', '計算言語学', '記述言語学', '方言地理学', '語源学', '歴史言語学', '神経言語学', '語用論', '規範言語学', '意味論', '社会言語学', '構造主義', '共時言語学', '方言学', '語彙学', '機械翻訳', '文法', '形態音学', '音韻論', '語彙統計学', '認知意味論', '直語', '形式的意味論', '語彙的意味論', 'トポニミー', '記述文法', '形態学', '規範文法', '構文', 'オルソピー', '複合形態学', '派生形態学', '屈折形態学', '生成文法', '比較言語学', '哲学的言語', '中世ラテン語', '発話行為', '統語論', '語彙論', 'ポスト構造主義', '日野資純', 'SYSTRAN', '語彙', '音韻', '発生学', '分類学', '予約語', 'シンタックス', '応用言語学', '英語学', '事大主義', '数理', '民族誌', '言語類型論', '実証主義', '記号論', '国語学', '日本語学', '国際化と地域化', '形態素解析', '形態論', '解釈学', '中国の歴史', '記法', '正規表現', '文型', '語順', '語法', '統語', '単語', '国文法', '文語文', '語い', '複文', 'ＳＶＯＣ', 'パーザ', '形態素', 'EBNF', '統辞']

Japonca kelime listesindeki insan isimleri filtrelendikten sonra liste uzunluğu: 80
```

Kaynak: Yazar tarafından derlenmiştir.

Görüldüğü üzere Japonca insan isimleri filtre edildikten sonra liste uzunluğumuz 80 kelimeye inmiştir. Bir sonraki adımda ise, elde edilen bu listedeki Japonca kelimelerin Latin harfleri ile okunuşlarının yanı sıra Türkçe ve İngilizce

çevirilerinin de elde edilip tüm bu elde edilen kelime çıktısının düzenli bir şekilde birleştirilerek program çıktısı ve ayrıca dosya kaydı olarak bilgisayarımıza aktarılması işlemi kalmıştır. Bunun için öncelikle Japonca kelime okunuşlarını elde edebilmek için, pykakasi kütüphanesi "https://pypi.org/project/pykakasi/" adresinden indirilip uygulamaya çalışmanın başında Model_v.1'e dahil edilmiştir. Bu adımda bu modülden faydalanılarak Japonca kelimelerin Latince okunuşları Hepburn sistemine göre elde edilir ve bu değerler de "romaji" adlı bir liste içerisine atanır. Daha sonra Japonca kelime listesi Türkçeye ve İngilizceye çevrilir ve bu değerler de tekrar liste haline getirilir.

Şekil 26: Model_v.1 – Romaji Japonca okunuşlarının eklenmesi

```
japonca = []
for i in ja:
    if i not in japonca and i != '':
        japonca.append(i)
sontext = ",".join(japonca)

en_text = cevirmen.translate(sontext, dest = 'en').text
ingilizce = list(en_text.split(","))

tr_text = cevirmen.translate(sontext, dest = 'tr').text
turkce = list(tr_text.split(","))

######### Romaji Ekleme ############
import pykakasi

kks = pykakasi.kakasi()
result = kks.convert(sontext)

ja_okunus = " "
for item in result:
    k = item['hepburn'].lower()
    ja_okunus += k
#print(ja_okunus)
romaji = list(ja_okunus.split(","))

###################################
print(romaji)
print("\nRomaji Uzunluk:", len(romaji))
print("\n")

print(japonca)
print("\nJaponca Uzunluk:", len(japonca))
print("\n")

print(turkce)
print("\nTürkçe Uzunluk:", len(turkce))
print("\n")

print(ingilizce)
print("\nİngilizce Uzunluk:", len(ingilizce))
print("\n")
```

Kaynak: Yazar tarafından derlenmiştir.

En son aşamada ise tüm bu listeler ve uzunlukları kontrol amaçlı ekran çıktısı olarak elde edildikten sonra, Türkçe, Japonca, Romaji ve İngilizce listeleri zip fonksiyonu yardımıyla belirlediğimiz formata göre ziplenerek Model_v.1'in çıktısı somut olarak elde edilmiş olur. Aşağıda bu işlem ile Model_v.1'in çıktısı kısmi olarak verilmiştir.

Şekil 27: Model_v.1'in son aşaması ve kısmi çıktı

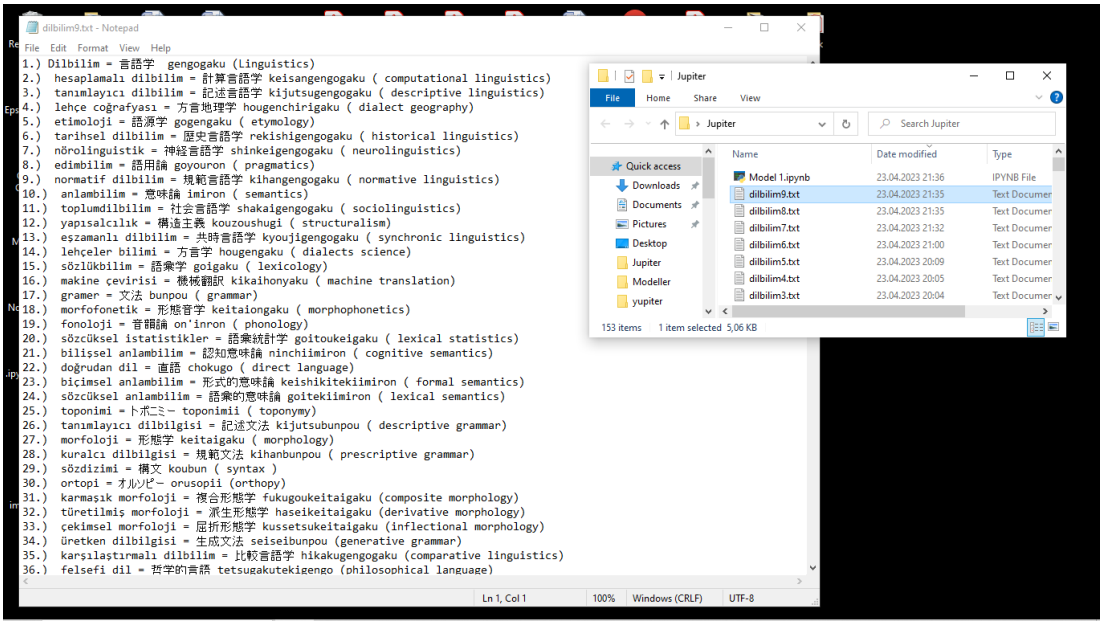
```
In [300]: import sys
          import os

          sayac = 1
          dosyadi = keytr+"{}.txt"
          while os.path.isfile(dosyadi.format(sayac)):
              sayac += 1
          dosyadi = dosyadi.format(sayac)


          sozluk =[]
          sozluk = zip(turkce, japonca, romaji, ingilizce)
          j=0


          for turkce, japonca, romaji, ingilizce in sozluk:
              j +=1
              with open(dosyadi, 'a', encoding='utf-8') as f:
                  print ("{}.) {} = {} {} ({})".format(j, turkce, japonca, romaji, ingilizce) , file=f)
              print ("{}.) {} = {} {} ({})".format(j, turkce, japonca, romaji, ingilizce))
```

```
1.) Dilbilim = 言語学  gengogaku (Linguistics)
2.)  hesaplamalı dilbilim = 計算言語学 keisangengogaku ( computational linguistics)
3.)  tanımlayıcı dilbilim = 記述言語学 kijutsugengogaku ( descriptive linguistics)
4.)  lehçe coğrafyası = 方言地理学 hougenchirigaku ( dialect geography)
5.)  etimoloji = 語源学 gogengaku ( etymology)
6.)  tarihsel dilbilim = 歴史言語学 rekishigengogaku ( historical linguistics)
7.)  nörolinguistik = 神経言語学 shinkeigengogaku ( neurolinguistics)
8.)  edimbilim = 語用論 goyouron ( pragmatics)
9.)  normatif dilbilim = 規範言語学 kihangengogaku ( normative linguistics)
10.)  anlambilim = 意味論 imiron ( semantics)
11.)  toplumdilbilim = 社会言語学 shakaigengogaku ( sociolinguistics)
12.)  yapısalcılık = 構造主義 kouzoushugi ( structuralism)
13.)  eşzamanlı dilbilim = 共時言語学 kyoujigengogaku ( synchronic linguistics)
14.)  lehçeler bilimi = 方言学 hougengaku ( dialects science)
15.)  sözlükbilim = 語彙学 goigaku ( lexicology)
16.)  makine çevirisi = 機械翻訳 kikaihonyaku ( machine translation)
```

Kaynak: Yazar tarafından derlenmiştir.

Yukarıdan da anlaşılacağı üzere, kullanıcıdan alınan anahtar kelime aynı zamanda ".txt" uzantılı olarak kaydedilen çıktı dosyasının adı ve {rakam} olacak şekilde tasarlanmıştır. Böylece eğer dizinde önceden aynı isimde bir dosya kaydedilmiş ise bu durumda bir üst numara dosya adının sonuna eklenir.

Şekil 28: Model_v.1'in bilgisayar dosya çıktısı



Kaynak: Yazar tarafından derlenmiştir.

3.4. MODEL_V.1 DEĞERLENDİRMESİ ve EDİNİLEN BULGULAR

Model_v.1'de karşımıza çıkan zayıf yönlerden biri çeviri kısıtlamasıdır. Bu modelde google translate modülünün ücretsiz olan versiyonu kullanıldığından bir liste halindeki eleman sayısı olarak söylenecek olursa bir limit söz konusu olduğu görülmüştür. Bu durum aslen karakter sayısı olarak hesaplansa da, çalışmada Wordnet'ten dönen İngilizce kelimelerin liste sayısı maksimum 1226 indis uzunluğuna ulaştığında hata döndürmüştür. Ancak, bu rakam da sadece İngilizce için geçerli olup, İngilizceden ›››› diğer diller şeklinde olduğunda daha da aşağılara düşmüştür. Örneğin Model_v.1'de tasarlandığı gibi Japoncadan ›››› Türkçeye ya da Japoncadan ›››› İngilizceye şeklinde çeviri söz konusu olduğunda bu rakam sadece 320 indis boyutu ile sınırlı kalmaktadır. Bu dolaylı da olsa ücretsiz translate hizmetinden faydalanıldığından Model_v.1 in kısıtlılıklarından birini oluşturmaktadır. Örneğin, Wordnet'ten "canlılar" şeklinde bir anahtar sözcük girerek denendiğimizde alt kavramların sayısı çok fazla olacaktır. Çalışmada bu şekilde denemeler yapılarak Japoncadan ›››› Türkçeye 350 boyutunda denendiğinde çeviri yapılmadan sonuç döndürmüştür. Bu durum Japoncayı İngilizceye çevirmeye kalktığımızda da aynıdır.

Bu sebeple çalışmada herhangi bir problemle karşılaşılmaması için, hazırlanan "altkavram" fonksiyonunun içerisinde "limit" adında bir sınırlayıcı değişken tanımlanarak WordNet'ten dönen kelime sayısı kontrol altına alınmış ve bu limit 100 olarak belirlenmiştir. Aslında bu rakam normalde elde etmek istenilen alana yönelik kelime listesi sayısı açısından değerlendirildiğinde ideal olduğu da söylenebilir. Oluşturduğumuz modelde çeşitli durum senaryoları değerlendirilmiş ve örneğin WordNet'ten dönen kelime sayısı 500 olarak belirlendiği durumda esas elde etmek istenilen alan kelimelerinden oldukça uzak yerlere varıldığı tespit edilmiştir. Ayrıca, diğer bir yandan yüksek sayıda bir kelime içeriğine sahip teknik alan sözlüğü oluşturmak aslında çok da istenilen bir durum değildir. Bunun sebebi, normal şartlarda ortalama bir çevirmenin aklında tutabileceği kelime sayısında da sınırlılıkların bulunmasından kaynaklanır. Bu sebeple Model_v.1'de bu limit 100 olarak belirlenmesi uygun görülmüştür. Ancak fonksiyonun çalışma prensibi gereği 100 limiti altında da, örneğin alt döngü öncesi kelime sayısı 99 ise bir alt kümeden veri çekmeye devam edebilmektedir. Böylece bazı durumlarda girilen input

kelimesine bağlı olarak WordNet'ten dönen kelime sayısı 100 kelimenin üstüne de çıkabilmektedir. Fakat bu durum modelde tolere edilmiştir.

Model_v.1'in güçlü yanlarından biri ise, WordNet'ten gelen ilişkili kelimelerin 3 farklı Word2Vec ön eğitimli derlemi üzerinden yeni ilişkili kelimeleri çekebilmesidir. Bunun için modelde WordNet'ten dönen kelimelerin öncelikle hangisinin o ön eğitimli derlem modeli içerisinde bulunduğu tespit edilmiş, daha sonrasında bu kelimelerin ilişkili olduğu yakın kelimeler, derlemde tespit edilen kelimelerin Word2Vec modeli içerisinde tespit edilme frekansına göre bir "tps" değeri "if-else" koşulu ile kontrol edilerek belirlenmiştir. Böylece eğer tespit edilen kelime sayısı çok ise derlemden dönecek ilişkili kelime sayısı minimal, eğer az ise maksimal bir şekilde döndürülerek, toplam kelime sayısının tüm model çıktısı üzerinde homojen bir şekilde dağılımı önemsenerek dikkate alınmıştır.

Model_v.1'in güçlü yanlarından bir diğeri ise, özellikle Japonca olarak dönen ilişkili kelimeler sadece ileri seviyede Japonca bilenlerin okuyabileceği şekilde bırakılmayıp, "py.kakashi" modülü yardımıyla Japonca olan bu kelimelerin Latince harflerle okunuşunu ifade eden "romaji" okunuşları da ilave edilmiştir. Böylece, her seviyedeki çevirmen ve eğitim sürecinde olan Japonca öğrencilerinin de Model_v.1'den faydalanabilmesi mümkün hale gelmiştir.

Model_v.1'in diğer güçlü yönlerinden biri ise, Sudachi kütüphanesi kullanılarak elde edilen ilişkili Japonca kelimelerin içerisine derlemden kaynaklanan bir takım kişi isimlerini filtre edebilmesidir. Örneğin, "dilbilim" anahtar sözcüğü girildiğinde filtreleme öncesinde elde edilen Japonca listesindeki toplam eleman sayısı 88 iken, Sudachi yardımıyla uyguladığımız kişi adları post-tagı[21] sayesinde filtre edilerek bu rakam 80'e düşmüştür.

Model_v.1'in güçlü yönlerinden bir başkası ise, elde edilen sonuçları ekrana bastırmasının yanı sıra, kullanıcıdan alınan anahtar sözcüğü dosya adı yaparak ".txt" uzantılı dosya olarak somut olarak bilgisayara çıktı olarak kaydedebilmesidir. Ayrıca hazırlanan kod sayesinde aynı girdi kelimesi iki kez kullanılsa dahi Model_v.1 öncelikle bu isimde bir dosyanın çalışma dizininde mevcut olup olmadığına bakmakta ve eğer var ise dosya sonu numarasını 1 arttırarak dosyayı sonu numara ile

[21] Sudachi post tagları sayesinde oldukça büyük bir oranda kişi adları filtrelenebilmiştir. Ancak yine de bazı durumlarda Sudachi kaynaklı olarak gözden kaçan kişi adlarının bulunabildiği tespit edilmiştir.

bitecek şekilde kaydetmektedir. Örneğin yapılan bu çalışmada anahtar sözcük "dilbilim" şeklinde girildiğinde, o anahtar sözcük eğer modele ilk kez verilmiş ise model ilk kez çalıştırıldığında sayaç başlangıç değerimiz 1 olarak verildiğinden "dilbilim1.txt" şeklinde olmuştur. Model_v.1'in bu şekilde tasarlanmasının sebebi, hem alan sözlüklerini ileride bir veri tabanı halinde kullanabilme, hem de eğer aynı anahtar sözcük olsa da farklı ayarlar ve derlemler kullanılmış olabileceği durum senaryolarının da düşünülmüş olmasından kaynaklanmaktadır. Böylece istenildiği zaman öncelikle bu kayıtlar kontrol edilerek zamandan tasarruf sağlanabilecektir.

DÖRDÜNCÜ BÖLÜM
KELİME DESTEK MODÜLÜ ÇIKTILARININ TEST EDİLMESİ

Bu bölümde bir önceki bölümde geliştirilen kelime destek modülü çıktıları test edilmiştir. Bu amaca uygun olarak modeli her ne kadar farklı yöntemlerle test ederek değerlendirmek mümkün olsa da, öncelikle modelin ne derece etkili olabildiği sorununun yanıtlanması gerekmektedir. Modelin *etkililik (effectiveness)* konusunda ne derece başarılı olabildiğinin ölçümünün yapılabilmesi için yine Python kodlarından faydalanılmıştır. İstatistiksel yöntemlere dayalı sonuçlar üretebilen bir ölçüm modülü geliştirilmiştir. Daha sonra Model_v.1'nin çalıştırılmasının ilk adımında kullanıcıdan istenen anahtar kelime ile aynı alanda bulunan Japonca akademik makaleler internet üzerinden indirilmiştir. Sonrasında ise, Model_v.1 modülü çıktıları farklı yazarlar tarafından hazırlanan ve PDF formatında bulunan bu yazınlar üzerinden değerlendirilmiştir. Böylece Model_v.1 çıktılarının, hem Japonca akademik yazınlar üzerinden etkililik testi gerçekleştirilerek başarı ölçümlendirilmesi yapılmış, hem de elde edilen bulgular neticesinde model optimizasyonu için gereken bilgilere de ulaşılmıştır.

Değerlendirme modülü 2 farklı versiyon olarak hazırlanmıştır. Bu modüller "Evaluator_P" ve "Evaluator_nP" olarak adlandırılmıştır. Çalışma prensipleri aynı olsa da, bu iki modül arasındaki en temel fark Japoncanın yazı sisteminden kaynaklanmaktadır. Japoncada, diğer dillerin pek çoğunda görüldüğü gibi kelimeler arasında herhangi bir boşluk söz konusu değildir. Bu durum özellikle Japoncanın diğer diller gibi işlenmesi hususunda oldukça büyük bir sorun teşkil etmekte olduğundan bu soruna çözüm olarak pek çok Japonca doğal dil işleme kütüphanesi oluşturulmuştur. Çalışmada, bir önceki bölümde bahsi geçen Sudachi kütüphanesi bu amaçla kullanılmıştır. Japonca dil işlemede mevcuttaki sorunu en basit hali ile açıklamak gerekirse, örneğin "devlet üniversitesi araştırmacısı (national university researcher)" şeklinde bir kavram Japoncada "国立大学研究員 (kokuritsudaigakukenkyuuin)" şeklinde boşluksuz olarak yazılmakta ve görüldüğü üzere birbirine bitişik Japonca Kanji imlerinin bir biri ardına gelmesinden oluşmaktadır. Ancak Japoncadaki bu kavramı işleyebilmek için farklı şekillerde, örneğin "国立大学(kokuritsudaigaku)" "研究員(kenkyuuin)" şeklinde "ulusal-üniversite" ve "araştırmacı" olarak ikiye bölmek ya da "国立(kokuritsu)" , "大学(daigaku)" , "研究員(kenkyuuin)" şeklinde "ulusal" ,

"üniversite", "araştırmacı" olarak üçe bölmek ve yahut "国立(kokuritsu)" , "大学(daigaku)" , "研究(kenkyuu) , "員(in)" şeklinde "ulusal" , "üniversite", "araştırma", "mensubu" olarak dörde bölmek dahi mümkündür. Aşağıda verilen bu örneğin Sudachi kütüphanesi çıktısı görülmektedir.

Şekil 29: Sudachi kütüphanesi bölümleme modları



Kaynak: Yazar tarafından derlenmiştir.

Hazırlanan değerlendirme modülleri olan "Evaluator_P" ve "Evaluator_nP" nin her ikisinde de Sudachi kütüphanesinin bu 3 farklı modundan faydalanılmıştır. Ancak "Evaluator_P" ve "Evaluator_nP" olarak adlandırılan bu iki modül arasındaki en temel fark bu bölümleme modlarının hangi aşamada modülde uygulanıp uygulanmadığıdır. Örneğin, "Evaluator_P" olarak adlandırılan modülde, Model_v.1'den elde edilen Japonca kelime listesine uygulanmayarak tüm kelimeler olduğu şekli tamlama halinde Japonca makalelerde taratılacak şekilde bir tasarım yapılmıştır. Bu sebeple "Evaluator_P" modülü daha "kuralcı (prescriptive)" bir yapıya sahiptir ve tam eşleşme olmadığı sürece olumlu dönüş vermemektedir. Hazırlanan ikinci modül olan "Evaluator_nP" ise ilkine göre daha az kuralcı ya da "kuralcı olmayan (non-prescriptive)" şeklinde nitelendirilebilecek bir yapıya sahiptir ve örneğin "makine çevirisi" kavramından "makine" ve "çeviri" kavramlarını sentezleyebilen bir çevirmenin var olduğunu hesaba katarak tasarlanmıştır. Dolayısıyla, değerlendirme modüllerinin her ikisinde de Japonca makalelerin çözümlenmesi aşamasında standart olarak Sudachi kütüphanesinin tüm modları uygulanacak şekilde bir tasarım yapılsa da, Sudachi'nin bu bölümleme modların Model_v.1'den gelen kelime listesine uygulanıp uygulanmayacağı hususu modüle göre farklı tutulmuştur. Böylece Model_v.1'in çıktılarını, aynı makale üzerinde hem "kuralcı (prescriptive)" bir yaklaşıma bağlı kalan,

hem de "kuralcı olmayan (non-prescriptive)" yaklaşıma da imkan tanıyan, iki farklı yönteme göre analiz etme imkanı doğmuştur.

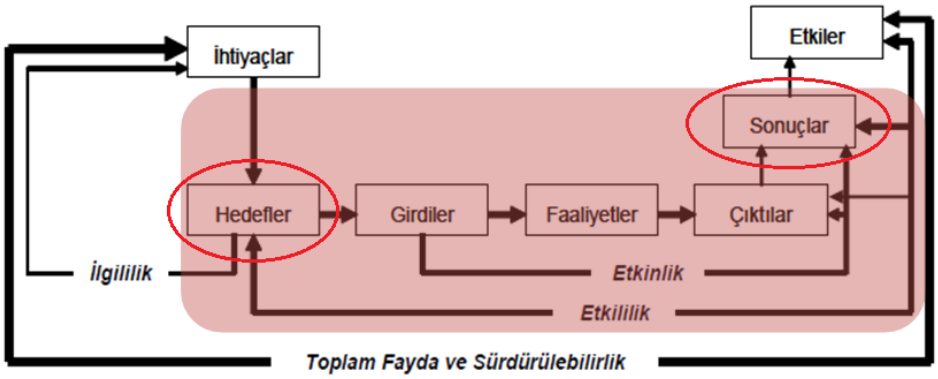
4.1. ETKİLİLİK DEĞERLENDİRMESİ

Bir proje ya da modelin değerlendirilmesindeki kriterlere ilişkin, elde edilmek istenilen sonuçlara bağlı olarak çeşitli kriterlerden söz etmek mümkündür. Proje alanı farklılık gösterse de, genelde bir proje ya da modelin değerlendirilmesinde standart olarak belirlenebilen ve çoğu projede de ortak olarak karşımıza çıkan bir takım değerlendirme kriterlerinin mevcut olduğu görülür. 2006 yılında, Avrupa Birliği tarafından yayınlanan değerlendirme raporunda da benzer şekilde aşağıdaki değerlendirme kriterlerinin verildiği görülür (European Communities, 2006: 49-52).

- *İlgililik (relevance)*
- *Etkililik (effectiveness)*
- *Etkinlik (efficiency)*
- *Sürdürülebilirlik (sustainability)*
- *Etki (impact)*
- *Tutarlılık / tamamlayıcılık (coherence / complementarity)*
- *Topluluk katma değeri (community value added)*

Yine, Sarı (2010: 19-21)'de de bir proje ya da modelin değerlendirilmesinde "İlgililik (relevance)" , "Etkinlik (efficiency)" , "Etkililik (effectiveness)" , "Etki (impact)" ve "Sürdürülebilirlik (sustainability)" şeklinde verilen 5 temel kriterden söz edilmektedir. Ayrıca, Sarı (2010)'da değerlendirme kriterleri ile sonuçlarının ilişkisi hususunda aşağıdaki şemanın verildiği görülmektedir.

Şekil 30: Değerlendirme ile sonuç ilişkisi



Kaynak: Sarı (2010: 19)'da görülen şemanın yazar tarafından modifiye edilmiş hali.

Çalışmanın bu bölümünde bu değerlendirme kriterlerinden biri olan ve modelin başlangıç hedeflerinin, modelin çalıştırılması neticesinde elde ettiği sonuçlara ne derece uygun ve yakın olduğunu ortaya koyabileceği düşünülen "*Etkililik (effectiveness)*" kriteri üzerinde durulacaktır. Bunun sebebi ise, yukarıda bahsi geçen kriterlerden özellikle "*Etkililik (effectiveness)*" kriterinin, bir proje ya da modelin çıktı ve sonuçlarının, o projenin özel amaç ve hedeflerine olan katkı düzeyini sorgulayabilmesinden kaynaklanır (Sarı, 2010: 21).

Model_v.1'in etkililik açısından değerlendirilmesi kapsamında, aşağıdaki 3 soruya cevap aranacaktır.

1) Çevirmenler için teknik alana yönelik olarak hazırlanan Türkçe - Japonca kelime destek modülü olan Model_v.1'in ürettiği terimler o alandaki Japonca yazınları okurken kelime bilgisi açısından destek sağlayabilir mi?

2) Model_v.1'in, ürettiği terimlerin o alanda yazılmış Japonca metinler üzerinde ölçümlendirilebilir bir uygunluk derecesi mevcut mu?

3) Model_v.1 modülü kelime çıktıları Japonca akademik yazınlar üzerinde tespit edilebilir ölçüde bir katkı sağladı mı?

Çalışmanın geri kalanında temel olarak bu 3 soruya cevap aranarak Model_v.1'in etkililik açısından değerlendirilmesi yapılacaktır. Ancak öncelikle burada 2. ve 3. soruya yanıt verebilmek amacıyla aşağıdaki şekilde bir hesaplama yöntemi geliştirilmiştir. Bu iki soruya verilecek yanıtlar aynı zamanda 1. sorunun cevabına zemin hazırlayacaktır.

<u>İkinci soru için geliştirilen uygunluk formülü:</u> Burada öncelikle uygunluk kavramı üzerinde açıklama yapmak gerekmektedir. Bu durum bir işletmenin ürettiği ürünlerden hangilerinin müşteriler için daha uygun ve işletme açısından da daha satılabilir olduğu sorusuna yanıt verilerek açıklanabilir. Örneğin, bir işletmenin üretmiş olduğu 10 farklı tür ve özellikteki üründen diyelim ki sadece hep aynı 4 ürünün müşteriler tarafından tercih edildiğini, diğer 6'sını ise raflarda kalarak hiç

satılmadan iade edildiğini düşünelim. Bu durumda işletme açısından bu 4 ürüne odaklanılacak ve ürün türleri arasından sadece bu 4 ürünün müşteri ihtiyaçlarına göre daha uygun olduğu sonucuna varılacaktır. Tıpkı bu örnekte olduğu gibi, Model_v.1 kelime çıktı türlerinin belirli bir metin içerisinde görülme (hit) sayısı da bizlere o kelimelerin uygunluk düzeyini verecektir.

Dolayısıyla, Model_v.1'in ürettiği terimlerin ne ölçüde uygun olduğunu, kullanıldığı yere diğer bir deyişle analizi yapıldığı metne bağlı olarak, Model_v.1 kelime çıktılarının o metin içerisinde tekrarsız görülme (tekrarsız hit = tür) sayısının, Model_v.1 kelime çıktılarının toplamına olan oran bizlere uygunluk derecesini verecektir. Bu oranın yüzdesel ifadesi ise aşağıdaki şekilde formüle edilebilir.

Şekil 31: Terimsel Uygunluk

$$\text{Terimsel Uygunluk Puanı} = \frac{\text{Model_v.1 kelimeleri tekrarsız hit}}{\text{Model_v.1 kelime listesi toplam sayısı}}$$

$$\text{Terimsel Uygunluk (\%TU)} = \frac{\text{Model_v.1 kelimeleri tekrarsız hit}}{\text{Model_v.1 kelime listesi toplam sayısı}} \times 100$$

Kaynak: Yazar tarafından hazırlanmıştır.

Üçüncü soru için kullanılan formül: Model_v.1 modülü kelime çıktıları Japonca akademik yazınlar üzerinde ne ölçüde katkı sağladığını ölçmede, bir dokümandaki terim sayısını hesaplamada kullanılan ve Terim Frekansı (TF) olarak bilinen formülden yararlanılmıştır. Ancak bu formül tek bir terim olarak değil, aşağıdaki şekilde doküman genelinde Model_v.1 tarafından üretilen terimlerin tamamına uyarlanmış, böylece Model_v.1 kelimelerinin tamamı hedef terim olarak ele alınarak katkı oranı cinsinden hesaplanması düşünülmüştür.

Şekil 32: Katkı Oranı

$$\text{Katkı Oranı (Modül-TF\%)} = \frac{\text{Tekrarlı hit sayısı}}{\text{Dokümandaki tekrarlı kelime sayısı}} \times 100$$

Kaynak: Standart TF formülünün modifiye edilerek uyarlanması.

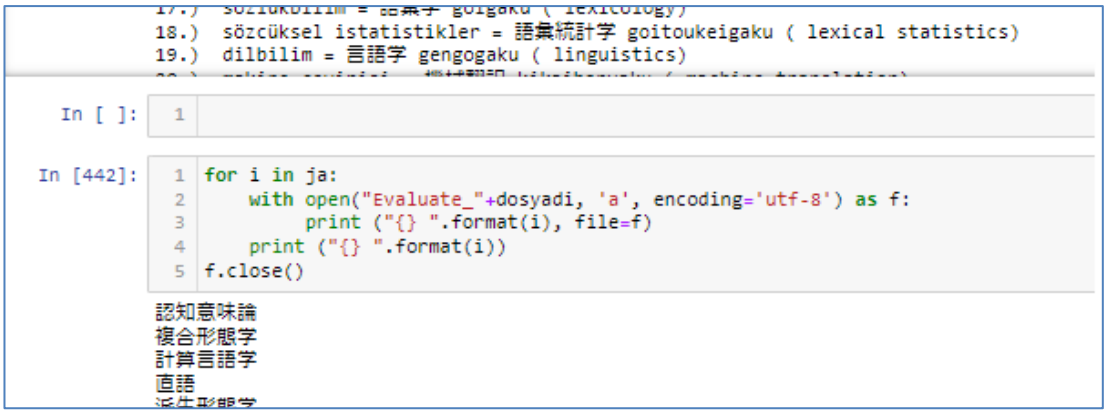
Bu aşamadan sonra aşağıda Model_v.1'in etkililik açısından değerlendirme adımına geçilmiştir. Ancak bunun öncesinde unutulmaması gereken, bir metindeki kelimelerin tamamının terim olmasının mümkün olmadığıdır. Herhangi bir metindeki mevcut terim sayısı, hem alana hem de yazarın vermek istediği bilgiye göre değişiklik gösterdiğinden her ne kadar tam olarak belirgin rakamlardan söz etmek çok mümkün olmasa da, bir fikir vermesi açısından, Rinaldi ve Yuste'nin (2004) yılında belirli bir teknik alana ait bir derlem üzerinde yapmış oldukları çalışmaya göre yaklaşık %30 civarı olarak tespit edildiği bilinmektedir. Dolayısıyla Model_v.1'in ürettiği kelime listesi terimlerinin de her hangi bir metin içerisinde geçme yüzdelerinin belirli bir seviyenin altında olması beklenen doğal bir durumdur.

Diğer bir yandan bir sonraki adımda modelin Japonca yazınlar üzerindeki etkililik ölçümünün yapılabilmesi için gereken analiz verileri daha önce de bahsedildiği üzere, "kuralcı" ve "kuralcı olmayan" şeklinde iki tipte hazırlanan değerlendirme modülü kodları vasıtasıyla oluşturulduğundan, aşağıda bu değerlendirme türleri farklı iki başlık altında ele alınmıştır.

4.2. EVALUATOR_P MODÜLÜ İLE ETKİLİLİK DEĞERLENDİRMESİ

Model_v.1'in bir önceki III. Bölümde elde edilen çıktılardan Japonca kelimeler öncelikle hangi terim listesinin adı olduğunu ayırt edebilmemiz için daha önce Model_v.1'in çıktılarına ad verme ile benzer olacak şekilde ancak aynı olmaması adına başına "Evaluate_" ekleyerek bilgisayara kaydedilmesi sağlanmıştır.

Şekil 33: "Evaluate_" başlıklı dosya oluşumu



Kaynak: Yazar tarafından derlenmiştir.

Evaluator_P test modülü oluşumu için öncelikle “pdftotext” kütüphanesine ihtiyaç duyulduğundan bu kütüphane “pip install” komutu ile yüklenmiştir. Bunların haricinde de gerekli olan kütüphaneler kod görüntüsü içerisinde mevcut olduğundan burada tek tek açıklamaya gerek görülmemiştir. Daha sonra test verilerimizi belirli bir klasörde tutmak amacıyla hazırlanan aşağıdaki kod vasıtasıyla klasör ve dosya oluşturumu yapılmıştır. Analizimizde kullanabilmek amacıyla “.csv” uzantılı dosya da yine bu klasörün içerisinde yaratılmıştır. Ayrıca zaman etiketinden “csv” dosya adı oluşumunda faydalanılmıştır.

Şekil 34: Klasör ve Dosya Oluşumu - 1

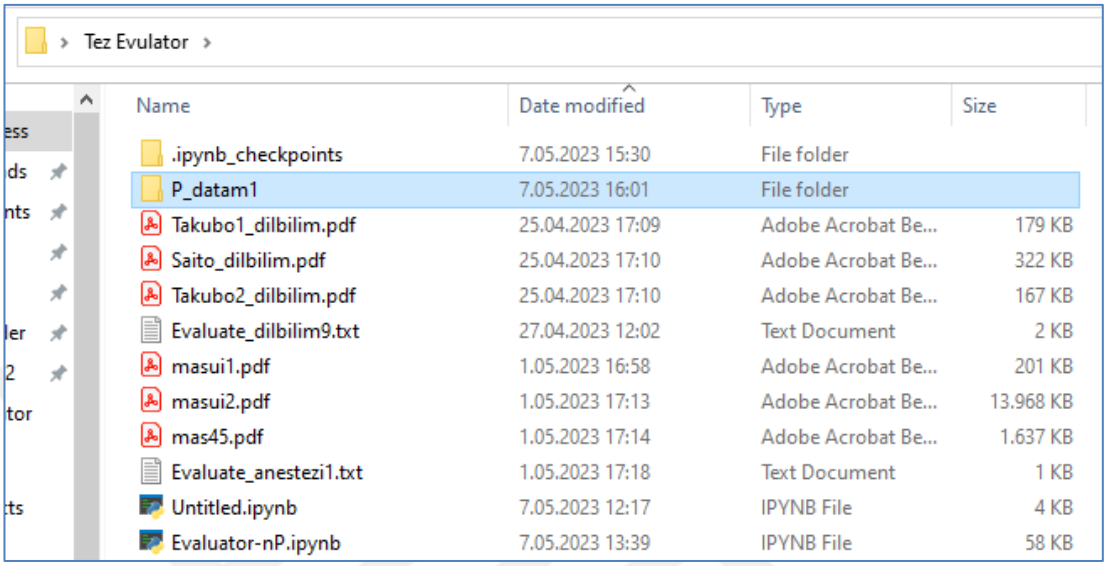
```
import csv
import sys
import os
from datetime import datetime

dater = datetime.now().strftime("_%d-%m-%Y (Saat %H.%M.%S)")
tar = dater

say = 1

data = "P_datam{}"
while os.path.isdir(data.format(say)):
    say += 1
os.mkdir(data.format(say))
datap = (data.format(say))

fp ='C:/Users/koluk/Desktop/Tez Evulator/{}/'.format(datap)
print(fp)

data1 = ((fp+"P_data[{}] "+tar+".csv").format(say))

with open(data1, mode='w', encoding='utf-8') as e_file:
    e_writer = csv.writer(e_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)
    e_writer.writerow(['Dokuman_ADI','AnalizDosya','KATKI','TekrarsizHits','TekrarliHits', 'U
e_file.close()

C:/Users/koluk/Desktop/Tez Evulator/P_datam1/
```

Kaynak: Yazar tarafından derlenmiştir.

Yukarıdaki kod çalıştırıldığında, kodda bahsi geçen konuma aşağıdaki şekilde “P_datam1” klasörü ile bu klasör içerisinde “P_data[1] _07-05-2023 (Saat 13.36.04).csv” dosyası oluşturulmuştur. Ayrıca aşağıdaki şekilde de görüldüğü üzere analiz etmek istenilen Japonca dokümanlar ile Model_v.1’in çıktılarının testi için yukarıda ilk adımda bahsedilen “Evaluate_” ile başlayan Model_v.1’in çıktı dosyaları da buraya aktarılmıştır. Burada, III. Bölümde bahsedilen ve Model_v.1 kelime destek modülünün çıktıları olan “dilbilim9.txt” dosyasının yanı sıra, objektif olarak değerlendirebilme yapılabilmesi

amacıyla farklı alanlarda da modül tarafından üretilen kelime destek çıktıları konumlandırılmıştır. Aşağıda oluşturulan "P_datam1" dosyası ile bunların bir kısmı görüntülenmektedir.

Şekil 35: Klasör ve Dosya Oluşumu - 2



Kaynak: Yazar tarafından derlenmiştir.

Daha sonra Model_v.1 çıktı kelimelerini test etmek amacıyla hazırlanan "Evaluator_P" test modülünün çalıştırılma aşamasına geçilmiştir. Bu amaç için hazırlanan "Evaluator_P" nin kodları biraz uzun olduğundan sıra ile bölünerek aşağıda gösterilmiştir.

Şekil 36: Evaluator_P Test Modülü - 1

Evaluator_P Test Modülü

```
import numpy as np
import pdftotext
from sudachipy import tokenizer, dictionary
from datetime import datetime

adate = datetime.now().strftime("%d/%m/%Y Saat: %H:%M:%S")
tarih = adate

deg = input("Lütfen değerlendirilecek 'Model_v.1 Evaluate' dosyasını anahtar sözcüğü ve numarası ile giriniz:\n")
deger =""
deger = "Evaluate_"+deg+'.txt'
f = open(deger, 'r', encoding='utf-8')
satirlar = f.readlines()
ja=[]
for i in satirlar:
    ja.append(i.strip())

f.close()
print("\n",ja)
print("\n{} dosyasının uzunluğu: {}".format(deger, len(ja)))

pdfadi = input("Değerlendirmek istenilen PDF dosya adını uzantısız giriniz:")
with open(pdfadi+".pdf", "rb") as f:
    pdf = pdftotext.PDF(f)

```

Kaynak: Yazar tarafından derlenmiştir.

Şekil 37: Evaluator_P Test Modülü - 2

```
25
26 tokenizer_obj = dictionary.Dictionary().create()
27
28 mode = [tokenizer.Tokenizer.SplitMode.A , tokenizer.Tokenizer.SplitMode.B , tokenizer.Tokenizer.SplitMode.C]
29 for md in range(0,3):
30
31         dosyam = pdfadi+"_P_"+deger+"_AnaMode"+str(md)
32
33         #print(len(pdf))
34         genlist, genkes, genoran, gkesoz, sonuc, gentekrarsizlst = [],[],[],[],[],[]
35         listem1,listem2,listem3 = [],[],[]
36         kes1,kes2,kes3 = [],[],[]
37
38         for i in range(0, len(pdf)):
39
40             x = [n.surface() for n in tokenizer_obj.tokenize(pdf[i], mode[md])]
41
42             tksiz1,tksiz2,tksiz3 = [],[],[]
43
44             liste1 =[]
45             for t in x:
46                 y = [n.part_of_speech() for n in tokenizer_obj.tokenize(t, mode[0])]
47                 if y[0][1] == '普通名詞':
48                     liste1.append(t)
49                     genlist.append(t)
50                     listem1.append(t)
51                     for z in genlist:
52                         if z not in gentekrarsizlst:
53                             gentekrarsizlst.append(z)
54                     for h in liste1:
55                         if h not in tksiz1:
56                             tksiz1.append(h)
57             kes = np.intersect1d(ja, liste1)
58             if len(kes) != 0:
59                 sonuc.append([i,kes])
60                 genkes.append(len(kes))
61                 for s in kes:
62                     kes1.append(s)
63                     if s not in gkesoz:
64                         gkesoz.append(s)
65                 oran1 = ((len(kes)*100)/len(liste1))
66                 genoran.append(oran1)
67                 with open(fp+dosyam+"_subModA.txt", 'a', encoding='utf-8') as f:
68                     print ("{}.sayfada tespit edildi({})\n\n{}\n".format(i+1,tarih,kes.tolist()),file=f)
69                     print ("{}.sayfada tespit edilen Model_v.1 kelime sayısı\t : {}".format(i+1, len(kes)),file=f)
70                     print ("{}.sayfadaki toplam kelime sayısı            \t : {}".format(i+1, len(liste1)), file=f)
71                     print ("{}.sayfa genelinde Model_v.1 kelime tespit oranı\t : %{:.5f}".format(i+1, oran1),file=f)
72                     print ("{}.sayfanın mevcut tekrarsız kelime sayısı\t : {}\n\n".format(i+1, len(tksiz1)),file=f)
73
74                 f.close()
75
```

```
75
76             liste2 = []
77             for t in x:
78                 y = [n.part_of_speech() for n in tokenizer_obj.tokenize(t, mode[1])]
79                 if y[0][1] == '普通名詞':
80                     liste2.append(t)
81                     genlist.append(t)
82                     listem2.append(t)
83                     for z in genlist:
84                         if z not in gentekrarsizlst:
85                             gentekrarsizlst.append(z)
86                     for h in liste2:
87                         if h not in tksiz2:
88                             tksiz2.append(h)
89             kes = np.intersect1d(ja, liste2)
90             if len(kes) != 0:
91                 sonuc.append([i,kes])
92                 genkes.append(len(kes))
93                 for s in kes:
94                     kes2.append(s)
95                     if s not in gkesoz:
96                         gkesoz.append(s)
97                 oran2 = ((len(kes)*100)/len(liste2))
98                 genoran.append(oran2)
99                 with open(fp+dosyam+"_subModB.txt", 'a', encoding='utf-8') as f:
100                    print ("{}.sayfada tespit edildi({})\n\n{}\n".format(i+1,tarih,kes.tolist()),file=f)
101                    print ("{}.sayfada tespit edilen Model_v.1 kelime sayısı\t : {}".format(i+1, len(kes)),file=f)
102                    print ("{}.sayfadaki toplam kelime sayısı            \t : {}".format(i+1, len(liste2)), file=f)
103                    print ("{}.sayfa genelinde Model_v.1 kelime tespit oranı\t : %{:.5f}".format(i+1, oran2),file=f)
104                    print ("{}.sayfanın mevcut tekrarsız kelime sayısı\t : {}\n\n".format(i+1, len(tksiz2)),file=f)
105
106                f.close()
107
```

Kaynak: Yazar tarafından derlenmiştir.

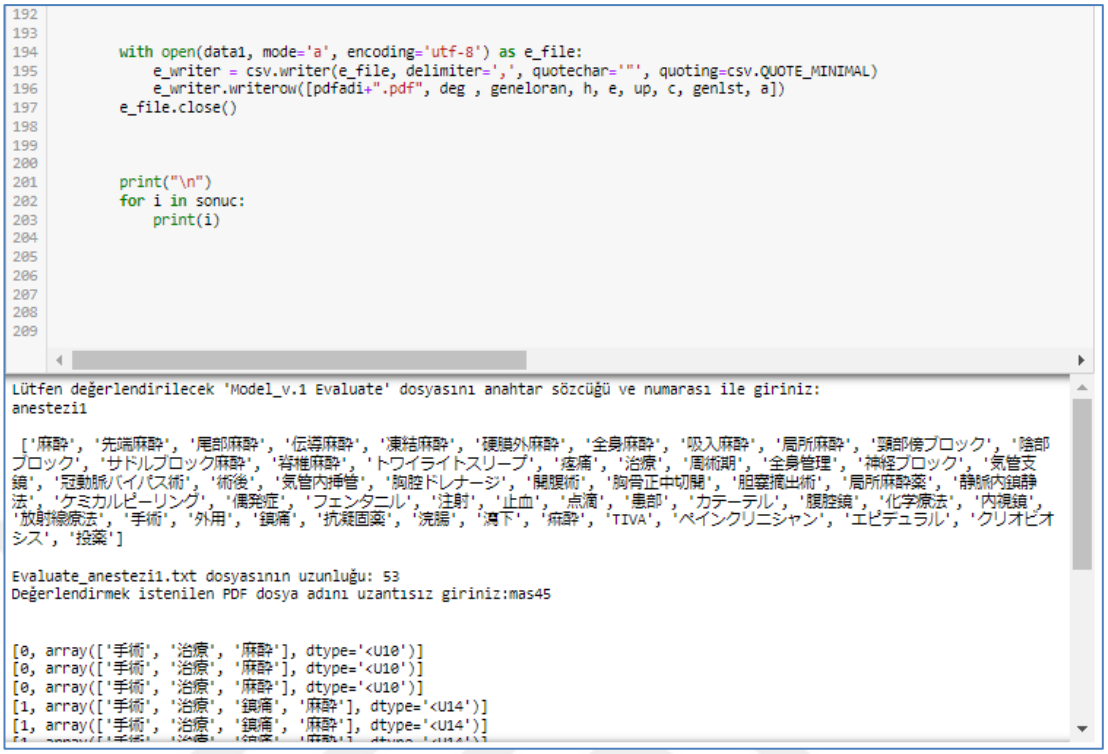
Şekil 38: Evaluator_P Test Modülü - 3

```
            liste3 = []
            for t in x:
                y = [n.part_of_speech() for n in tokenizer_obj.tokenize(t, mode[2])]
                if y[0][1] == '普通名詞':
                    liste3.append(t)
                    genlist.append(t)
                    listem3.append(t)
                    for z in genlist:
                        if z not in gentekrarsizlst:
                            gentekrarsizlst.append(z)
                    for h in liste3:
                        if h not in tksiz3:
                            tksiz3.append(h)
            kes = np.intersect1d(ja, liste3)
            if len(kes) != 0:
                sonuc.append([i,kes])
                genkes.append(len(kes))
                for s in kes:
                    kes3.append(s)
                    if s not in gkesoz:
                        gkesoz.append(s)
                oran3 = ((len(kes)*100)/len(liste3))
                genoran.append(oran3)
                with open(fp+dosyam+"_subModC.txt", 'a', encoding='utf-8') as f:
                    print ("{}.sayfada tespit edildi({})\n\n{}\n".format(i+1,tarih,kes.tolist()),file=f)
                    print ("{}.sayfada tespit edilen Model_v.1 kelime sayısı\t: {}".format(i+1, len(kes)),file=f)
                    print ("{}.sayfadaki toplam kelime sayısı            \t: {}".format(i+1, len(liste3)), file=f)
                    print ("{}.sayfa genelinde Model_v.1 kelime tespit oranı\t: %{:.5f}".format(i+1, oran3),file=f)
                    print ("{}.sayfanın mevcut tekrarsız kelime sayısı\t: {}\n\n".format(i+1, len(tksiz3)),file=f)

                f.close()

        genlst = int(len(genlist)/3)

        genelkes = 0
        for i in genkes:
            genelkes += i

        geneloranım = ((genelkes*100)/len(genlist)) if len(genlist) else 0

        geor = 0
        for i in genoran:
            geor += i
        geneloran = (geor / len(genoran)) if len(genoran) else 0

        genelkelimeoran = ((len(gkesoz)*100)/len(gentekrarsizlst)) if len(gentekrarsizlst) else 0
```

```
        if md == 0: sd = "A"
        if md == 1: sd = "B"
        if md == 2: sd = "C"

        pdfadim ="\""+pdfadi+".pdf"+"\""

        tfa = (len(kes1) / len(listem1)) if len(listem1) else 0
        tfb = (len(kes2) / len(listem2)) if len(listem2) else 0
        tfc = (len(kes3) / len(listem3)) if len(listem3) else 0

        #########  Değer Hesaplama ###########
        a = len(ja)             # model kelime listesi toplam sayısı
        c = len(gentekrarsizlst) # Dokümandaki tekrarsız kelime sayısı
        h = len(gkesoz)          # tekrarsız hit sayısı
        e = int(genelkes/3)      # tekrarlı hit sayısı 3-sub mod var x3 kez yazıldığından 3'e bölünür

        up = (h/a) if a else 0    # Terimsel Uygunluk Puanı  (# Sıfıra bölme hatası 0 olarak güncelleme)

        with open(fp+dosyam + "_GenelDeg.txt", 'a', encoding='utf-8') as f:
            print ("***************** Analiz Tarih: {}  (doküman:{}) *****************\n ".format(tarih,pdfadim), file=f)

            print ("Dokümanda {} ana mod - Model_v.1 modülü tarafından bulunan tekrarsız eşleşme kelime adeti: {}".format(sd
            print ("Dokümanda {} ana mod - Model_v.1 modülü kelimlerinde kaç kez tekrarlı eşleşme görüldüğü  : {}".format(sd

            print ("\nDokümanda Model_v.1 kelimeleri {} ana moduna göre toplam tekrarsız kelime sayısı: {}".format(sd, len(g
            print ("Dokümanda Model_v.1 kelimeleri {} ana moduna göre toplam tekrarlı kelime sayısı : {}".format(sd, genlst)

            print ("\nDoküman genelinde tekrarlı kelime sayılarına ilişkin detaylar => \nSudachi A sub-moduna göre Listem1 =

            print ("Sudachi A sub-moduna göre TF-a = {:.3f} \nSudachi B sub-moduna göre TF-b = {:.3f} \nSudachi C sub-moduna
            print ("\n{} ana mod iken \"{}\" adlı dosya - Model_v.1 Terim Sayısı : {}".format(sd, deger, len(ja)), file=f)
            print ("{} ana mod iken Model_v.01 çıktılarının Terimsel Uygunlu  - Soru2 : %{}".format(sd, up*100), file=f)
            print ("Model_v.1 Katkı Oranı (Modül-TF%) - Soru3  : %{}".format(geneloran), file=f)

            print("\n\n", file=f)

        f.close()
```

Kaynak: Yazar tarafından derlenmiştir.

Şekil 39: Evaluator_P Test Modülü - 4



Kaynak: Yazar tarafından derlenmiştir.

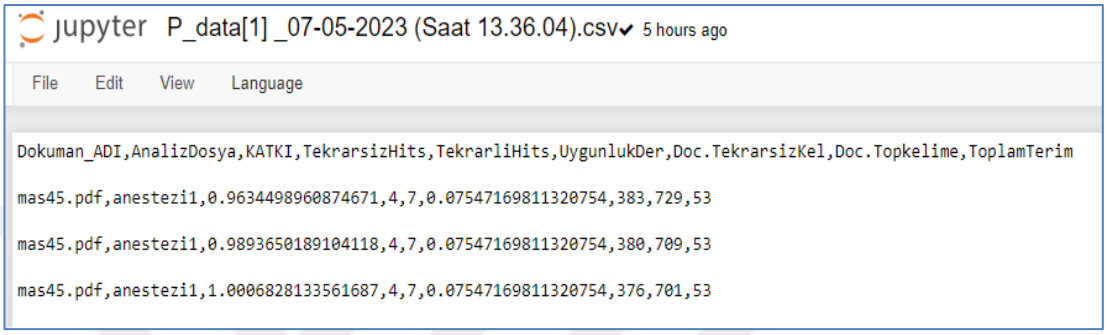
"Evaluator_P" test modülü çalıştırıldığında 3 tipte çıktı vermek üzere tasarlanmıştır. Burada her pdf sayfası bir "for" döngüsünde çözümlenmekte ve Japonca doküman kaç sayfa ise bunların her biri tek tek analiz edilip, sayfa analizleri bir ".txt" dosyası içerisine yazılmaktadır. Devamında bu analiz sonuçları yukarıda bahsettiğimiz "csv" uzantılı dosya içerisine tüm değerler hesaplandıktan sonra dokümanın genel sonuçları olarak aktarılmaktadır. Böylece görsel analiz için veri hazırlanmaktadır. Ayrıca dosyanın çalıştığını takip edebilmek amacıyla, yukarıda görüldüğü şekilde ekran çıktısı da vermektedir.

Analiz çıktı dosyalarının tamamı, yukarıda ilk adımda hazırlanan klasör içerisine depolanmaktadır. Program kapatılana kadar yapılan testlerin tamamı bu klasör içerisine ve tek bir "csv" dosyasında, ancak "txt" formatındaki detaylı analiz dosyaları ise hangi dosyanın analiz edildiğini gösterir nitelikte adlandırma yapılarak kaydedilmektedir.

Aşağıda Evaluator_P" test modülünün "Evaluate_anestezi1.txt" dosyalarının "mas45.pdf" adlı bir Japonca yazın üzerinden test edildiğinde elde edilen sonuçlar

görülmektedir. Bu dosya analiz dokümanları arasında bulunan ve Japonya'daki anestezi bilimi ile ilgili olan "masui2.pdf" ten bir kesittir ve 2 sayfadan oluşmaktadır. Burada açıklama amacıyla ele alınmıştır ancak analizlerde dosyanın bütünü kullanılmıştır. Aşağıda test modülü çalıştırılması neticesinde elde edilen "csv" ve "txt" dosyalarının içerikleri görülmektedir.

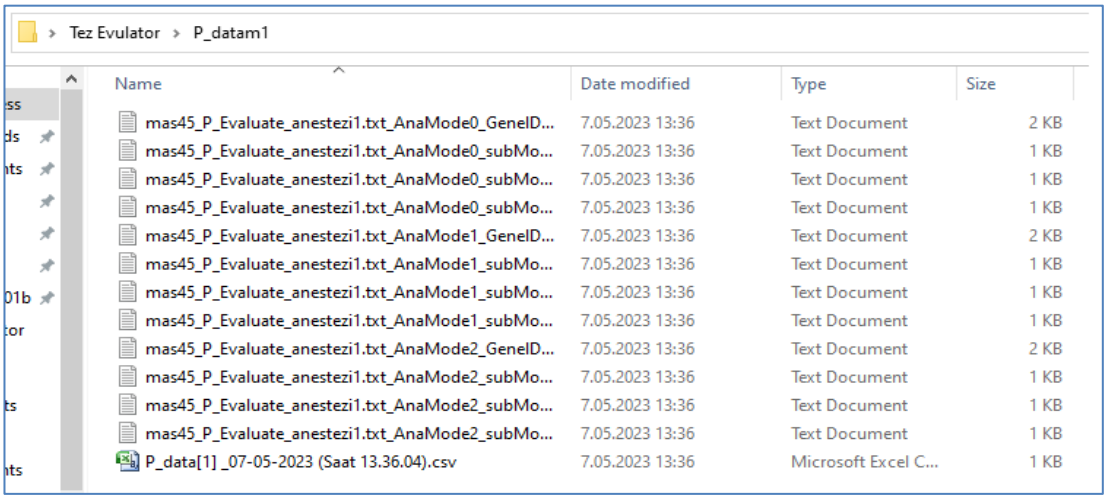
Şekil 40: Evaluator_P "csv" Dosyası Çıktıları



Kaynak: Yazar tarafından derlenmiştir.

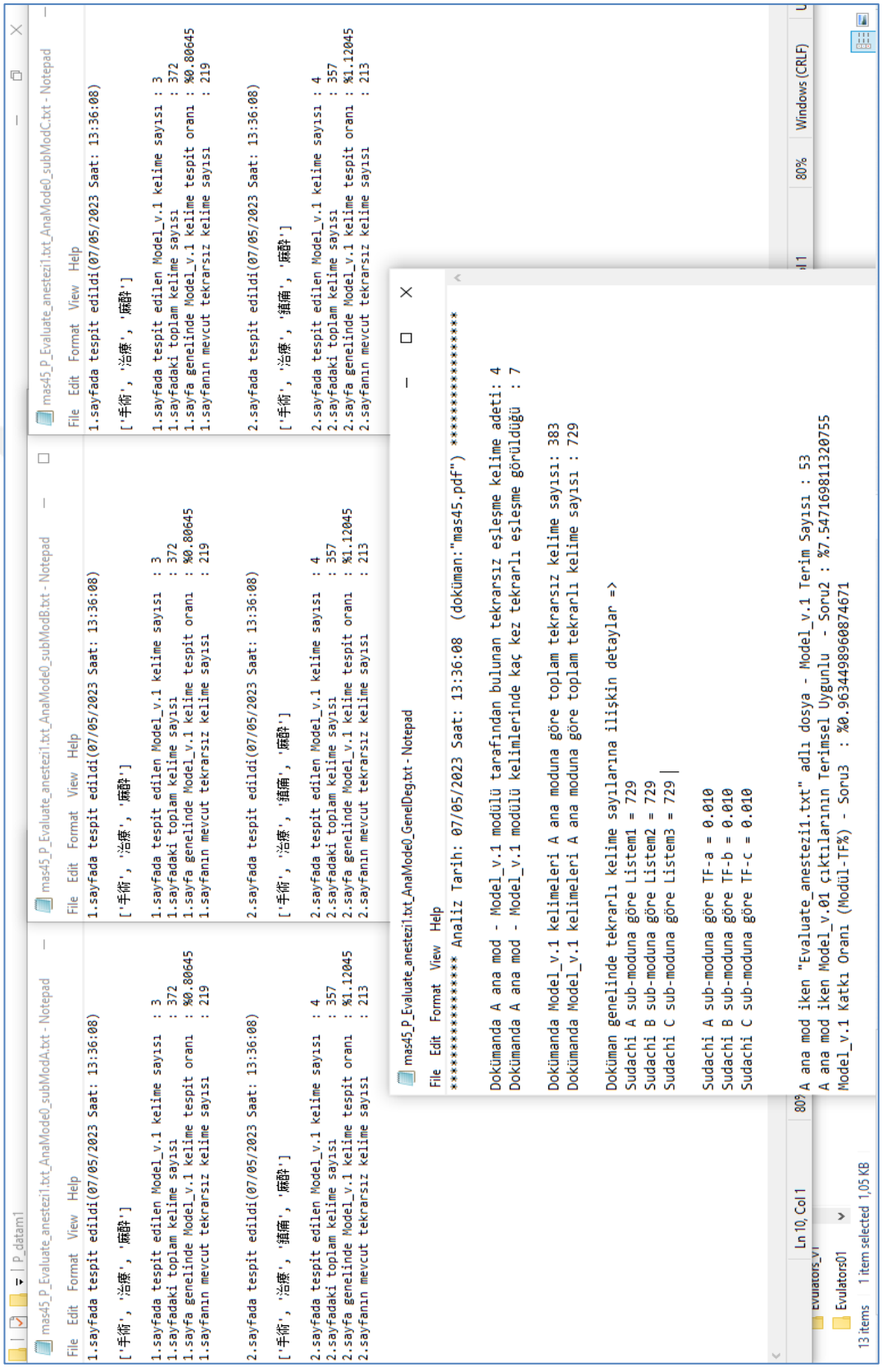
Csv dosyasında doküman adı ve analizi yapılan Model_v.1 çıktı dosyasına ilişkin meta türünde etiketlerin ardından, en ön sırada yanıt aradığımız soru 3'ün cevabı olan KATKI adlı Model_v.1 çıktı katkısının görüldüğü kısım yer almaktadır. Burada, ana metin üzerinde de 3 farklı Sudachi modu söz konusu olduğundan bu modlara göre ayrı ayrı hesaplama ve değerlendirme yapılmaktadır. Ardından genel bir değerlendirme yapılır ve tüm içerik csv dosyasına aktarılır. Aşağıda ise "txt" formatındaki dosyalar görülmektedir.

Şekil 41: Evaluator_P Çıktı Dosyaları



Kaynak: Yazar tarafından derlenmiştir.

Şekil 42: Evaluator_P "txt" Dosyası Çıktıları



Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü üzere, test edilen Model_v.1 kelime destek modülü çıktı kelimeleri Japonca yazında tespit edildiğinde bunlar "txt" uzantılı dosyalara aktarılmış ve bunların hangi sayfada tespit edildiği, kaç adet oldukları gibi bilgilerin yanı sıra, hangi kelimeler oldukları ve sayfadaki tekrarlı - tekrarsız kelime sayısı toplamı gibi analiz açısından önemli bilgiler de her sayfa için ayrı ayrı hesaplanıp bu dosyaların içerisine yazdırılmıştır. Bu türdeki dosyalar yukarıdaki şekilde üst kısımda bulunan dosyalardır ve aslında veri üretimi açısından en alt seviyede yer alırlar. Bu dosyalar sayesinde test edilen model kelimelerinin sayfalara göre davranışlarını ve bu davranışlara ilişkin detayları tek tek incelemek mümkündür ancak zaman alıcıdır. Bu sebeple tüm bu sayfa sayfa yapılan detaylı analizin toplu olarak görüntülenebildiği bir üst seviyedeki dosya türüne ihtiyaç duyulduğundan test modülünün kodlaması buna göre yapılmıştır. Bu seviyede olan dosyaların adı ise "…..GenelDeg.txt" şeklindedir. Farklı Sudachi sub-modlarında üretilen verileri toparlayarak toplu halde görüntülenmesine ve bu bilgilerin hesaplanarak bir üst seviyeye aktarılmasına imkan sağlarlar. Yukarıda orta-aşağı kısımda görüldüğü gibi bu tür dosyaların içerisinde, modül kelimelerinin tekrarlı ve tekrarsız eşleşmelerinin sayılarına, mod ayrımına göre olan kelime sayı değişikliklerine ya da mod farklılığına göre değişebilen terim frekans değerleri ile soru2 ve soru3 gibi cevabını merak ettiğimiz bilgilere ait alt seviyeden gelen verilere bağlı olarak değişebilen hesaplama sonuçlarına göre elde edilen ara değerlendirme rakamlarına yer verilir. Bu "Genel Değerlendirme" dosyaları Evaluator_P modülünde ön Sudachi moduna yer verilmediğinden sadece her ana mod için 1 adet üretilir. Dolayısıyla, 3 farklı Sudachi ana modu mevcut olduğundan her Japonca makale analizinde 3 adet üretilir. Evaluator_P test modülünün çalışması neticesinde elde edilen bu veriler son aşamada ise derlenerek "csv" dosyasına aktarılırlar. Böylece test edilen Model_v.1 modülü kelime çıktıları ile bu çıktıların Japonca yazınlar üzerinde test edilmesi neticesinde veri elde edilerek analiz için data oluşturulur.

Evaluator_P test modülünde test edilmek üzere hazırlanan Model_v.1 modülü kelime çıktıları ile bu çıktı türleri ile üzerinde testi gerçekleştirilen Japonca yazınlara ilişkin bilgiler ise aşağıdaki şekildedir. Gerçekleştirilen testte Sosyal Bilimler alanından 2 branş ile Fen Bilimleri alanından 2 branş kelimesi hedef olarak seçilerek destek modülü kelime çıktısı oluşturulmuş ve böylece denge hedeflenmiştir.

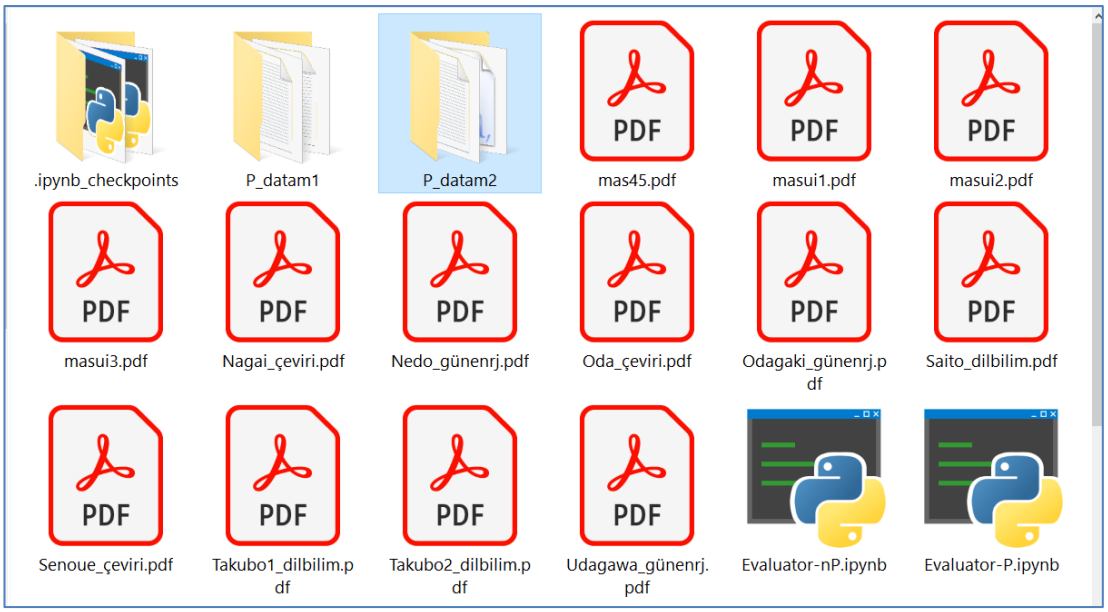
Tablo 13: Test edilen Model_v.1 Kelime Dosyaları

Evaluate_anestezi1.txt	Date modified: 1.05.2023 17:18 Size: 838 bytes	53
Evaluate_çeviri1.txt	Date modified: 8.05.2023 19:48 Size: 1,04 KB	77
Evaluate_dilbilim9.txt	Date modified: 27.04.2023 12:02 Size: 1,10 KB	80
Evaluate_güneşenerjisi1.txt	Date modified: 9.05.2023 01:19 Size: 1,84 KB	131

Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü gibi, tablonun sağında destek modülü kelime çıktı sayıları verilmiştir. En düşük kelime sayısına sahip olan anestezi1 sadece 53 kelimeden oluşmakta iken, en yüksek kelime sayısına sahip dosya güneşenerjisi adlı dosya ise 131 kelimeden oluşmaktadır. Üzerinden testi gerçekleştirilen Japonca yazınlar ise aşağıdaki şekilde olup, yazarı belli olanlarda yazar soyadını takiben hangi alana ait olduğunu ayırt eder nitelikte dosya adları düzenlenmiştir ve her alandan üç farklı Japonca yazın dokümanı testte kullanılmıştır.

Şekil 43: Testte Kullanılan Japonca Yazın Dosyaları



Kaynak: Yazar tarafından derlenmiştir.

Test işleminin tamamı Japonca yazın metin uzunluğuna göre değişiklik gösterse de mevcut donanımla 24 saatte yakın sürmüştür. Özellikle 15 - 20 sayfalık metinlerden sonuç alınmasının yaklaşık 3 - 4 saat sürdüğü durumlar olmuştur. Tüm metinler üzerindeki test işlemi tamamlanana kadar Jupiter oturumu açık tutulmuştur. Yapılan teste bağlı olarak elde edilen analiz sonuçları aşağıdaki şekilde olmuştur.

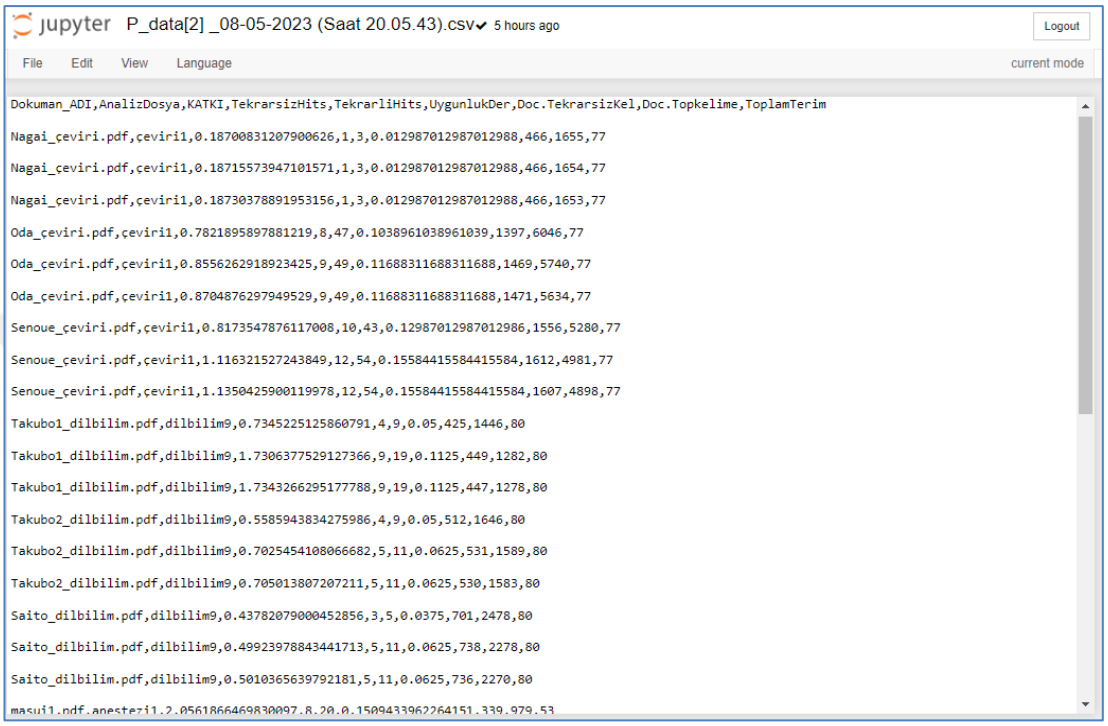
Şekil 44: Evaluator_P tarafından en alt seviyede üretilen analiz dosyaları



Kaynak: Yazar tarafından derlenmiştir.

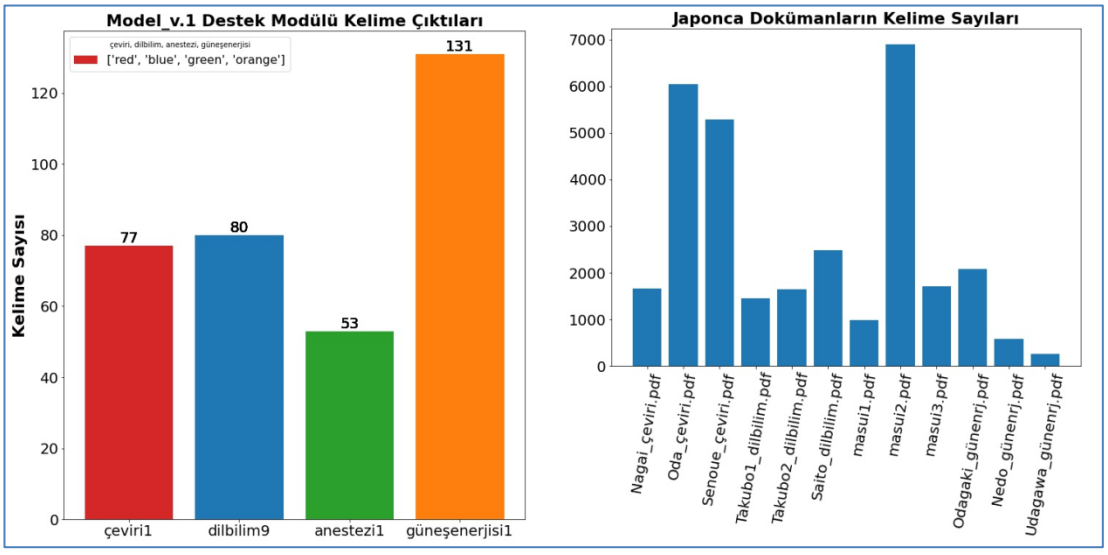
Elde edilen tüm bu test verileri neticesinde ise analiz için Evaluator_P modülü tarafından oluşturulan “P_data[2] _08-05-2023 (Saat 20.05.43).csv” dosya içeriği ise aşağıdaki şekilde kısmi olarak görüntülenmiştir.

Şekil 45: Evaluator_P Analiz Sonuçları Dosyası (CSV)



Kaynak: Yazar tarafından derlenmiştir.

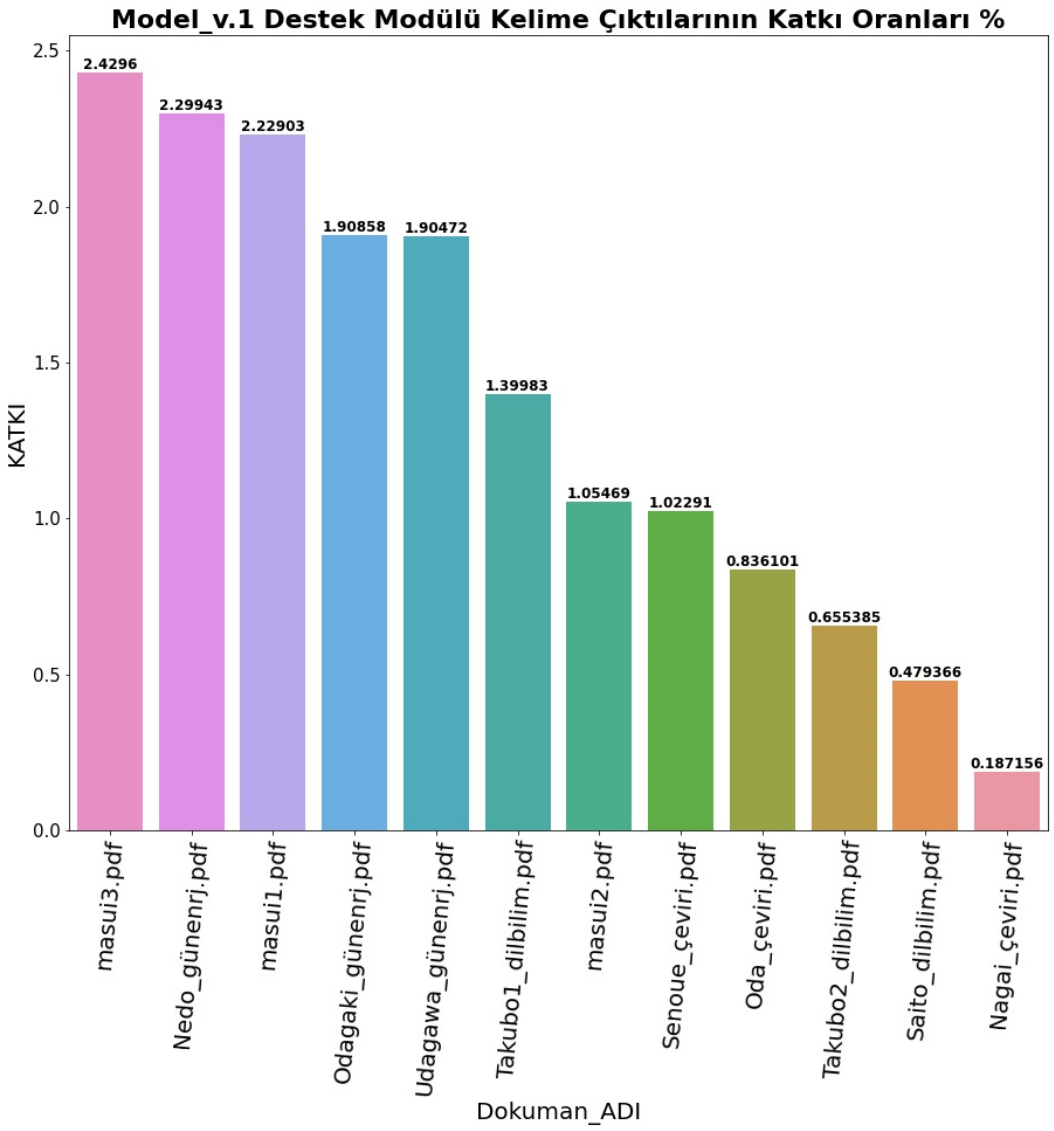
Şekil 46: Model_v.1 Kelime Destek Modülü ile Japonca Doküman Kelimeleri



Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü üzere bir sonraki adımda, elde edilen tüm bu verilerin analiz aşamasına geçilmiştir. Analiz aşamasında veri görselleştirme tekniğinden faydalanılmış ve bunun için "Matplotlib", "Pandas" ve "Seaborn" kütüphanelerinden yararlanılmıştır. Yukarıda, test edilen Model.v1 Kelime Destek Modülü kelime çıktıları ile üzerinden testi gerçekleştirilen Japonca yazınlar hususunda bilgiler verilmiştir. Buna göre en yüksek kelime sayısına sahip küme "güneşenerjisi" alanındadır ve toplamda 131 adet kelimeden oluşmuştur. Aşağıda ise destek modülü çıktılarının her dokümanda ne derece başarılı olduğunu gösteren KATKI oranlarının Japonca yazınlar üzerindeki dağılımı görülmektedir.

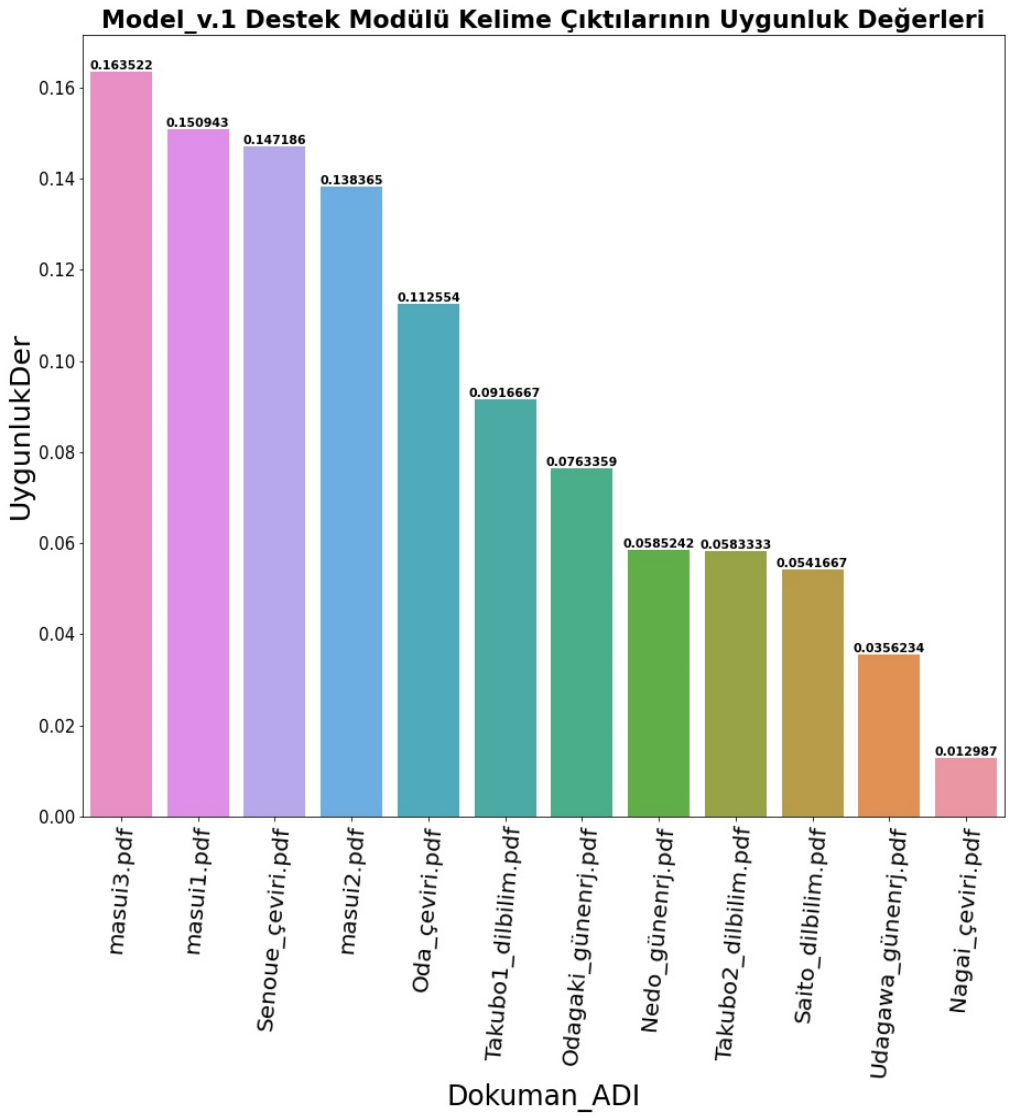
Şekil 47: Model_v.1 Kelime Destek Modülü Katkı Oranları (Evaluator_P)



Kaynak: Yazar tarafından derlenmiştir.

Model.v1 Kelime Destek Modülü kelime çıktılarının KATKI değerlendirmelerine bakıldığında, aslen “麻酔を受けられるみなさまへ (masui o ukerareru minasama e)” başlıklı olan ve Türkçede “anestezi alan herkese” anlamına gelen masui3 adlı Japonca yazın en yüksek KATKI oranını yakaladığı anlaşılmıştır. Bir sonraki adımda da Uygunluk Değerlerine bakılmış ve bu değerler de aşağıdaki şekilde her doküman için değerleri ayrı ayrı gösterecek şekilde görselleştirilmiştir.

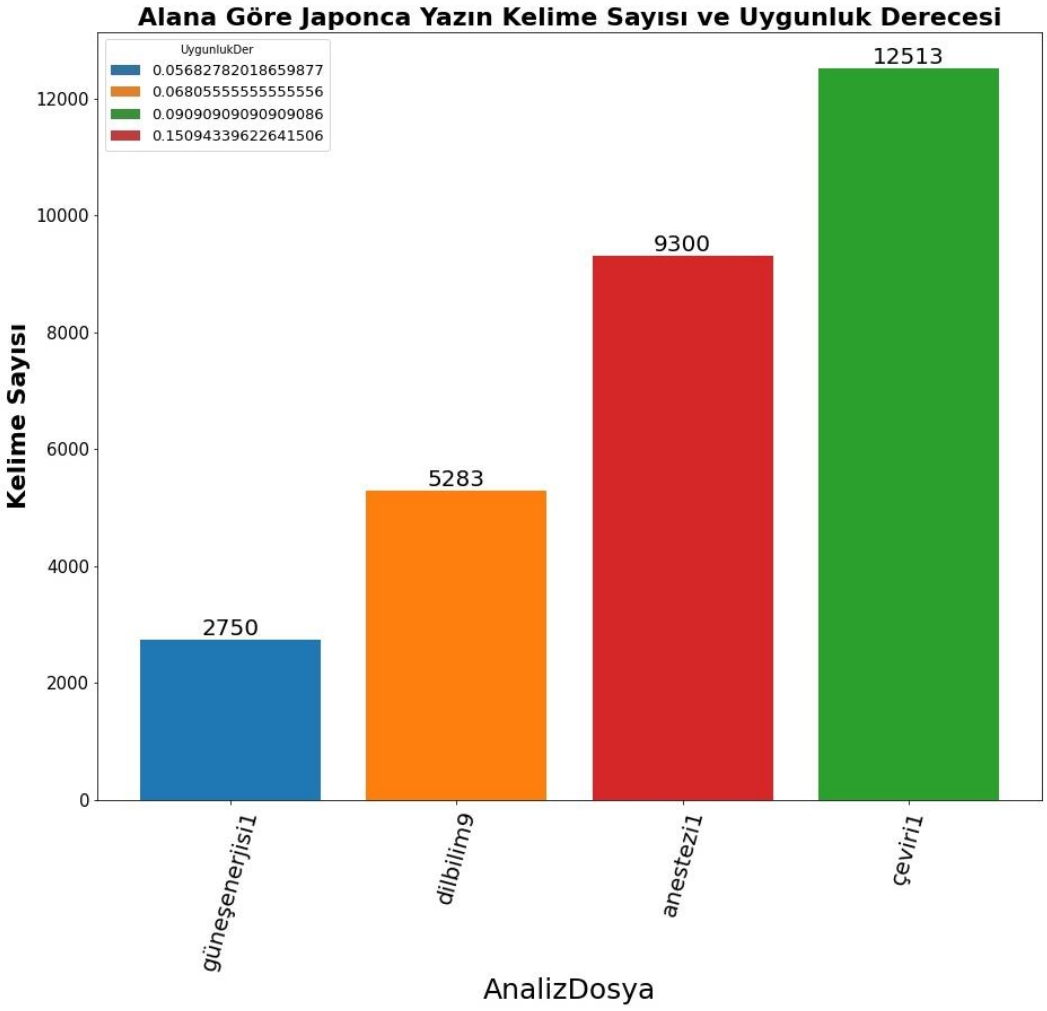
Şekil 48: Model_v.1 Kelime Destek Modülü Uygunluk Değerleri (Evaluator_P)



Kaynak: Yazar tarafından derlenmiştir.

Uygunluk değerlerine bakıldığında ise, yine masui3 adlı dosya ilk sırada yer almıştır. Ancak ikinci sıradaki doküman adı KATKI grafiğindekinden bu kez farklı olarak KATKI oranında 3. sırada bulunan masui1 adlı doküman olmuştur. Öte yandan bu farklılığın mevcut olması normaldir çünkü, Uygunluk Değer hesaplaması "Model_v.1 kelimeleri tekrarsız hit sayısı ÷ Model kelime listesi toplam sayısı" şeklinde hesaplandığından, doküman genelindeki tekrarsız hit sayısı, dokümandan dokümana farklılık göstermekte ve bu da uygunluk değerine yansımaktadır. Bu noktada alana göre üzerinden testi yapılan dokümanlardaki toplam kelime sayısı ile uygunluk değerlerine de bakmakta fayda vardır.

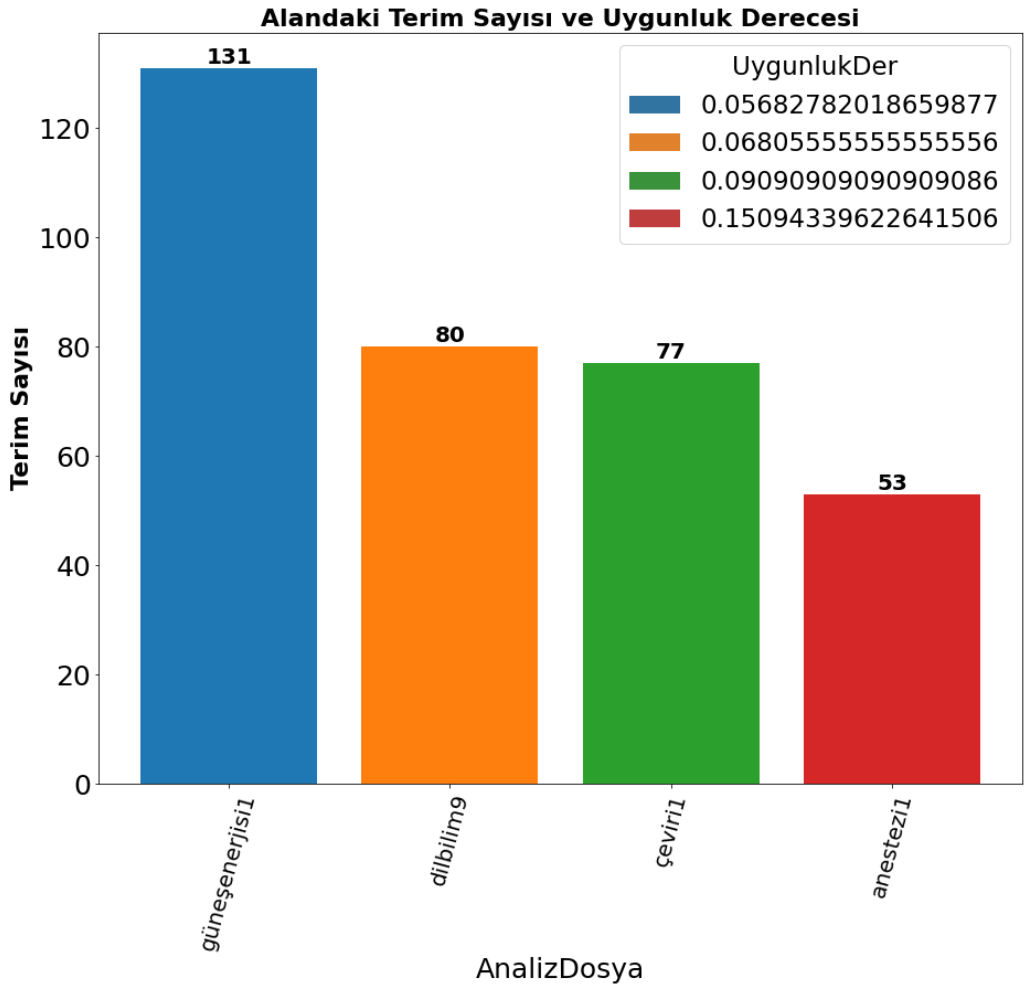
Şekil 49: Alana Göre Toplam Japonca Yazın Kelime Sayısı



Kaynak: Yazar tarafından derlenmiştir.

Yukarıdaki grafikten de anlaşıldığı üzere, uygunluk değeri en yüksek olan grubun anestezi adlı grup, yani Model_v.1 modülü destek kelimelerinden "anestezi1.pdf" adlı 53 kelimeden oluşan dosya olduğu görülmektedir. Uygunluk değerleri ile terim sayı kümesi arasında herhangi bir bağıntı olup olmadığını daha iyi görebilmek için ise Model_v.1 modülü kelime sayılarına da bakmakta fayda vardır.

Şekil 50: Alandaki Terim Sayısı ve Uygunluk Derecesi (Evaluator_P)



Kaynak: Yazar tarafından derlenmiştir.

Yukarıdaki grafikten de anlaşıldığı üzere uygunluk derecesi ile Model_v.1 modülü kelime sayıları arasında ters bir orantı mevcuttur. Diğer bir deyişle herhangi bir alandaki terim sayısı arttıkça terim başına düşen birim ağırlık düşmektedir de denilebilir.

Öte yandan, KATKI ile Uygunluk arasında bir bağlantı olsa da, tüm dokümanlarda aynı şekilde paralellik gösterir nitelikte değildir. Ancak genel itibariyle anestezi alanı kelimelerinin hem KATKI oranı hem de Uygunluk derecesi açısından ilk sırada yer aldığını söylemek mümkündür.

Şekil 51: Katkı Derecesi (Evaluator_P)

```
In [163]:
data = pd.read_csv("P_datam2/P_data[2] _08-05-2023 (Saat 20.05.43).csv")

grd = data.groupby('AnalizDosya').mean().sort_values('KATKI', ascending=False)
ortTer = grd[grd.columns[-1]]
ortUyg = grd[grd.columns[3]]
ortKat = grd[grd.columns[0]]

print(ortKat, "\n")
print(ortUyg, "\n")
print(ortTer, "\n")
```

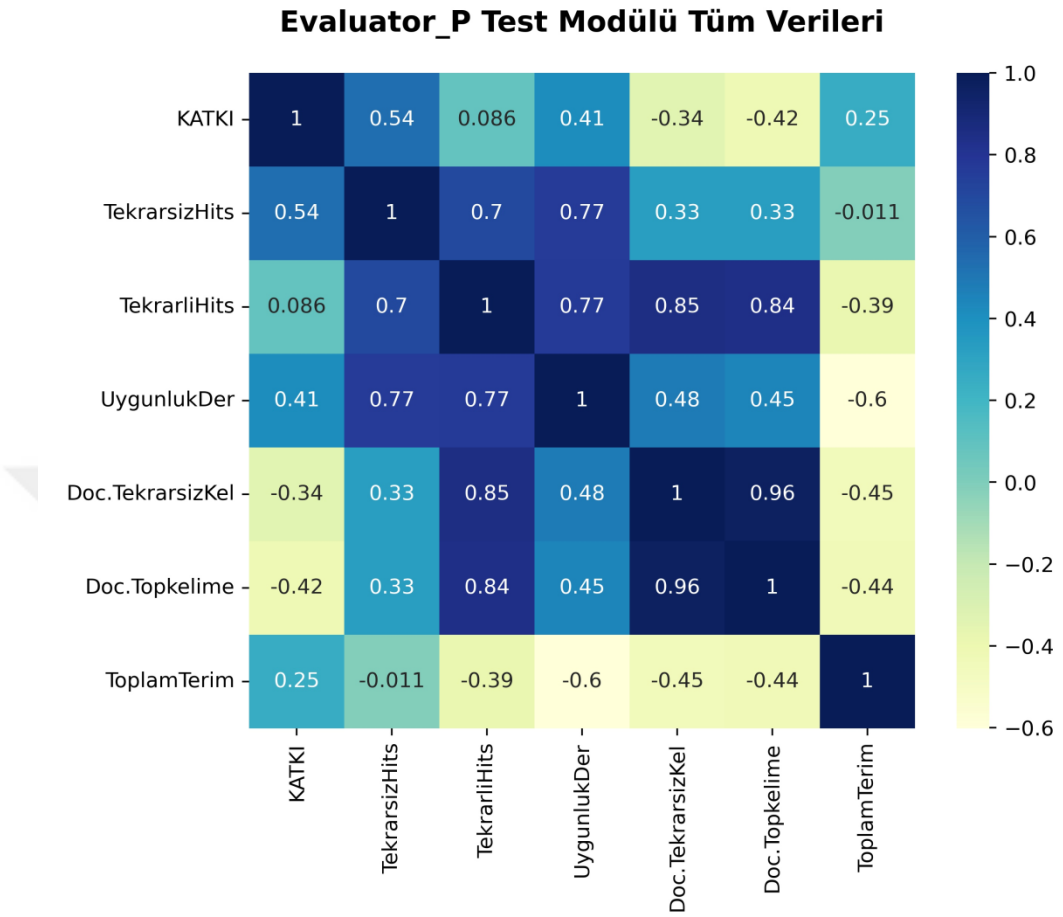
```
AnalizDosya
güneşenerjisi1    2.037574
anestezi1         1.904441
dilbilim9         0.844860
çeviri1           0.682054
Name: KATKI, dtype: float64

AnalizDosya
güneşenerjisi1    0.056828
anestezi1         0.150943
dilbilim9         0.068056
çeviri1           0.090909
Name: UygunlukDer, dtype: float64

AnalizDosya
güneşenerjisi1    131.0
anestezi1          53.0
dilbilim9          80.0
çeviri1            77.0
Name: ToplamTerim, dtype: float64
```

Kaynak: Yazar tarafından derlenmiştir.

Diğer bir yandan, yukarıdaki grafiklerden ve kod-çıktı ekranından da anlaşıldığı ve daha önce de bahsedildiği üzere csv uzantılı analiz dosyasındaki değerler 3 farklı Sudachi moduna göre gerçekleştirildiğinden, her doküman üç kez analiz edilerek yazılmaktadır, ancak dosya analizi aşamasında bu değerlerin ".mean()" komutu ile ortalaması alınmıştır.

Evaluator_P test modülünün bir sonraki analiz aşaması olarak ise, tüm bu verilerin tek bir ekranda görülebilmesi için "Heatmap" ile son değerlendirme adımına geçilmiştir.

Şekil 52: Heatmap Sonuçları (Evaluator_P)



Kaynak: Yazar tarafından derlenmiştir.

Yukarıdan da anlaşıldığı üzere, sol üst köşedeki KATKI'ya göre, Tekrarsız Hits, Uygunluk Derecesi ve Tekrarlı Hits pozitif yönde korelasyon gösterirken, Doküman Tekrarsız Kelime Sayısı ile Doküman Toplam Kelime Sayısı negatif yönde korelasyon gösterir. Öte yandan sağ alt köşedeki Toplam Terim Sayısı açısından değerlendirildiğinde ise, en büyük negatif yöndeki korelasyonun Uygunluk Derecesi ile olduğu buradan da anlaşılmaktadır.

Buraya kadar yapılan analiz neticesinde Evaluator_P test modülü çıktıları göz önünde bulundurularak Model_v.1'in etkililik açısından değerlendirilmesi hususunda, artık aşağıdaki sorulara cevap vermek mümkündür.

1) Çevirmenler için teknik alana yönelik olarak hazırlanan Türkçe - Japonca kelime destek modülü olan Model_v.1'in ürettiği terimler o alandaki Japonca yazınları okurken kelime bilgisi açısından destek sağlayabilir mi?

⇒ **EVET**

2) Model_v.1'in, ürettiği terimlerin o alanda yazılmış Japonca metinler üzerinde ölçümlendirilebilir bir uygunluk derecesi mevcut mu?

⇒ **EVET (Genel Uygunluk Yüzdesi : % 9.168396571941507)**

3) Model_v.1 modülü kelime çıktıları Japonca akademik yazınlar üzerinde tespit edilebilir ölçüde bir katkı sağladı mı?

⇒ **EVET (Genel Katkı Yüzdesi : % 1.3672324287563256)**

Soru 2 ve 3'ün cevabı aşağıdaki hesaplama ekranından da teyit edilebilir.

Şekil 53: Evaluator_P'ye Göre Model_v.1'in Uygunluk ve Katkı Oranları

```
In [28]:
data = pd.read_csv("P_datam2/P_data[2] _08-05-2023 (Saat 20.05.43).csv")

grd = data.groupby('AnalizDosya').mean().sort_values('KATKI', ascending=False)

genKat = grd[grd.columns[0]].mean()
genUyg = grd[grd.columns[3]].mean()

# Uygunluk Değeri yüzde olarak verilirken formül gereği bölüm sonucu x100'dür.
print("Genel Uygunluk Yüzdesi: %", genUyg*100, "\n")
print("Genel Katkı Yüzdesi   : %", genKat, "\n")


Genel Uygunluk Yüzdesi: % 9.168396571941507

Genel Katkı Yüzdesi   : % 1.3672324287563256
```

Kaynak: Yazar tarafından derlenmiştir.

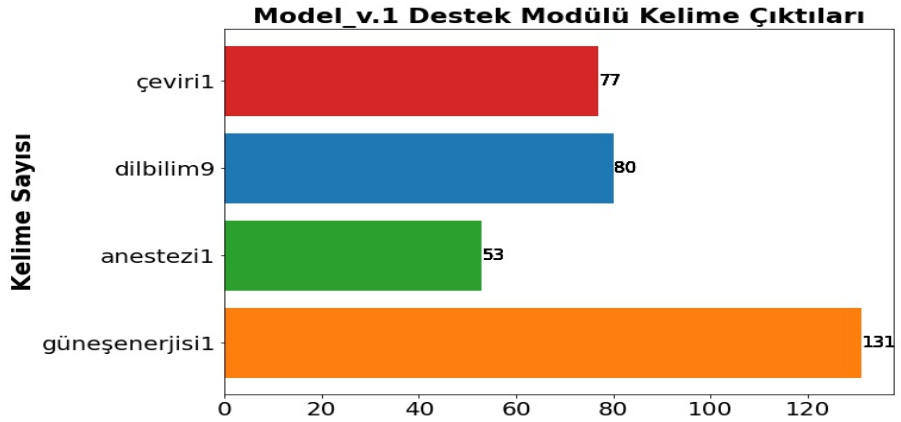
Sonuç itibariyle, Evaluator_P test modülüne göre yapılan test ve analiz sonuçlarına bağlı olarak Model_v.1 Modülü çıktıları, değerleri açısından çok yüksek oranlarda olmasa da *Etkililik (effectiveness)* açısından beklentileri karşılar niteliklere sahiptir.

4.3. EVALUATOR_nP MODÜLÜ İLE ETKİLİLİK DEĞERLENDİRMESİ

Evaluator_nP test modülü daha az kuralcı bir modül olarak, Model_v.1'in kelime çıktılarını anlamlı bütünler halinde bölerek Japonca yazınlar içerisinde arayacak şekilde tasarlanmıştır. Yukarıda anlatılan Evaluator_P test modülünden en büyük farkı, başlangıç aşamasında Model_v.1 modülü destek kelime listesini Sudachi modlarına göre bölümleyerek test modülüne göndermesidir. Bu durum, başlangıcında Sudachi modlarının 3'ünü de sıra ile çalıştıran bir "for" döngüsü içerisine Evaluator_P modül kodlarının alınması ve diğer kısımlarında da gereken değişikliklerin yapılması sayesinde elde edilmiştir. Dolayısıyla, for döngüsü dışındaki işlemlerin hepsi hemen hemen aynı olduğundan burada tekrardan kodların paylaşımına gerek duyulmamıştır.

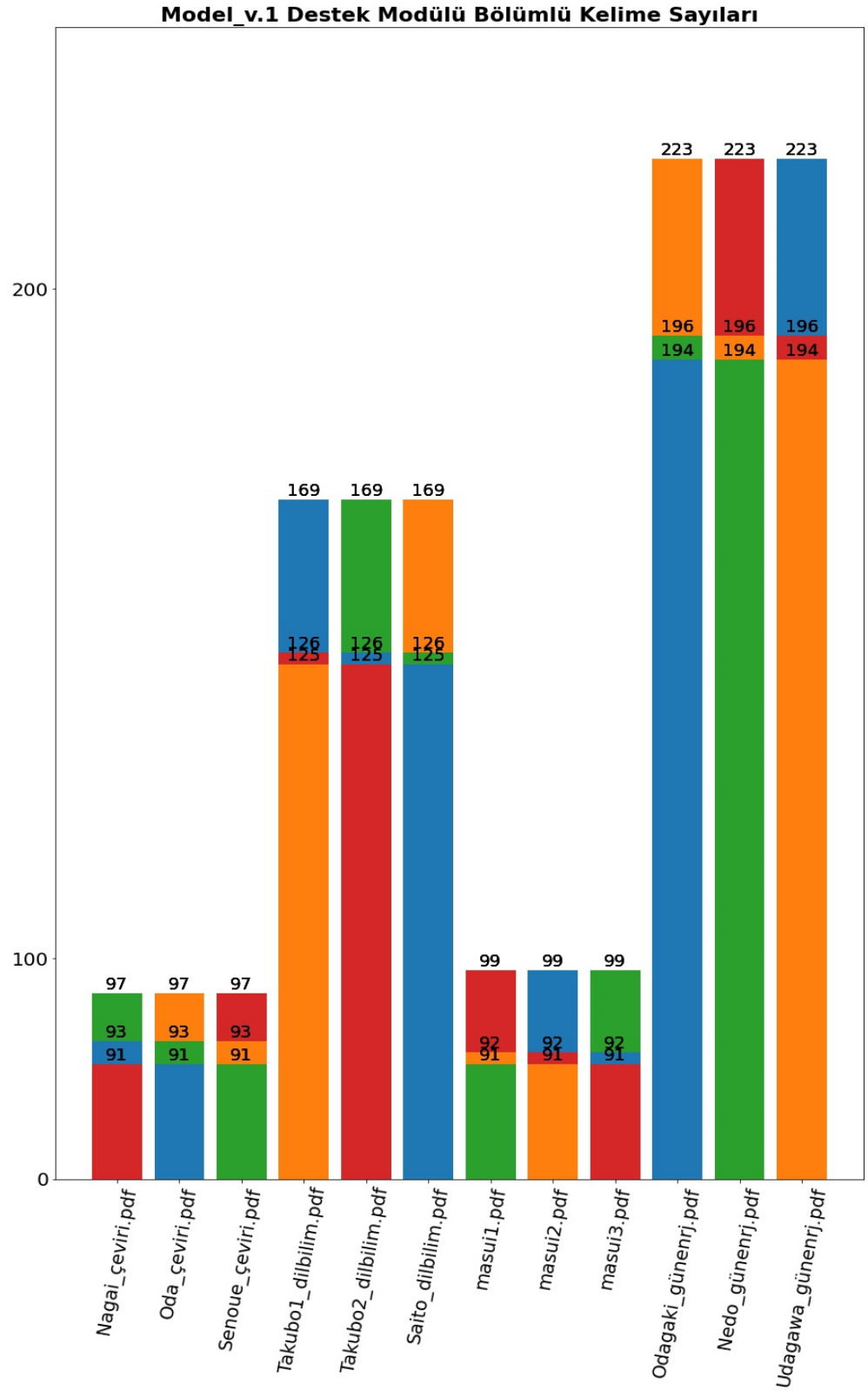
Evaluator_nP test modülü de çalıştırıldığında yine "txt" ve "csv" uzantılı analiz dosyaları üretmektedir. Ancak bu kez, en altta 9 yerine 3 x 9 = 27 adet en alt seviyede dosya üretir. Daha sonra bu en alt seviyedeki dosya verileri bu kez 3 yerine 9 adet üretilen "...GenelDeg.txt" şeklinde dosyalara her 3 modun tamamlanmasını takiben gönderilir. Burada "...GenelDeg.txt" dosyaları orta seviyedeki veri katmanı olarak düşünülebilir. En son aşama olarak ise ara katmandaki bu veriler toplanarak bir üst seviye olan csv dosyasına kaydedilir. Öte yandan bu kez ekstradan bir for döngüsü de ilave edildiğinden tüm bu analizlerin tamamlanması mevcut donanımla 24 saatten fazla sürmüştür. Aşağıda hatırlama amacıyla ön bölümleme öncesinde Model_v.1 modülünün kelime çıktı sayıları ile ön bölümleme sonrasındaki durum görüntülenmektedir.

Şekil 54: Model_v.1 Destek Modülü Bölümleme Öncesi Kelime Sayıları



Kaynak: Yazar tarafından derlenmiştir.

Şekil 55: Evaluator_nP tarafından yapılan ön bölümleme sayıları



Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü üzere, Sudachi modlarına göre ön bölümleme yapıldığında, destek modülünün kelime sayıları Sudachi moduna göre 3 farklı baremde artış göstermiştir. Aynı bölümleme modunda olmasına rağmen bazılarında bir bazılarında iki kelime fark etmesi, farklı alanlara mensup destek modülü kelimeleri içerisinde yer alan kelimelerinin kendi özelliklerinden kaynaklanmaktadır. Dolayısıyla, bazı kelimeler daha çok bölümlenebilirken, bazıları ise daha az bölümlenebilmektedir. Burada bir diğer konu, örneğin dilbilim alanındaki kelime liste uzunluğu normal hali 80 iken maksimum 169'a çıkmış, ancak dilbilimden sadece 3 kelime az olan çeviri alanındaki kelime liste uzunluğu 77 iken sadece maksimum 97 kelime olarak bölümlenebilmiştir. Dilbilime benzer bir durum anestezi ve güneş enerjisi alanında da gerçekleşerek, kelime sayıları maksimumda yaklaşık 2 katı ya da 2 katına yakın bir şekilde artış göstermiştir. Ancak çeviri dosyasına bakıldığında ise bu durum farklı olarak sadece dörtte bir oranındaki bir artış olarak yansımıştır. Bu durum, Model_v.1 modülü kelimelerine bakıldığında, çeviri alanındaki kelimelerde genellikle 2 Kanji karakterinden oluşan kelimeler çoğunlukta iken, diğer alanlardaki kelimelerde 3 ve daha fazla Kanji karakterinden oluşmuş kelime ve kelime öbeklerinin daha fazla sayıda mevcut olmasından kaynaklanmaktadır.

Şekil 56: Evaluator_nP Analiz Sonuçları Dosyası (CSV)

jupyter nP_data[1] _09-05-2023 (Saat 23.57.20).csv 4 hours ago

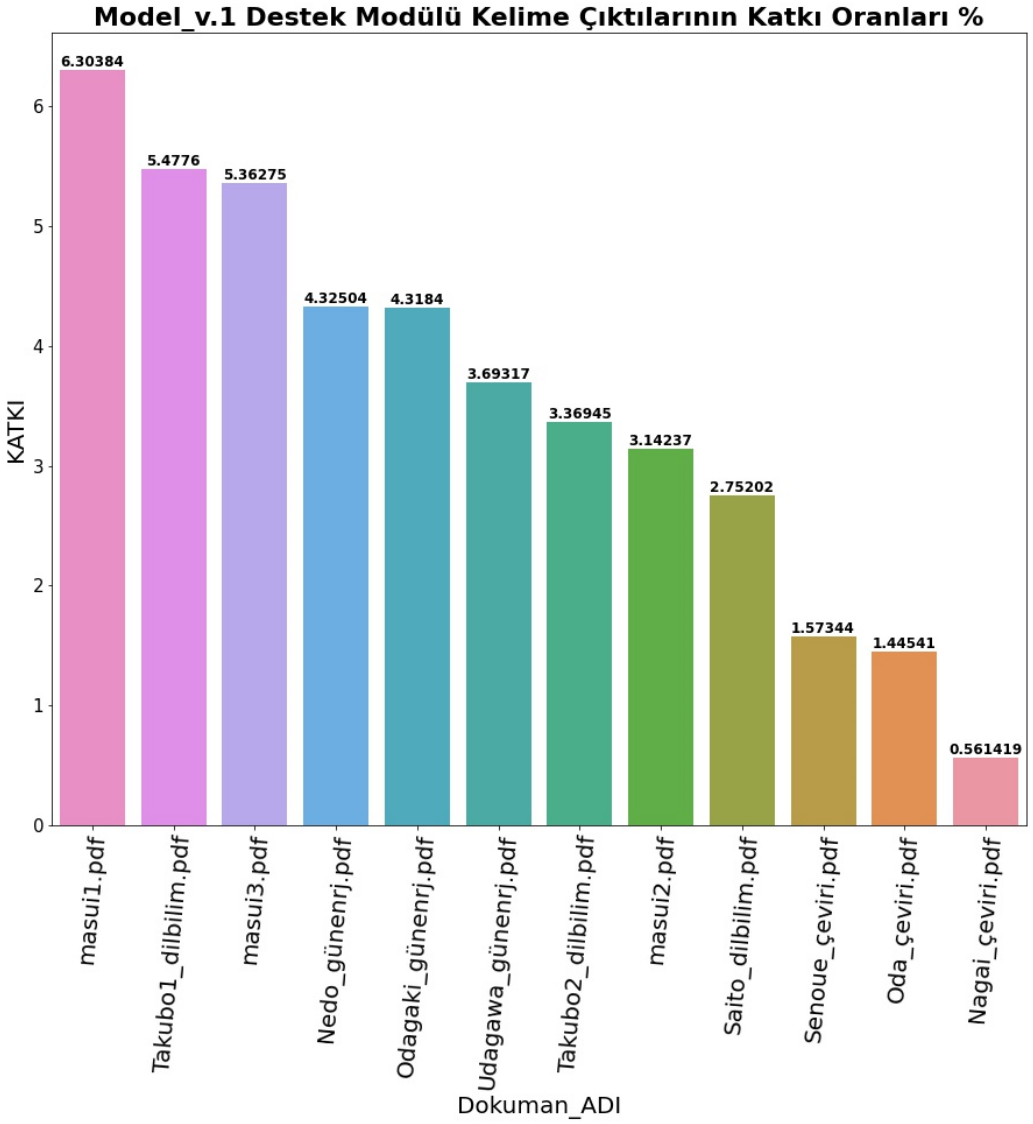
File Edit View Language

```
Dokuman_ADI,AnalizDosya,KATKI,TekrarsizHits,TekrarliHits,UygunlukDer,Doc.TekrarsizKel,Doc.Topkelime,ToplamTerim
Nagai_çeviri.pdf,çeviri1,0.9350415603950315,6,15,0.061855670103092786,466,1655,97
Nagai_çeviri.pdf,çeviri1,0.8656032587585875,5,14,0.05154639175257732,466,1654,97
Nagai_çeviri.pdf,çeviri1,0.8661954565526507,5,14,0.05154639175257732,466,1653,97
Nagai_çeviri.pdf,çeviri1,0.44404463536249433,3,7,0.03225806451612903,466,1655,93
Nagai_çeviri.pdf,çeviri1,0.37431147894203143,2,6,0.021505376344086023,466,1654,93
Nagai_çeviri.pdf,çeviri1,0.3746075778390631,2,6,0.021505376344086023,466,1653,93
Nagai_çeviri.pdf,çeviri1,0.44404463536249433,3,7,0.03296703296703297,466,1655,91
Nagai_çeviri.pdf,çeviri1,0.37431147894203143,2,6,0.02197802197802198,466,1654,91
Nagai_çeviri.pdf,çeviri1,0.3746075778390631,2,6,0.02197802197802198,466,1653,91
Oda_çeviri.pdf,çeviri1,1.6460256080695883,15,99,0.15463917525773196,1397,6046,97
Oda_çeviri.pdf,çeviri1,1.628960303145866,15,93,0.15463917525773196,1469,5740,97
Oda_çeviri.pdf,çeviri1,1.6555391319436137,15,93,0.15463917525773196,1471,5634,97
Oda_çeviri.pdf,çeviri1,1.27260425754167,12,76,0.12903225806451613,1397,6046,93
Oda_çeviri.pdf,çeviri1,1.372008287685775,13,78,0.13978494623655913,1469,5740,93
Oda_çeviri.pdf,çeviri1,1.3944757118716922,13,78,0.13978494623655913,1471,5634,93
Oda_çeviri.pdf,çeviri1,1.27260425754167,12,76,0.13186813186813187,1397,6046,91
Oda_çeviri.pdf,çeviri1,1.372008287685775,13,78,0.14285714285714285,1469,5740,91
Oda_çeviri.pdf,çeviri1,1.3944757118716922,13,78,0.14285714285714285,1471,5634,91
Sepoue_çeviri.pdf,çeviri1,2.0114811234943497,19,106,0.1958762886597938,1556,5280,97
```

Kaynak: Yazar tarafından derlenmiştir.

Yukarıda öncelikle “nP_data[1] _09-05-2023 (Saat 23.57.20).csv” adlı analiz sonuç dosya içeriğinin kısmi görüntüsü paylaşılmıştır. Bir sonraki aşamada ise Evaluator_nP test modülünün çalıştırma aşamasına geçilmiştir.

Şekil 57: Model_v.1 Kelime Destek Modülü Katkı Oranları (Evaluator_nP)

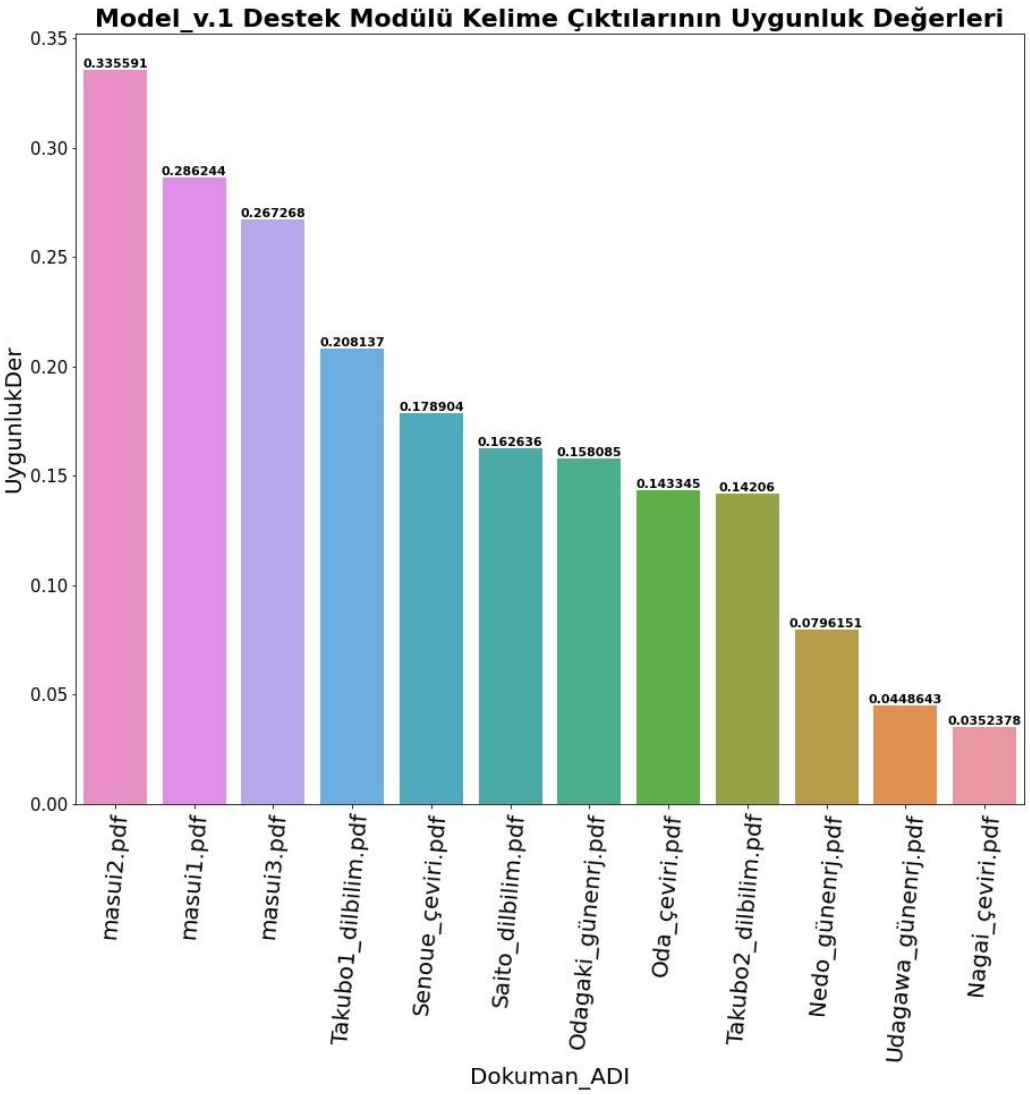


Kaynak: Yazar tarafından derlenmiştir.

Grafikten de anlaşıldığı üzere, Evaluator_nP ile yapılan test neticesinde KATKI oranları Evaluator_P’ye kıyasla daha yüksek seviyede çıkmıştır. Bu daha az kuralcı olarak nitelendirdiğimiz Evaluator_nP test modülünde bir kelime öbeğinin anlamlı halde birden fazla şekilde taranabilmesinden kaynaklanmaktadır.

Model.v1 Kelime Destek Modülü kelime çıktılarının KATKI değerlendirmelerine bakıldığında, bu kez Evaluator_P'de ilk sırada yer alan masui3'ün yerini masui1 adlı dosya almıştır. Bir sonraki adımda ise yine Uygunluk Değerlerine bakılmıştır.

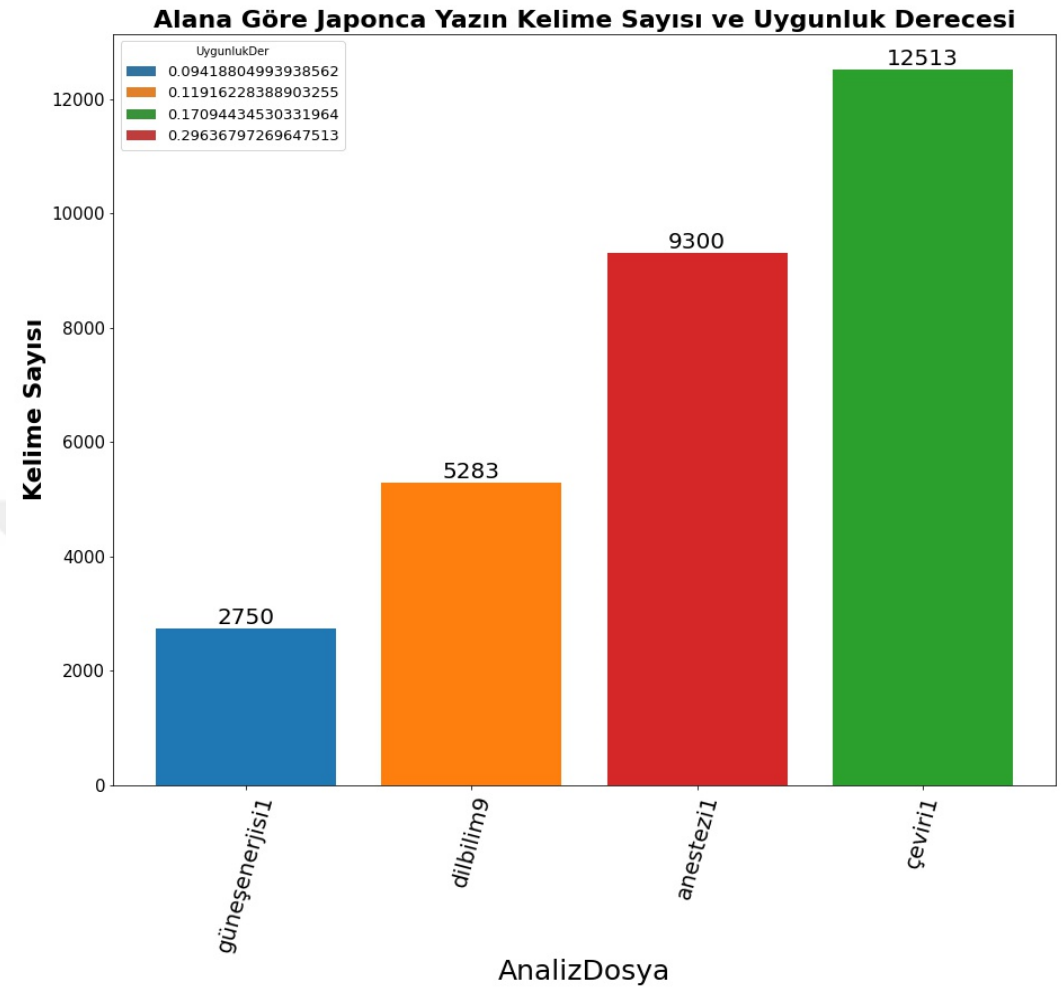
Şekil 58: Model_v.1 Kelime Destek Modülü Uygunluk Değerleri (Evaluator_nP)



Kaynak: Yazar tarafından derlenmiştir.

Uygunluk değerlerine bakıldığında, bu kez masui2 adlı ve anestezi alanındaki Japonca yazın ilk sırada yer almıştır. "Masui" kelimesi Japoncada "anestezi" anlamına gelir. Bu noktada yine alana göre üzerinden testi yapılan dokümanlardaki toplam kelime sayısı ile uygunluk değerlerine de bakmakta fayda vardır.

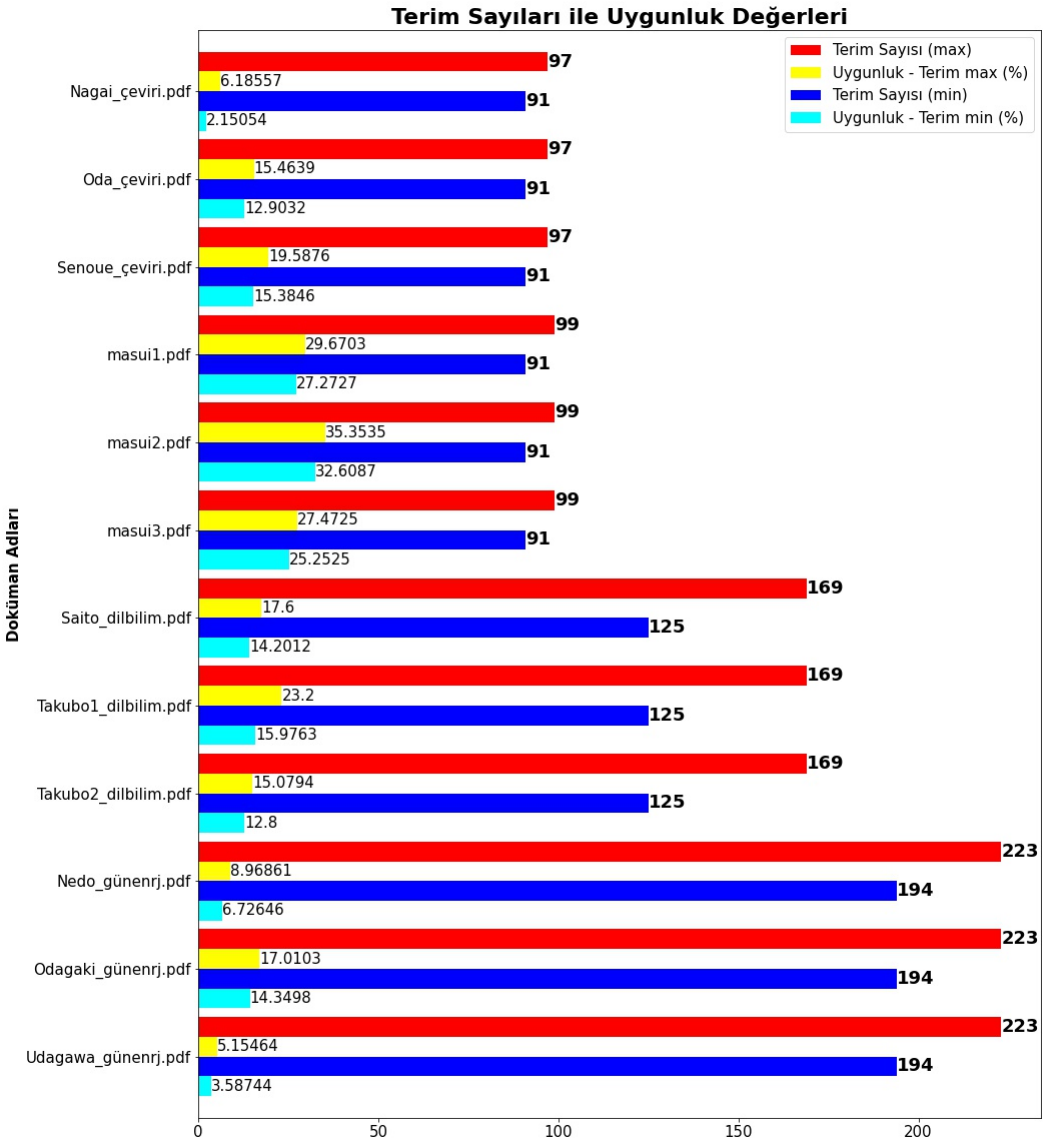
Şekil 59: Alana Göre Toplam Japonca Yazın Kelime Sayısı (Evaluator_nP)



Kaynak: Yazar tarafından derlenmiştir.

Burada yine Uygunluk Değerleri ile Doküman Toplam Kelime sayısı arasında direk açıklanabilir bir bağlantı olduğunu söylemek zordur. Ancak esas dikkat çekici olan yine Sudachi ön bölümleme öncesinde en düşük terim sayısına sahip olan anestezi alanının ortalamada yine en yüksek uygunluk değerini vermesidir. Uygunluk Değeri formülünden kaynaklı olduğu düşünülebilir ancak formüle tekrar bakıldığında, "Model_v.1 kelimeleri tekrarsız hit sayısı ÷ Model kelime listesi toplam sayısı" şeklinde hesaplandığından, doküman genelindeki tekrarsız hit sayısı bağımsızdır, dokümandan dokümana farklılık gösterebilmektedir. Bu kez ön bölümlemeye bağlı olarak tekrarsız hit sayıları değişmiş olmasına rağmen yine benzer şekilde bir korelasyon görülmesi oldukça ilgi çekicidir. Aşağıda bu durumu metinler üzerinden daha da incelemekte fayda vardır.

Şekil 60: Maksimum ve Minimum Ön Bölümleme ile Uygunluk (Evaluator_nP)

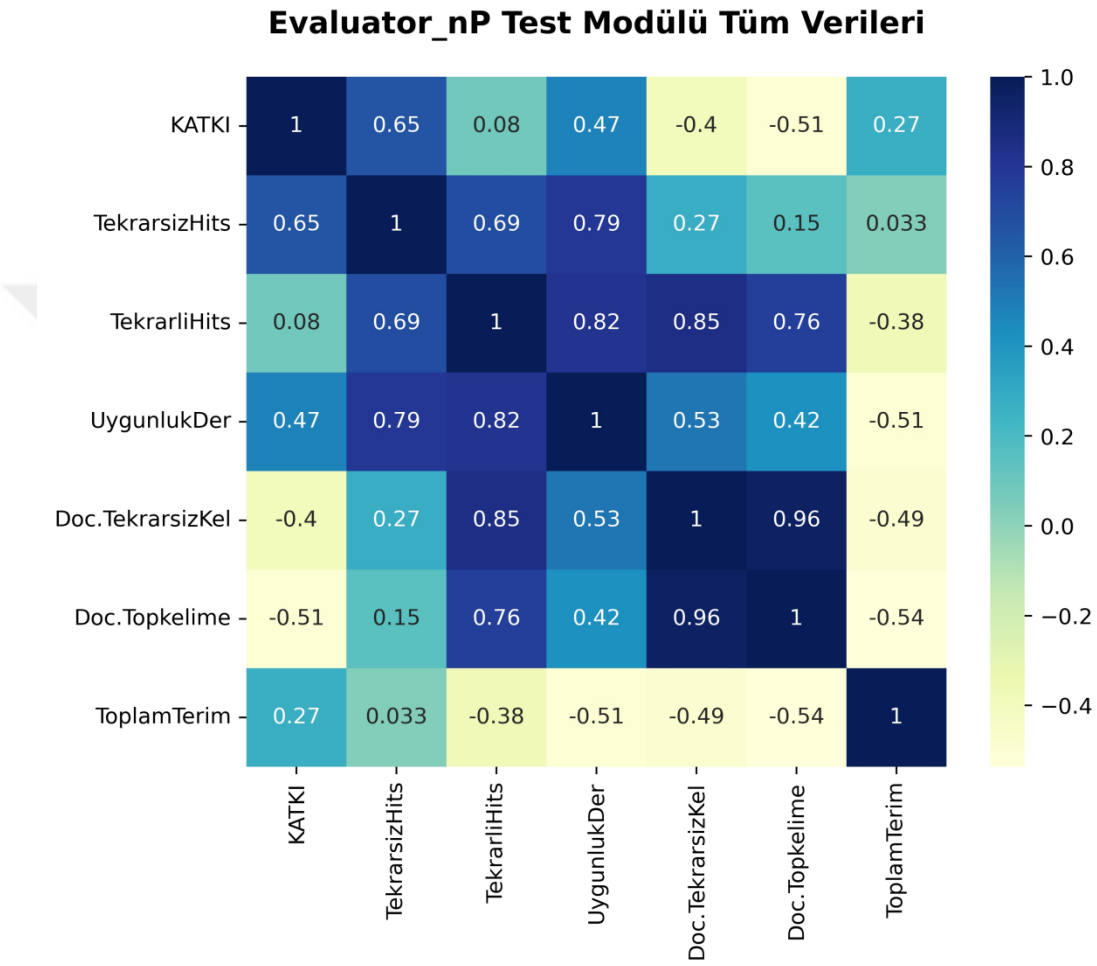


Kaynak: Yazar tarafından derlenmiştir.

Yukarıda da görüldüğü gibi, aslında ön bölümleme neticesindeki terim artışının uygunluk değerlerine aynı metin içerisinde olumlu yönde katkısı olmaktadır. Ancak başka bir metin söz konusu olduğunda ise bu etki negatif yönde değişmektedir. Bunun bir sebebi olarak, Sudachi sub-modları gösterilebilir. Ancak bu durumun tam olarak anlaşılması ancak sabit bir mod üzerinden benzer bir uygulama gerçekleştirildiği takdirde anlaşılabilir. Bu sebeple, çalışmanın kapsamı dışında olduğundan ileri araştırma konusu olarak bu noktada bırakılması uygun görülmüştür.

Son değerlendirme aşaması olarak yine burada tüm verileri tek bir ekranda görme amacıyla Evaluator_nP test modülü çıktıları bir "Heatmap" üzerinden aşağıda değerlendirilmiştir.

Şekil 61: Heatmap Sonuçları (Evaluator_nP)



Kaynak: Yazar tarafından derlenmiştir.

Grafikten anlaşıldığı üzere, sol üst köşedeki KATKI ile pozitif yönde en yüksek korelasyona sahip Tekrarsız Hit sayısı olmuştur. Bunu ise yine pozitif olarak sırasıyla, Uygunluk Derecesi, Toplam Terim sayısı ve Tekrarlı Hit sayısı izlemiştir. Diğer bir yandan tekrarlı ya da tekrarsız dokümandaki kelime sayısı arttıkça KATKI üzerinde negatif yönde bir korelasyon göstermiştir. Dolayısıyla buradan hareketle uzun metinlere kıyasla Model_v.1 kelime destek modülü çıktılarının daha kısa alan metinleri üzerinde daha iyi performans göstermekte olduğu söylenebilir.

Buraya kadar yapılan analizler neticesinde Evaluator_nP test modülü çıktılarını göz önünde bulundurarak artık Model_v.1'i etkililik açısından değerlendirebilme amacıyla aşağıdaki sorulara cevap verebilmek mümkündür.

1) Çevirmenler için teknik alana yönelik olarak hazırlanan Türkçe - Japonca kelime destek modülü olan Model_v.1'in ürettiği terimler o alandaki Japonca yazınları okurken kelime bilgisi açısından destek sağlayabilir mi?

   ⇒ **EVET**

2) Model_v.1'in, ürettiği terimlerin o alanda yazılmış Japonca metinler üzerinde ölçümlendirilebilir bir uygunluk derecesi mevcut mu?

   ⇒ **EVET** **(Genel Uygunluk Yüzdesi : % 17.01656629570532)**

3) Model_v.1 modülü kelime çıktıları Japonca akademik yazınlar üzerinde tespit edilebilir ölçüde bir katkı sağladı mı?

   ⇒ **EVET** **(Genel Katkı Yüzdesi : % 3.527075143791614)**

Soru 2 ve 3'ün cevabını aşağıdaki hesaplama ekranından da teyit edilebilir.

Şekil 62: Evaluator_nP'ye Göre Model_v.1'in Uygunluk ve Katkı Oranları

```
In [108]:
data = pd.read_csv("nP_datam1/nP_data[1] _09-05-2023 (Saat 23.57.20).csv")

grd = data.groupby('AnalizDosya').mean().sort_values('KATKI', ascending=False)

genKat = grd[grd.columns[0]].mean()
genUyg = grd[grd.columns[3]].mean()

# Uygunluk Değeri yüzde olarak verilirken formül gereği bölüm sonucu x100'dür.
print("Genel Uygunluk Yüzdesi: %", genUyg*100, "\n")
print("Genel Katkı Yüzdesi   : %", genKat, "\n")


Genel Uygunluk Yüzdesi: % 17.01656629570532

Genel Katkı Yüzdesi   : % 3.527075143791614
```

Kaynak: Yazar tarafından derlenmiştir.

Sonuç itibariyle, Evaluator_nP test modülüne göre de yapılan test ve analiz sonuçlarına bağlı olarak Model_v.1 Modülü çıktıları, değerleri açısından Evaluator_P test modülüne göre daha yüksek oranlarda *Etkililik (effectiveness)* göstererek belirli ölçüde beklentileri karşılar nitelikte olduğu kanaatine varılmıştır.

Diğer bir yandan, yapılan testler neticesinde Model_v.1 kelime destek modülü çıktılarının kısa metinlerde daha iyi performans gösterdiği tespit edildiğinden, aslında başlangıç amacına da uygun olarak sözlü tercümeler için daha uygun olduğu sonucuna varılmıştır. Ancak elbette ki, daha iyi Katkı sonuçları alabilmek için modelin daha da geliştirilmesi gerekmektedir. Aslen şuana kadar yapılan testler neticesinde Model_v.1'in her hangi bir etkililiğinin olup olmadığının değerlendirilmesi, aslında belirli ölçüde de olsa modelin beklentileri karşılayıp karşılamama durumunun sorgulanması amacına sahip olduğundan, bir anlamda Model_v.1'in bundan sonraki aşamalarda geliştirme ve iyileştirmeye amacıyla harcanacak işgücü ve masrafları karşılayabilme kapasitesinin mevcut olup olmadığının da araştırılması niteliğinde olmuştur. Yapılan testler ise, Model_v.1'in bu masrafları kaldırabilir seviyede etkililiğe sahip olduğunu göstermiştir.

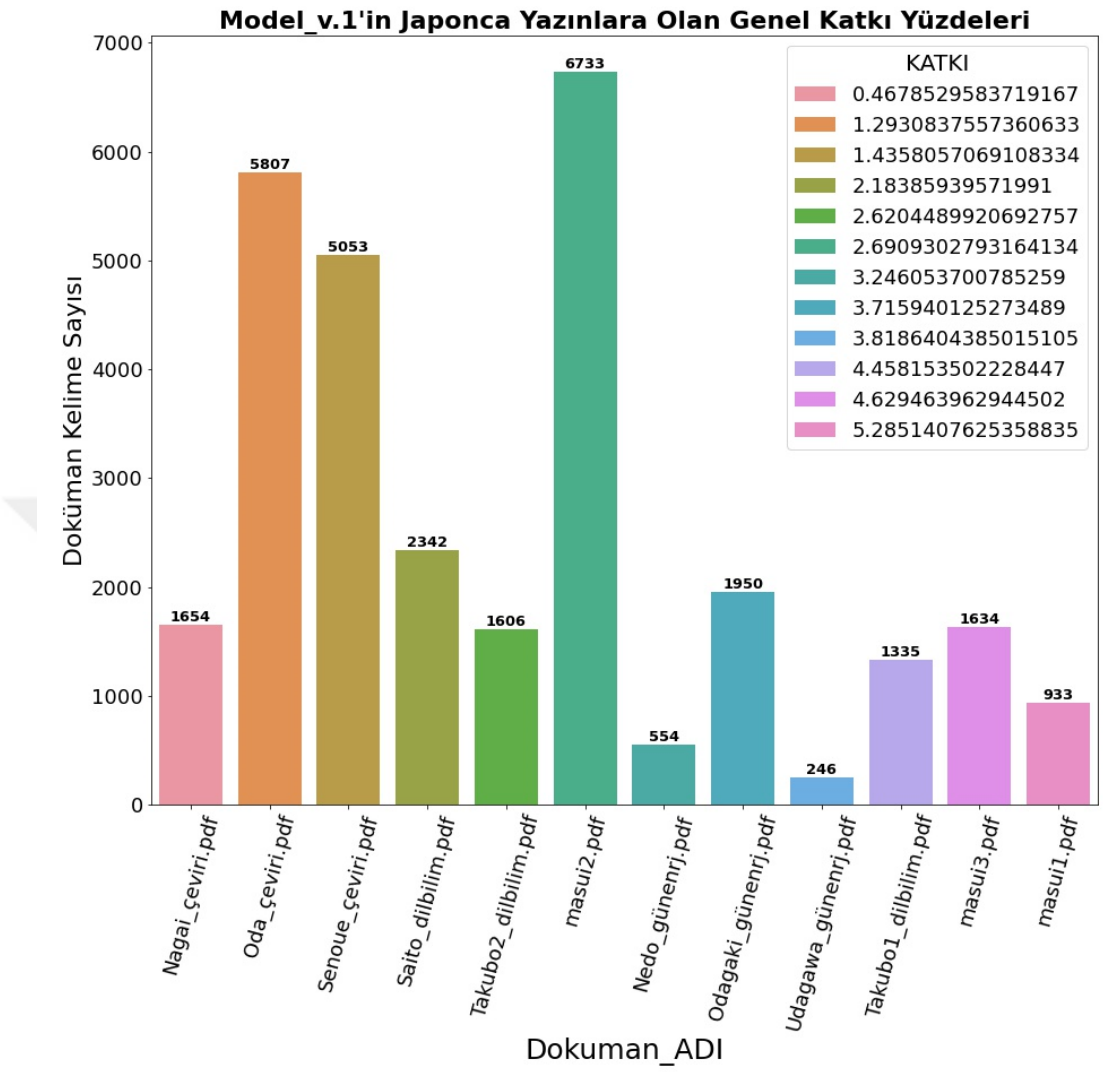
SONUÇ

Çevirmenlik, şüphesiz öncesinde uzun süren bir dil eğitiminin yanı sıra, tecrübe ve emek isteyen bir meslek grubudur. Çevirmenlerin yüzleştiği mesleki kısıtların en başında ise belirli bir sürede bir işin yetiştirilmesi gerekliliği ile alan bilgisi gelir. Bunlardan ikincisi ilkini destekler ve çözüm üretir niteliktedir. Çevirinin konusu olan teknik alandaki tüm kelimelerin, diğer bir deyişle terimsel ifadelerin biliniyor olması çevirmene şüphesiz tamamlayacağı işte hız kazandırır. İşte bu çalışmada, bu nokta mercek altına alınarak, çevirmenler için teknik alana yönelik Türkçe - Japonca kelime destek modülü geliştirilmesi hedeflenmiştir. Yapılan çalışma neticesinde, Python kodları kullanılarak Bilgi Tabanlı ve Derlem Tabanlı hibrit bir model olan Model_v.1 kelime destek modülü geliştirilmiştir.

Geliştirilen bu modülün çıktılarının test edilebilmesi için ise kuralcı ve kuralcı olmayan iki test modülü vasıtasıyla, Model_v.1 kelime çıktıları Japonca yazınlar üzerinden *Etkililik (effectiveness)* açısından teste tabii tutulmuştur. Testler neticesinde elde edilen bulgular ışığında, Model_v.1'in belirli ölçüde alanda etkililiğinin mevcut olduğu sonucuna varılmış ve böylece ilerleyen zamanlarda modelin geliştirilip, eksik yönlerinin giderilmesinin önü açılmıştır.

Öte yandan çalışmada sosyal alana yönelik çeviri ve dilbilim ile sayısal alana yönelik anestezi ve güneşenerjisi şeklinde 4 farklı teknik alan örneklem kümesi olarak seçilerek, her biri ait olduğu alana göre biri uzun, biri orta ve biri kısa olmak üzere toplamda 3 farklı Japonca metin üzerinde test edilmiştir. Hem Evaluator_P üzerinden yapılan testler de, hem de Evaluator_nP üzerinden yapılan testler neticesinde, Model_v.1'in kelime sayısı az olan metinler üzerinde daha iyi sonuç verdiği tespit edilmiştir. Diğer bir yandan bu durumun alandaki toplam terim sayısı ile de bir bağlantısı olduğu görülmüş ve en büyük katkı oranlarına sahip anestezi alanındaki kelimelerin örneklem kümeleri içerisinde en az terime sahip olan küme olması dikkat çekmiştir. Yine bir alandaki terim sayısı ile o terimlerin uygunluk değerleri arasında ise negatif (ters yönlü) bir korelasyonun var olabileceği, Model_v.1'in değerlendirilmesi neticesinde anlaşılmıştır. Ancak çalışmanın sonuçlarına göre anlaşılan bu konunun daha derinlemesine araştırılması gerektiğinden ileri bir araştırma konusu olarak bırakılmıştır.

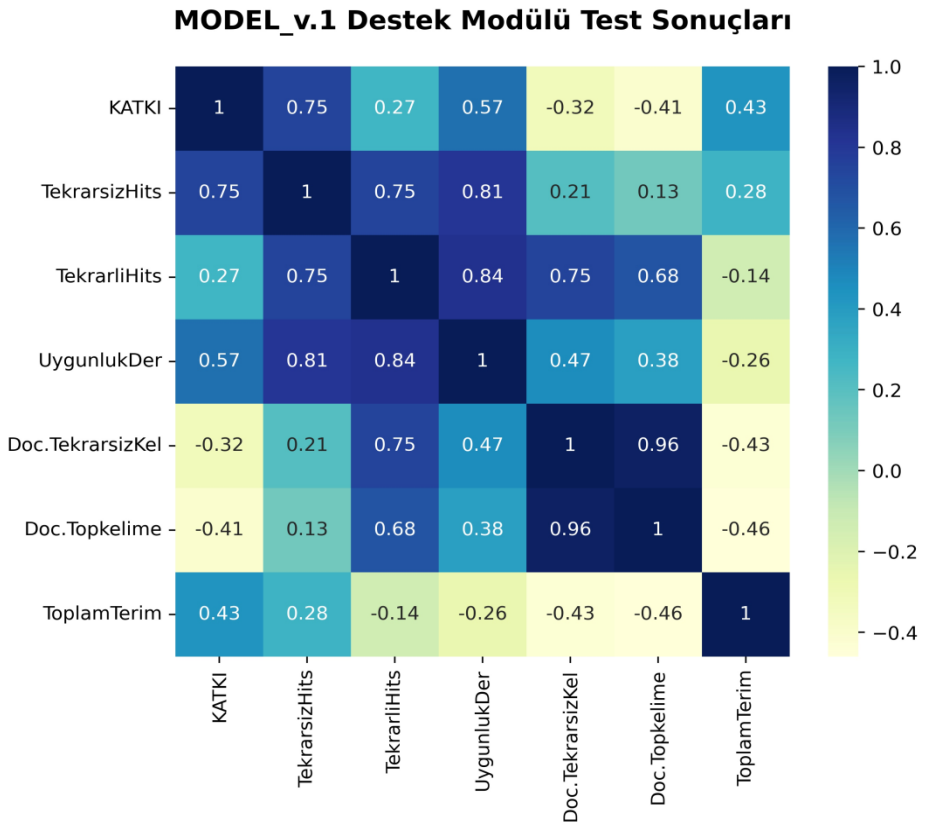
Şekil 63: Model_v.1'in Japonca Yazınlara Olan Genel Katkı Oranı



Kaynak: Yazar tarafından derlenmiştir.

Yukarıda Model_v.1'in genel KATKI oranlarının Japonca yazınlar üzerindeki dağılımları görülmektedir. Buradaki grafikte sadece kelime sayıları ile olan KATKI değişimi gösterilmiştir. Ancak KATKI ya etki eden faktörler arasında doküman kelime sayısının yanı sıra, Model_v.1 kelimelerindeki terim sayısı, tekrarlı ve tekrarsız hit gibi birden fazla faktörün rol oynadığı unutulmamalıdır. Tüm bu faktörlerin tamamının aynı dokümanda görülebilmesi amacıyla hem kuralcı olan hem de kuralcı olmayan test modülünün sonuçlarının tek bir dosyada toplanmış haline dayanan tüm veriler son olarak aşağıda bir Heatmap üzerinden görselleştirilmiştir.

Şekil 64: Model_v.1 Destek Modülü Genel Test Sonuçları



Kaynak: Yazar tarafından derlenmiştir.

Heatmap'te de görüldüğü üzere, kelime destek modülü ile en yüksek pozitif korelasyona sahip olan Tekrarsız Hit sayılarıdır. Diğer bir yandan, Doküman Toplam Kelime sayısının, KATKI ile negatif yönde maksimum korelasyona sahip olması ise, modül çıktılarının kısa metinlerde daha fazla başarı sağlayacağını aktarmaktadır. Yine Toplam Terim sayısı, yani Model_v.1 destek modülü tarafından üretilen alana yönelik kelimelerin sayısındaki artışın da KATKI oranına olumlu yönde etki ettiği görülmektedir. Uygunluk değeri ile KATKI arasında da pozitif yönde bir korelasyon görülmektedir. Ancak Toplam Terim sayısı arttıkça Model_v.1 kelimelerinin uygunluk değerleri ters orantılı olarak düşmektedir.

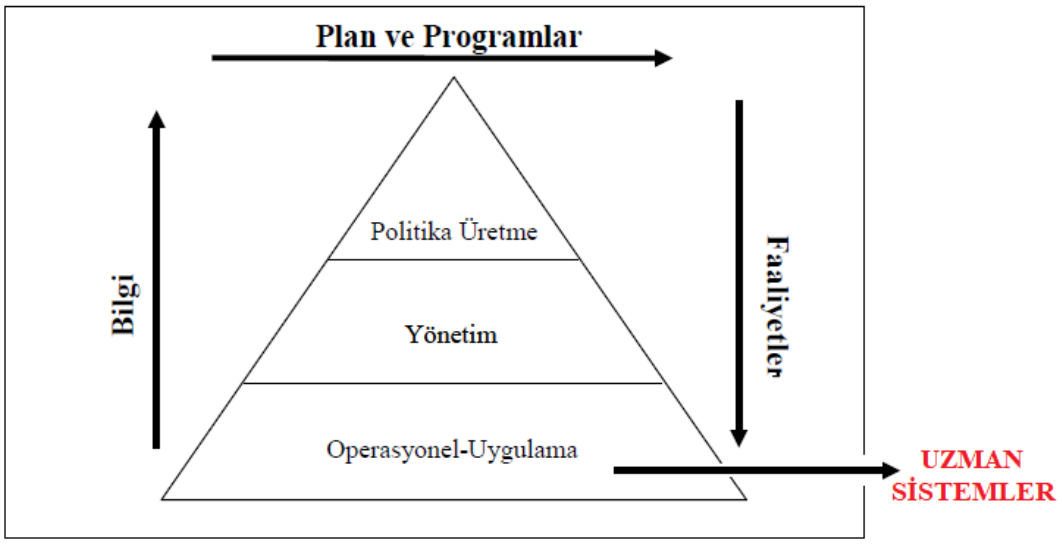
Öte yandan, yapılan çalışma neticesinde, dokümana göre farklılık gösterse de Model_v.1'in çıktıları maksimumda % 5 oranında çevirmene alana yönelik kelimeler hususunda katkı sağlayabilmiştir. Bu oran ilk bakışta küçük bir katkı gibi gözükse de, sözlü ya da yazılı bir alan metnindeki toplam kelimelerin yaklaşık % 30'unun alana yönelik kelimelerden oluştuğu düşünüldüğünde % 5 küçümsenecek bir oran değildir.

Sonuç itibariyle, çalışma kapsamında hazırlanan kelime destek modülü "Model_v.1" tarafından üretilen kelime çıktılarının, Türkçe - Japonca çevirmenlerin teknik alanda karşılaştıkları sıkıntılara belirli ölçüde katkı sağlayabilecek niteliklere sahip olduğu anlaşılmış ve bir sonraki aşamada daha da geliştirilerek çeşitli durum senaryolarının çıktıları üzerinde durulması planlanmıştır.

Ayrıca, yapılan çalışma, Yönetim Bilişim Sistemleri (YBS)'nin alt dalları ve tekniklerinden olan bilgi ve bilişim teknolojileri, yazılım geliştirme, matematiksel ve istatistiksel analiz ile veriyi akıllıca kullanma ve tüm bunları ihtiyaçlar dahilinde bir araya getirip işletme ve bireylerin sorunlarına yönelik çözümler üretebilme kapsamında değerlendirildiğinde ise, yarı akıllı tekniklere dayalı matematiksel ve istatistiksel yöntemlerle kelime işleme neticesinde elde edilen verileri, bilgi teknolojileri ve yazılım geliştirme mantığı ile harmanlayarak hibrit bir şekilde çalışabilen kelime destek modülü elde etmeyi başarabilmiştir.

Firma ve kuruluşların bilişim sistemlerindeki bilgi, genel olarak en alt tabandaki uygulamalardan en üst seviyede bulunan yönetim aşamasına doğru bir akış takip etmektedir ve bunlar aşağıda da görüldüğü şekilde "politika üretme", "yönetim" ve en altta da "operasyonel-uygulama" olarak karşımıza çıkmakta olup, Uzman Sistemler (US) ise en altta operasyonel seviye olarak bilinen uygulama kısmına dahil edilmektedir (Tecim, 2008: 22-25).

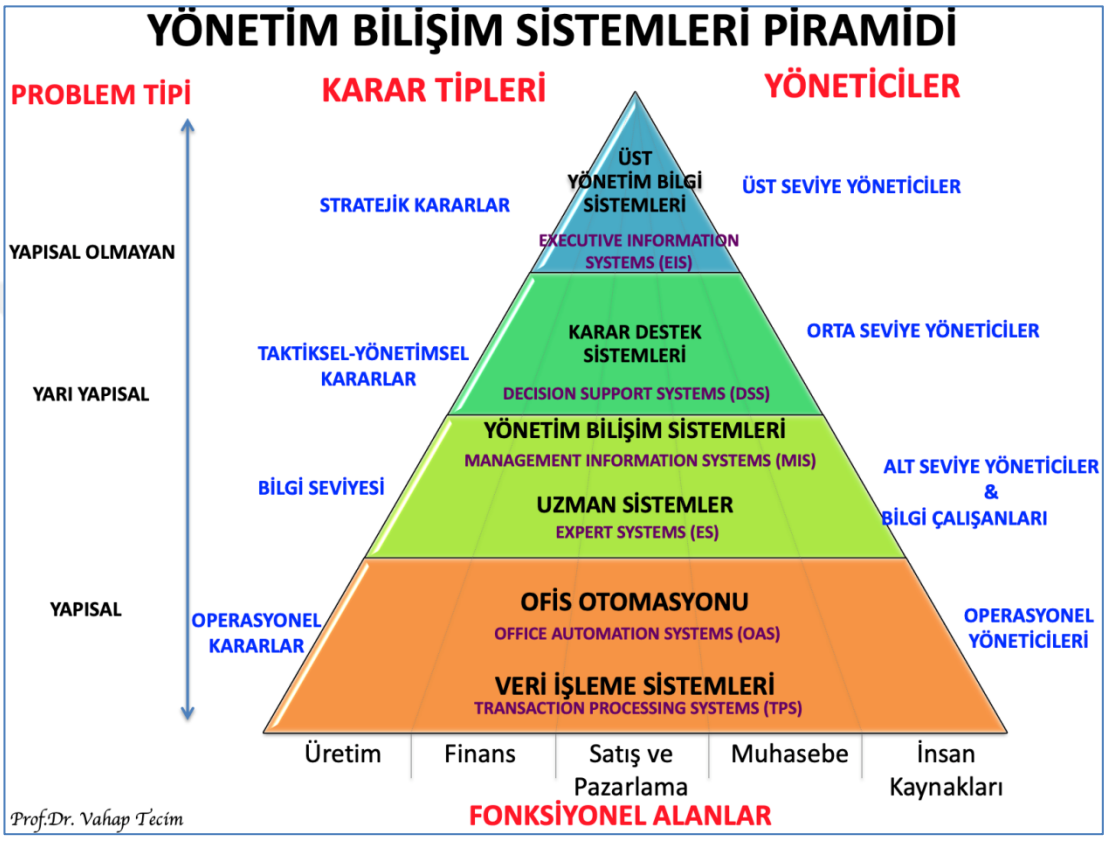
Şekil 65: Bilgi Akışı ve Uzman Sistemler



Kaynak: Tecim(2008: 23).

YBS piramidi açısından değerlendirildiğinde, aşağıda da görüldüğü üzere Uzman Sistemler yine alt seviye grubunda ancak operasyonel seviyenin bir kademe üstünde yer almaktadır.

Şekil 66: Bilgi Hiyerarşisi ve Uzman Sistemler



Kaynak: www.vahaptecim.com.tr/yonetim-bilisim-sistemleri/

Kapanoğlu (2016: 19)'da ise, Uzman Sistemler insanın akıl yürütmesine uygun kuralları, geçerli durum bilgileri ile işleme ve yorumlama yeteneğine sahip bilgi sistemleri olup karar vericinin kararını yönlendirme veya bazı karar aşamalarını yerine getirme gibi olanaklara sahip olduklarından, bu yönleriyle özel bir Karar Destek Sistemi (KDS) olarak da tanımlanmaktadırlar. Dolayısıyla, çeviri alanında oluşturulmuş bir Uzman Sistem olarak değerlendirilebilen Model_v.1, geniş bir çerçeveden bakıldığında, özel bir Karar Destek Sistemi olarak da ele alınabilmektedir.

Tablo 14: KDS Sınıf ve Özellikleri

Tür	Bilgi Sistemleri Derneği	Holsapple ve Whinston	Özelliği
1	İletişim-yönelimli ve grup KDS	(yok)	Grup tabanlı karar verme ve işbirliği iletişimi
2	Veri-yönelimli KDS	Veritabanı-yönelimli KDS	Verileri ve verilerden elde edilebilir çıkarımları görünür ve anlamlı kılma.
3	Belge-yönelimli KDS	Metin-yönelimli KDS	Elektronik veya basılı yazılı belgelerin yönetim ve kullanımı
4	Yöntembilgisi-yönelimli KDS, Veri Madenciliği ve Uzman Sistemler Uygulamaları	Kural-yönelimli KDS	Yapay zekâ yöntemleri, veri madenciliği uygulamaları veya dar bir alanda probleme özel kural ve yöntemlerin kullanımı
5	Model-yönelimli KDS	Çözücü-yönelimli KDS	Optimizasyon, simülasyon veya algoritma kullanımı
6	(yok)	İşlemtablosu-yönelimli KDS	İşlemtablosunun sunduğu modelleme, programlama, veri yönetimi, dosya paylaşımı ve kurumsal bilinirliğinden yararlanma
7	Karma KDS	Birleşik KDS	Farklı türlerin özelliklerinden yararlanma

Kaynak: Kapanoğlu (2016: 17).

Ancak hangi kapsamda ele alınırsa alınsın, Model_v.1'in dar bir alanda çözüm ürettiği kısmı değişmemektedir. Uzman sistemlerin en belirgin ve belki de diğer sistemlerden ayıran en önemli özellikleri, dar bir alandaki belirli bir sorunu çözmeye odaklamalarının yanı sıra, mevcutta sorunun uzmanlık bilgisi gerektiriyor olmasıdır. Bu açıdan bakıldığında Model_v.1'in Japonca alanındaki teknik alana yönelik terimleri girilen anahtar kelimeden yola çıkarak bulup getirmesi de bu kapsamda ele alınabilmektedir.

Normal şartlarda, Japonca bilgisinin yanı sıra, girilen anahtar kelimenin ait olduğu alan bilgisine de ihtiyaç duyulduğu bir durum olmasına rağmen, geliştirilen Model_v.1 sayesinde uzmanlık bilgisi, bilişim teknolojileri ile yazılım tekniklerinin harmanlanması neticesinde elde edilerek, o alana ilişkin herhangi bir uzman görüşüne gereksinim duyulmadan Türkçe-Japonca alanında çevirmenler için kalıcı, hızlı, maliyetsiz ve gelişmiş çözümler elde edilebilmiştir.

KAYNAKÇA

Adalı, E. (2020). *Türkçe Doğal Dil İşleme*. Ankara: Akçağ Yayınları.

Aksoy, N. B. (1998). Teknik Çeviri. *Hacettepe Üniversitesi Edebiyat Fakültesi Dergisi*. Cilt 15(2) : 71 - 80.

Anamur, H. (1997). Terimbilim Sorunları ve Bir Çözüm Önerisi. *Forum: Türkiye'de Çeviri Eğitimi – Nereden Nereye?* (39 - 44). B.Ö. Işın (Edt.), İstanbul: Sel Yayıncılık.

Aydın, G. ve Hallaç, İ. R. (2021). Türkçe Metinlerde Otomatik Konu Tespiti. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, Vol.33(2): 599-606. https://doi.org/10.35234/fumbd.899917

Bansal, M., Gimpel, K., ve Livescu, K. (2014). "Tailoring Continuous Word Representations for Dependency Parsing". *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol.2: 809-815, Baltimore, Maryland. https://aclanthology.org/P14-2131.pdf

Blei, D. M., Ng, Andrew Y. ve Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, Vol.3: 993-1022. https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Brang P. (1963). Das Problem der Übersetzung in sowjetischer Sicht. *Das Problem des Übersetzens* (pp. 384 - 401). H.J. Störig (Edt), Darmstadt: Wissenschaftliche Buchgesellschaft.

Chethan Kumar, G.N. (Sep 25, 2018). "NLP vs NLU vs NLG (Know what you are trying to achieve) NLP engine (Part-1)", Published in Towards Data Science – Medium. https://towardsdatascience.com/nlp-vs-nlu-vs-nlg-know-what-you-are-trying-to-achieve-nlp-engine-part-1-1487a2c8b696, (29/01/2023).

Church, K.W. ve Hanks, P. (1990). Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, Vol.16(1): 22-29. https://aclanthology.org/P89-1010.pdf

Daniel J. ve James H. M. (September 19, 2018). "Chapter C - WordNet: Word Relations, Senses, and Disambiguation", Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/old_oct19/C.pdf , (18/12/2022).

Deerwester S. C., Dumais S. T., Furnas G. W., Harshman R. A., Landauer T. K., Lochbaum K. E. ve Streeter L. A. (1988). *Computer information retrieval using latent semantic structure* (United States Patent No: US4839853A), Bell Communications Research, Inc., Livingston, N.J., USA. https://patents.google.com/patent/US4839853A/en

Doğan, A. (2014). *Sözlü ve Yazılı Çeviri Odaklı Söylem Çözümlemesi*. Ankara: Siyasal Kitabevi.

Dharma, E. M., Gaol, F. L., Warnars, H. L. H. S. ve Soewito, B. (2022). The Accuracy Comparison among Word2vec, Glove, and Fasttext towards Convolution Neural Network (CNN) Text Classification. Journal of Theoretical and Applied Information Technology (JATIT), Vol.100(2): 349-359, Islamabad, PAKISTAN. http://www.jatit.org/volumes/Vol100No2/5Vol100No2.pdf

Erk, K. (2012). Vector Space Models of Word Meaning and Phrase Meaning: A Survey. *Language and Linguistics Compass*, Vol.6: 635-653. https://doi.org/10.1002/lnco.362

Eruz-Esen, S. (2008). *Akademik Çeviri Eğitimi: Çeviri Amaçlı Metin Çözümlemesi*. İstanbul: Multilingual.

European Communities (2006). *Evaluation Methods For The European Union's External Assistance - Methodological Bases For Evaluatıon - Volume 1*. Directorate General External Relations - EuropeAid Co-operation Office, ISBN: 92-79-00681-9, France. https://www.oecd.org/dac/evaluation/dcdndep/47469160.pdf

Fano, R.M. (1961). *Transmission of Information: A Statistical Theory of Communication*. The MIT Press, Cambridge.

Firth, J. (1957). A Synopsis of Linguistic Theory, 1930-1955. In Studies in Linguistic Analysis (pp. 1-31). *Special Volume of the Philological Society*. Oxford: Blackwell. https://cs.brown.edu/courses/csci2952d/readings/lecture1-firth.pdf

Frauenfelder, M. (November 1, 2001). "A Smarter Web", MIT Technology Review. https://www.technologyreview.com/2001/11/01/235407/a-smarter-web-3/, (31/12/2022).

Goddard, C. (1998). *Semantic Analysis: A Practical Introduction.* NY: Oxford University Press.

Göktürk, A. (1994). *Çeviri: Dillerin Dili*. İstanbul: Yapı Kredi Yayınları.

Gutmann, M. ve Hyvärinen, A. (2012). Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *Journal of Machine Learning Research*, Vol.13: 307-361. https://www.jmlr.org/papers/volume13/gutmann12a/gutmann12a.pdf

Günther, F., Rinaldi, L. ve Marelli, M. (2019). Vector-Space Models of Semantic Representation From a Cognitive Perspective: A Discussion of Common Misconceptions. *Perspectives on Psychological Science*, Vol.14(6): 1006–1033. https://doi.org/10.1177/1745691619861372

Harris, Z. S. (1954). Distributional Structure. *WORD*, Vol.10(2-3): 146–162. https://www.tandfonline.com/loi/rwrd20

Haspolat, T (PathAwayer). (Dec 13, 2018). “Ontoloji Bilimi”, Published in Patikan – Medium. https://medium.com/patikan/ontoloji-bilimi-b3b26cbe6975, (17/12/2022).

Itoh, H. (2000). Using Ontology for Knowledge Sharing and Reuse. *Unisys Technology Review*, Vol.64: 115-129 (617-631) , Feb. (伊藤英毅 (2000). オントロジーを利用した知識の共有／再利用, *Unisys Technology Review*, 第 64 号: 115-129 (pp.617-631)). https://pr.biprogy.com/solution/biz/bi/s33drt000007btf5-att/6407.pdf

İmer, K., Kocaman, A. ve Özsoy, S. (2011). *Dilbilim Sözlüğü*. İstanbul: Boğaziçi Üniversitesi Yayınevi.

Jones, M. N. ve Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, Vol.114(1): 1-37. https://doi.org/10.1037/0033-295X.114.1.1

Kapanoğlu, M. (2019). *Karar Destek Sistemleri*. T.C. Anadolu Üniversitesi Yayını No: 3458, Anadolu Üniversitesi, Eskişehir.

Kavlakoğlu, E. (Nov 12, 2020). “NLP vs. NLU vs. NLG: the differences between three natural language processing concepts”, Published in IBM – Watson Blog. https://www.ibm.com/blogs/watson/2020/11/nlp-vs-nlu-vs-nlg-the-differences-between-three-natural-language-processing-concepts/, (29/01/2023).

Kilmen, S. ve Bulut, O. (May 2, 2022). “Okan Bulut: Text Vectorization Using Python: Word2Vec”. *Okan Bulut (Kişisel Websitesi)*. Retrieved from: https://okan.cloud/posts/2022-05-02-text-vectorization-using-python-word2vec/

Köksal, D. (2008). *Çeviri Eğitimi: Kuram ve Uygulama*. Ankara: Nobel Yayın Dağıtım.

Lakoff, G. (1987). *Women, Fire, and Dangerous Things: What Categories Reveal About The Mind*. Chicago: University of Chicago Press.

Landauer, T. K., ve Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, Vol.104(2): 211-240. https://doi.org/10.1037/0033-295X.104.2.211

Lenci, A. (2008). Distributional semantics in linguistic and cognitive research. *Rivista di Linguistica*, Vol.20(1): 1-31. https://www.italian-journal-linguistics.com/app/uploads/2021/05/1_Lenci.pdf

Levy, O. ve Goldberg, Y. (2014). "Dependency-Based Word Embeddings". *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 2: 302–308, Baltimore, Maryland. https://aclanthology.org/P14-2050.pdf

Lowe, W. (2001). Towards a Theory of Semantic Space. *Proceedings of the Annual Meeting of the Cognitive Science Society*, Vol.23. Retrieved from https://escholarship.org/uc/item/0wk159m0

Lund, K. ve Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments & Computers*, Vol.28(2): 203-208. https://link.springer.com/content/pdf/10.3758/BF03204766.pdf

Manning, C. (Winter, 2019). "Lecture 1 – Introduction and Word Vectors". *Standford University – Stanford CS224N: NLP with Deep Learning - Winter 2019 (Online Courses)*. https://www.youtube.com/watch?v=8rXD5-xhemo

Markman, Ellen M. (1991). *Categorization and Naming in Children*. Cambridge: MIT Press.

Melli, G. (June 28, 2021). “Hyperspace Analogue to Language Algorithm”, Gabor Melli’s Research Knowledge Base. http://www.gabormelli.com/RKB/Hyperspace_Analogue_to_Language_Algorithm, (21/02/2023)

Mikolov, T., Chen, K., Corrado, G. ve Dean, J. (2013a). “Efficient Estimation of Word Representations in Vector Space”. *International Conference on Learning Representations - ICLR Workshop*, (pp.1-12). https://doi.org/10.48550/arXiv.1301.3781

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. ve Dean, J. (2013b). “Distributed representations of words and phrases and their compositionality”. *Advances in neural information processing systems 26 (NIPS 2013) - NeurIPS Proceedings 2013*, (pp. 3111-3119) (pp.1-9). https://arxiv.org/abs/1310.4546

Miller, G. A. (1985). “Wordnet: A Dictionary Browser”, *First conference of the UW Centre for the New Oxford English Dictionary : Information in Data - Proceedings of the Conference* (pp. 25 - 34), *November 6-7, 1985, Waterloo, Canada*. Waterloo, Canada: University of Waterloo.

Miller, G.A., Beckwith, R., Fellbaum, C.D., Gross, D. ve Miller, K.J. (1990). “Introduction to WordNet: An On-line Lexical Database”. *International Journal of Lexicography*, Vol. 3: 235-244. https://wordnetcode.princeton.edu/5papers.pdf

Mitchell, J. ve Lapata, M. (2010), Composition in Distributional Models of Semantics. *Cognitive Science*, Vol.34: 1388-1429. https://doi.org/10.1111/j.1551-6709.2010.01106.x

MLNerds (14 Feb, 2019). “What is the difference between word2Vec and Glove ?”. *Machine Learning Interviews - Tools to crack your data science Interviews*. https://machinelearninginterview.com/topics/natural-language-processing/what-is-the-difference-between-word2vec-and-glove/ (19/03/2023)

Mnih, A ve Teh, Y. (2012). "A fast and simple algorithm for training neural probabilistic language models". *Proceedings of the 29th International Conference on Machine Learning - ICML 2012*, Vol.2: 1751-1758, Edinburgh, Scotland, UK, 2012. https://www.cs.toronto.edu/~amnih/papers/ncelm.pdf

Mnih, A. ve Hinton, G. (2008) "A scalable hierarchical distributed language model." *Advances in neural information processing systems 21 (NIPS 2008) - Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*, Vancouver, British Columbia, Canada, December 8-11, 2008. https://www.cs.toronto.edu/~amnih/papers/hlbl_final.pdf

Morin, F ve Bengio, Y. (2005) "Hierarchical probabilistic neural network language model." *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, PMLR - R5 (pp.246-252), 2005. https://www.iro.umontreal.ca/~lisa/pointeurs/hierarchical-nnlm-aistats05.pdf

Murdock, B. B. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review*, Vol.89(6): 609-626. https://doi.org/10.1037/0033-295X.89.6.609

Önal, Ö. (2019). Geç Çocukluk Dönemi Anadili Ediniminde Taksonomik Ulamlaştırma Becerileri. *DTCF Dergisi*, Cilt:59(1): 662 – 692. https://dergipark.org.tr/en/download/article-file/2153821

Özdoğan, A. G. ve Turan, M. (2019). Metin Madenciliği Kullanarak İngilizce Doküman Sınıflama. *İstanbul Ticaret Üniversitesi Teknoloji ve Uygulamalı Bilimler Dergisi*, Vol.2(1): 37-46. https://dergipark.org.tr/tr/pub/icujtas/issue/57146/806504

Pennington, J., Socher, R. ve Manning, C. (2014). “GloVe: Global Vectors for Word Representation”. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1532–1543), Association for Computational Linguistics, Doha, Qatar. https://aclanthology.org/D14-1162.pdf

Perfetti, C. A. (1998). The limits of co-occurrence: Tools and theories in language research. Discourse Processes, Vol.25: 363-377. https://www.tandfonline.com/doi/abs/10.1080/01638539809545033

Guiraud, P. (1999). *Anlambilim - La Sémantique*. (B. Vardar, Çev.). İstanbul: Multilingual Yayınları.

Reiss, K. (1971). *Möglichkeiten und Grenzen der Übersetzungskritik: Kategorien und Kriterien für eine sachgerechte Beurteilung von Übersetzungen*. München: Hueber Verlag.

Reiss, K. (1983). *Texttyp und Übersetzungsmethode: der operative Test*. Heidelberg: J. Gross Verlag.

Rigutini L., Maggini, M. ve Liu, B. (2005). “An EM based training algorithm for Cross-Language Text Categorization”. *WI '05: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 529-535). Publisher: IEEE Computer Society. September 19 - 22, 2005. https://www.cs.uic.edu/~liub/publications/WI-05-crossLang.pdf

Rinaldi, F. ve Yuste, E. (2004). “Exploiting Technical Terminology for Knowledge Management”, *Proceedings of the EKAW 2004 Workshop on Application of Language and Semantic Technologies to support Knowledge Management Processes*, Application of Language and Semantic Technologies to support Knowledge Management Processes (LSTKM 2004), Whittlebury Hall, 8th October, 2004. https://ceur-ws.org/Vol-121/02.pdf

Robyn, S. ve Catherine, H. (2012). “Representing General Relational Knowledge in ConceptNet 5”. *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)* (pp. 3679–3686), Istanbul, Turkey. European Language Resources Association (ELRA). https://aclanthology.org/L12-1639/

Robyn, S., Joshua, C. ve Catherine, H. (2017). “ConceptNet 5.5: An Open Multilingual Graph of General Knowledge”. *Proceedings - Thirty-First AAAI Conference on Artificial Intelligence (Proceedings of* AAAI *31)*, Vol: 31(1): 4444 – 4451. https://ojs.aaai.org/index.php/AAAI/article/view/11164

Sarı, V. İ. (2010). *Proje Döngüsü Yönetiminde Etkililik Değerlendirmesi*. Devlet Planlama Teşkilatı, ISBN 975-19-4772-7, ANKARA: Korza Yayıncılık. https://www.sbb.gov.tr/wp-content/uploads/2022/08/Proje-Dongusu-Yonetiminde-Etkililik-Degerlendirmesi-Volkan-Idris-Sari.pdf

Sarkar, K., ve Law, R. (2015). A Novel Approach to Document Classification using WordNet. *ArXiv*. https://doi.org/10.48550/arXiv.1510.02755

Serafin, R., ve Eugenio, B. D. (2004). “FLSA: Extending latent semantic Analysis with features for dialogue act classification”. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics* (pp. 692–699). East Stroudsburg, PA: Association for Computational Linguistics. https://aclanthology.org/P04-1088.pdf

Serrano, L. (March 16, 2020). “Latent Dirichlet Allocation (Part 1 of 2)”. Serrano.Academy (YouTube Channel). https://www.youtube.com/watch?v=T05t-SqKArY (26/02/2023).

Socher, R. ve Manning, C. (3 Apr, 2017). "Natural Language Processing with Deep Leraning CS224N/Ling284 – Lecture 3 | GloVe: Global Vectors for Word Representation". Standford University School of Engineering - Lecture Collection | Natural Language Processing with Deep Learning (Winter 2017) (Online Course). https://www.youtube.com/watch?v=ASn7ExxLZws

StackExchange (Nov 11, 2016). "How does negative sampling work in word2vec?". (Answer by "turdus-merula") https://stats.stackexchange.com/questions/244616/how-does-negative-sampling-work-in-word2vec (16/03/2023)

Su Nam K. ve Lawrence C. (2011). "Classifying Domain-Specific Terms Using a Dictionary". *Proceedings of Australasian Language Technology Association Workshop* (pp. 57 – 65), Australian National University, Australasian Language Technology Association Workshop 2011, Canberra – Australia. 1-2 December 2011. https://aclanthology.org/U11-1009.pdf

Suárez, A. ve Palomar, M. (2002). "A Maximum Entropy-based Word Sense Disambiguation system". *COLING '02: Proceedings of the 19th international conference on Computational linguistics*, Vol.1: 1-7, August 2002. https://doi.org/10.3115/1072228.1072343

Sübay, M.T. (2019). *Türkçe Kelime Vektörlerinde Görülen Anlamsal ve Biçimsel Yakınlaşmalar*. (Yayınlanmamış Yüksek Lisans Tezi). İstanbul: Maltepe Üniversitesi Fen Bilimleri Enstitüsü.

Tabernig de Pucciarelli, E. (1964). Aspectos tecnicos y literarios de la traduccion. *Boletin de Estudios germánicos*. Vol. 5: 137-155.

Taylor, John R. (1995). *Linguistic Categorization: Prototypes in Linguistic Theory (Second Edition)*. Oxford: Oxford University Press.

Tecim, V. (2008). *Coğrafi Bilgi Sistemleri - Harita Tabanlı Bilgi Yönetimi*. Dokuz Eylül Üniversitesi, Fen Bilimleri Enstitüsü, Coğrafi Bilgi Sistemleri Anabilim Dalı, ISBN 978 – 605-60047-0-4, Renk Form Ofset Matbaacılık.

Topuz, M. (16 Ağustos, 2017). "n-gram Nedir? Neden Kullanılır?". Üniversite Veri Madenciliği. https://merttopuz.com/universite/veri-madenciligi/n-gram-nedir-neden-kullanilir (07/03/2023)

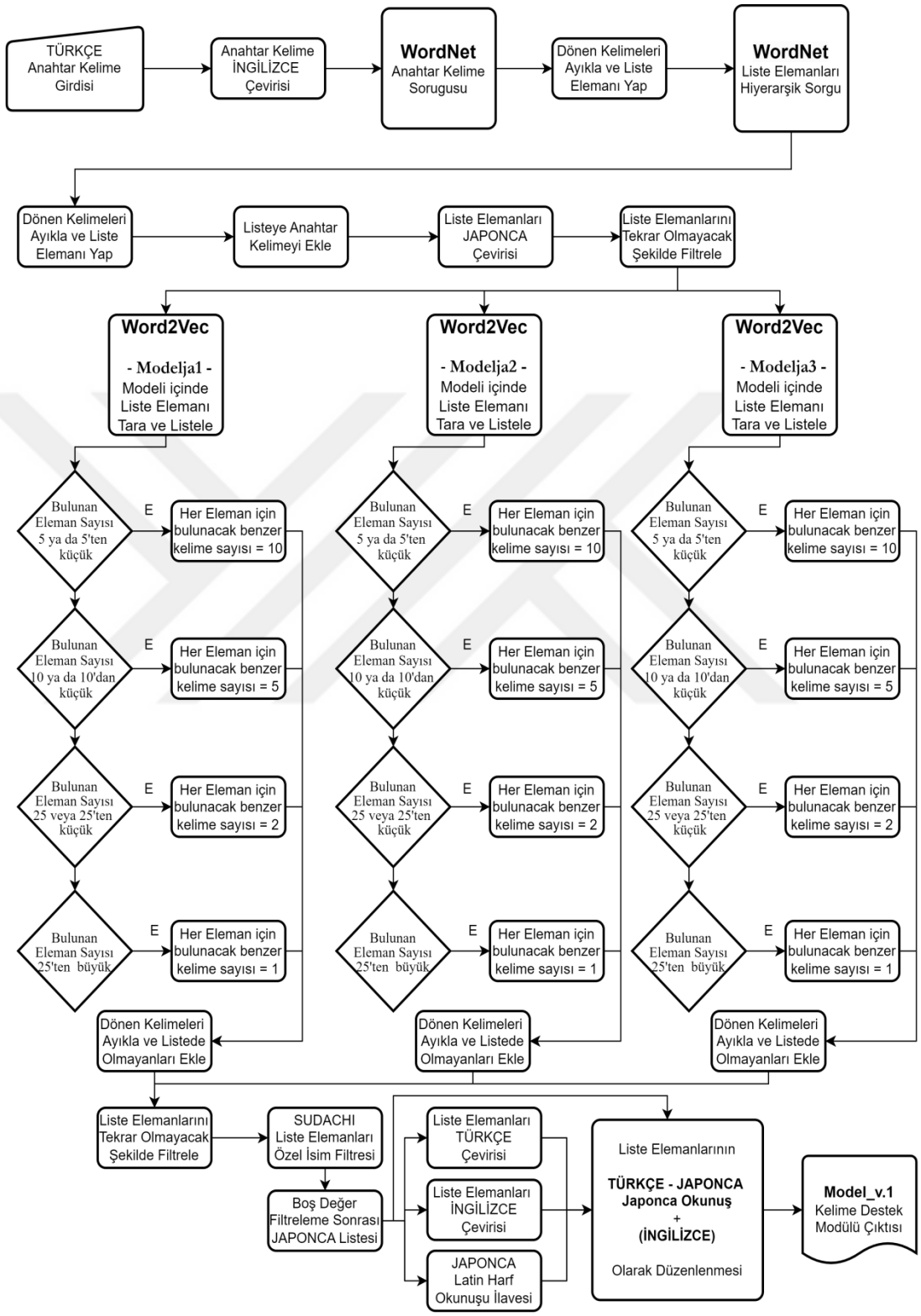
Tsujii, J. (2021). Natural Language Processing and Computational Linguistics. *Computational Linguistics*, Vol. 47(4): 707–727. https://direct.mit.edu/coli/article-pdf/47/4/707/1979478/coli_a_00420.pdf

Ülgen, G. (2004). *Kavram Geliştirme.* Ankara: Nobel Yayıncılık.

Wawer, A. (2012). "Mining Co-Occurrence Matrices for SO-PMI Paradigm Word Candidates". *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics (pp. 74–80), Avignon, France. https://aclanthology.org/E12-3009.pdf

EKLER

EK1: Model_v.1 UML Diagramı



EK2: Model_v.1 Destek Modülü Kelime Çıktıları – “çeviri1.txt”

çeviri1.txt - Notepad

File Edit Format View Help

```
1.) çeviri = 翻訳  honyaku (translation)
2.)  genlik = 振幅 shinpuku ( amplitude)
3.)  dikkat dağıtma = 気晴らし kibarashi ( distraction)
4.)  yer değiştirme = 脱臼 dakkyuu ( dislocation)
5.)  dönüştürme = 変容 henyou ( transformation)
6.)  yanlış çeviri = 誤訳 goyaku ( mistranslation)
7.)  modülasyon = 変調 henchou ( modulation)
8.)  permütasyon = 順列 junretsu ( permutation)
9.)  midilli = ポニー ponii ( pony)
10.)  nitelik = 資格 shikaku ( qualification)
11.)  yeniden düzenleme = 再編成 saihensei ( rearrangement)
12.)  retroversiyon = レトロバージョン retorobaajon ( retroversion)
13.)  revizyon = 改訂 kaitei ( revision)
14.)  kaydırma = シフト shifuto ( shift)
15.)  subluksasyon = サブラクセーション saburakuseeshon ( subluxation)
16.)  altyazı = サブタイトル sabutaitoru ( subtitle)
17.)  üstyazı = スーパータイトル suupaataitoru ( supertitle)
18.)  transmogifikasyon = トランスモグリフィケーション toransumogurifikeeshon ( transmogification)
19.)  çevirmen = 翻訳者 honyakusha ( translator)
20.)  AWGN = AWGN awgn ( AWGN)
21.)  kepek = ふけっ fuketsu ( dandruff)
22.)  Heyecan verici = 興ずる kyouzuru ( Exciting)
23.)  Kırık = 骨折 kossetsu ( Fracture)
24.)  Burkulma = 捻挫 nenza ( Sprain)
25.)  Yönler = 諸相 shosou ( Aspects)
26.)  Perspektif = 視座 shiza ( Perspective)
27.)  Çeviri = 訳し yakushi ( Translation)
28.)  Japonca çevirisi = 和訳 wayaku ( Japanese translation)
29.)  Modülasyon yöntemi = 変調方式 henchouhoushiki ( Modulation method)
30.)  Taşıyıcı dalga = 搬送波 hansouha ( Carrier wave)
31.)  Düzen = 順序 junjo ( Order)
32.)  Karşılaştırma yöntemi = 比較方法 hikakuhouhou ( Comparison method)
33.)  Buena Vista = ブエナ・ビスタ buena・bisuta ( Buena Vista)
34.)  Ulusal yeterlilik = 国家資格 kokkashikaku ( National qualification)
35.)  Gözden geçirilmiş baskı = 改訂版 kaiteiban ( Revised version)
36.)  Ek = 増補 zouho ( Supplement)
37.)  Anahtarlama = 切り替え kirikae ( Switching)
38.)  Valf kaldırma = バルブリフト baruburifuto ( Valve lift)
39.)  Altyazı = 副題 fukudai ( Subtitle)
40.)  İngilizce çevirisi = 英訳 eiyaku ( English translation)
41.)  Frekans = 振動数 shindousuu ( Frequency)
42.)  Mood dönüştürme = 気分転換 kibuntenkan ( Mood conversion)
43.)  keyif = 楽しみ tanoshimi ( enjoyment)
44.)  değiştirme = 変質 henshitsu ( alteration)
45.)  değiştirme = 変化 henka ( change)
46.)  yanlış baskı = 誤植 goshoku ( misprint)
47.)  büyütme = 増幅 zoufuku ( amplification)
48.)  nota değeri = 音価 onka ( note value)
49.)  penguen = ペンギン pengin ( penguin)
50.)  kedi = キャット kyatto ( cat)
51.)  revizyon = 改定 kaitei ( revision)
52.)  düzeltme = 修正 shuusei ( correction)
53.)  vardiya = 移行 ikou ( shift)
54.)  vardiya değişikliği = シフトチェンジ shifutochenji ( shift change)
55.)  yorumlama = 通訳 tsuuyaku ( interpretation)
56.)  öteleme = 訳文 yakubun ( translation)
57.)  dalga biçimi = 波形 hakei ( waveform)
58.)  yer değiştirme = 変位 hen'i ( displacement)
59.)  nefes alma = 息抜き ikinuki ( relaxation)
60.)  kabartma = 憂さ晴らし usaharashi ( relief)
61.)  başkalaşım = 変貌 henbou ( transfiguration)
62.)  değişiklik = 変革 henkaku ( transformation)
63.)  karışıklık = 迷訳 meiwake ( confusion)
64.)  modem = 変復調 henfukuchou ( modem)
65.)  fazlama = フェージング feejingu ( phasing)
66.)  çift-tek = 偶奇 guuki ( even-odd)
67.)  hisamitsu = ヒサミツ hisamitsu ( hisamitsu)
68.)  midilli = ポニー・ ponii・ ( pony)
69.)  koşullar = 条件 jouken ( conditions)
70.)  gereksinimler = 要件 youken ( requirements)
71.)  uzun giriş versiyonu = ロングイントロバージョン ronguintorobaajon ( long intro version)
72.)  nadir versiyon = レアバージョン reabaajon ( rare version)
73.)  shimano grip = シマノグリップ shimanogurippu ( shimano grip)
74.)  al-hauz = アルハウズ aruhauzu ( al-hauz)
75.)  subluksasyon = サブラクセイション saburakuseishon ( subluxation)
76.)  fiksasyon = フィクセーション fikuseeshon ( fixation)
77.)  başlık = タイトル taitoru ( title)
```

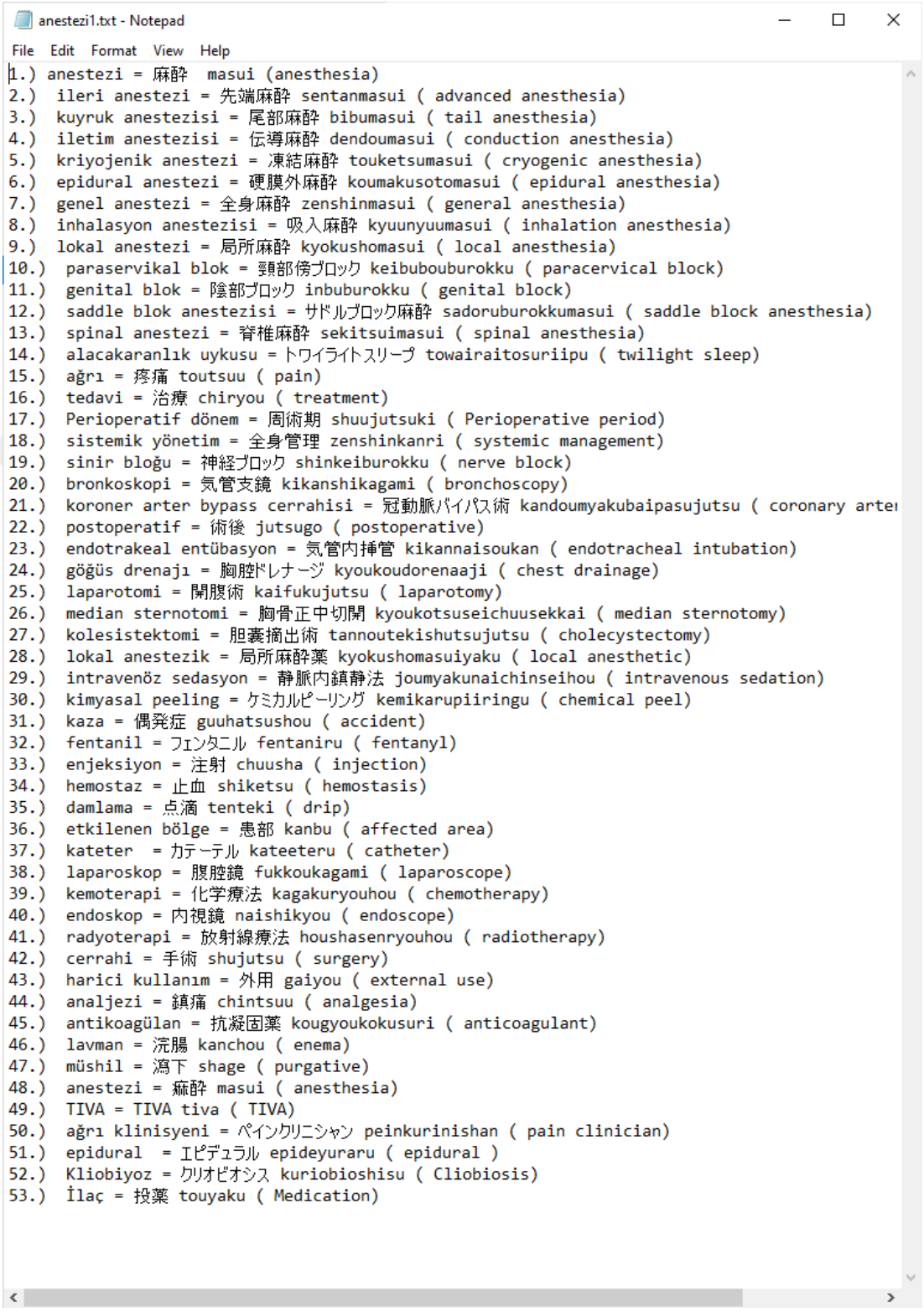
EK3: Model_v.1 Destek Modülü Kelime Çıktıları – “dilbilim9.txt”

dilbilim9.txt - Notepad

File Edit Format View Help

```
1.) Dilbilim = 言語学  gengogaku (Linguistics)
2.)  hesaplamalı dilbilim = 計算言語学 keisangengogaku ( computational linguistics)
3.)  tanımlayıcı dilbilim = 記述言語学 kijutsugengogaku ( descriptive linguistics)
4.)  lehçe coğrafyası = 方言地理学 hougenchirigaku ( dialect geography)
5.)  etimoloji = 語源学 gogengaku ( etymology)
6.)  tarihsel dilbilim = 歴史言語学 rekishigengogaku ( historical linguistics)
7.)  nörolinguistik = 神経言語学 shinkeigengogaku ( neurolinguistics)
8.)  edimbilim = 語用論 goyouron ( pragmatics)
9.)  normatif dilbilim = 規範言語学 kihangengogaku ( normative linguistics)
10.)  anlambilim = 意味論 imiron ( semantics)
11.)  toplumdilbilim = 社会言語学 shakaigengogaku ( sociolinguistics)
12.)  yapısalcılık = 構造主義 kouzoushugi ( structuralism)
13.)  eşzamanlı dilbilim = 共時言語学 kyoujigengogaku ( synchronic linguistics)
14.)  lehçeler bilimi = 方言学 hougengaku ( dialects science)
15.)  sözlükbilim = 語彙学 goigaku ( lexicology)
16.)  makine çevirisi = 機械翻訳 kikaihonyaku ( machine translation)
17.)  gramer = 文法 bunpou ( grammar)
18.)  morfofonetik = 形態音学 keitaiongaku ( morphophonetics)
19.)  fonoloji = 音韻論 on'inron ( phonology)
20.)  sözcüksel istatistikler = 語彙統計学 goitoukeigaku ( lexical statistics)
21.)  bilişsel anlambilim = 認知意味論 ninchiimiron ( cognitive semantics)
22.)  doğrudan dil = 直語 chokugo ( direct language)
23.)  biçimsel anlambilim = 形式的意味論 keishikitekiimiron ( formal semantics)
24.)  sözcüksel anlambilim = 語彙的意味論 goitekiimiron ( lexical semantics)
25.)  toponimi = トポニミー toponimii ( toponymy)
26.)  tanımlayıcı dilbilgisi = 記述文法 kijutsubunpou ( descriptive grammar)
27.)  morfoloji = 形態学 keitaigaku ( morphology)
28.)  kuralcı dilbilgisi = 規範文法 kihanbunpou ( prescriptive grammar)
29.)  sözdizimi = 構文 koubun ( syntax )
30.)  ortopi = オルソピー orusopii (orthopy)
31.)  karmaşık morfoloji = 複合形態学 fukugoukeitaigaku (composite morphology)
32.)  türetilmiş morfoloji = 派生形態学 haseikeitaigaku (derivative morphology)
33.)  çekimsel morfoloji = 屈折形態学 kussetsukeitaigaku (inflectional morphology)
34.)  üretken dilbilgisi = 生成文法 seiseibunpou (generative grammar)
35.)  karşılaştırmalı dilbilim = 比較言語学 hikakugengogaku (comparative linguistics)
36.)  felsefi dil = 哲学的言語 tetsugakutekigengo (philosophical language)
37.)  ortaçağ Latincesi = 中世ラテン語 chuuseiratengo (medieval Latin)
38.)  konuşma eylemi = 発話行為 hatsuwakoui (speech act)
39.)  sözdizimi  = 統語論 tougoron (syntax)
40.)  sözlükbilim = 語彙論 goiron (lexicology)
41.)  postyapısalcılık = ポスト構造主義 posutokouzoushugi (poststructuralism)
42.)  Sukezumi Hino = 日野資純 hinoshijun (Sukezumi Hino)
43.)  SYSTRAN = SYSTRAN systran (SYSTRAN)
44.)  sözcük dağarcığı = 語彙 goi (vocabulary)
45.)  fonoloji = 音韻 on'in ( phonology)
46.)  embriyoloji = 発生学 hasseigaku ( embryology)
47.)  taksonomi = 分類学 bunruigaku ( taxonomy)
48.)  ayrılmış sözcükler = 予約語 yoyakugo ( reserved words)
49.)  sözdizimi = シンタックス shintakkusu ( syntax)
50.)  uygulamalı dilbilim = 応用言語学 ouyougengogaku ( applied linguistics)
51.)  dilbilim = 英語学 eigogaku ( linguistics)
52.)  dilbilim = 事大主義 jidaishugi ( linguistics)
53.)  matematik = 数理 suuri ( mathematics)
54.)  etnografya = 民族誌 minzokushi ( ethnography)
55.)  dilbilim tipolojisi = 言語類型論 gengoruikeiron ( linguistic typology)
56.)  pozitivizm = 実証主義 jisshoushugi ( positivism)
57.)  göstergebilim = 記号論 kigouron ( semiotics)
58.)  dilbilim = 国語学 kokugogaku ( linguistics)
59.)  dilbilim = 日本語学 nihongogaku ( linguistics)
60.)  küreselleşme ve bölge Çin tarihi  = 国際化と地域化 kokusaikatochiikika ( globalization ar
61.)  morfolojik analiz = 形態素解析 keitaisokaiseki ( morphological analysis)
62.)  morfoloji = 形態論 keitairon ( morphology)
63.)  hermenötik = 解釈学 kaishakugaku ( hermeneutics)
64.)  Çin tarihi = 中国の歴史 chuugokunorekishi ( Chinese history)
65.)  notasyon = 記法 kihou ( notation)
66.)  düzenli ifade = 正規表現 seikihyougen ( regular expression)
67.)  cümle kalıbı = 文型 bunkei ( sentence pattern)
68.)  kelime sırası = 語順 gojun ( word order)
69.)  sözdizimi = 語法 gohou ( syntax)
70.)  sözdizimi = 統語 tougo ( syntax)
71.)  kelime = 単語 tango ( word)
72.)  ulusal dilbilgisi = 国文法 kokubunpou ( national grammar)
73.)  edebi cümle = 文語文 bungobun ( literary sentence)
74.)  kelime dağarcığı = 語い goi ( vocabulary)
75.)  bileşik cümle = 複文 fukubun ( compound sentence)
76.)  SVOC = SVOC svoc ( SVOC)
77.)  ayrıştırıcı = パーザ paaza ( parser)
78.)  biçimbirim = 形態素 keitaiso ( morpheme)
79.)  EBNF = EBNF ebnf ( EBNF)
80.)  sözdizimi = 統辞 touji ( syntax)
```

EK4: Model_v.1 Destek Modülü Kelime Çıktıları – “anestezi1.txt”



anestezi1.txt - Notepad

File Edit Format View Help

```
1.) anestezi = 麻酔  masui (anesthesia)
2.)  ileri anestezi = 先端麻酔 sentanmasui ( advanced anesthesia)
3.)  kuyruk anestezisi = 尾部麻酔 bibumasui ( tail anesthesia)
4.)  iletim anestezisi = 伝導麻酔 dendoumasui ( conduction anesthesia)
5.)  kriyojenik anestezi = 凍結麻酔 touketsumasui ( cryogenic anesthesia)
6.)  epidural anestezi = 硬膜外麻酔 koumakusotomasui ( epidural anesthesia)
7.)  genel anestezi = 全身麻酔 zenshinmasui ( general anesthesia)
8.)  inhalasyon anestezisi = 吸入麻酔 kyuunyuumasui ( inhalation anesthesia)
9.)  lokal anestezi = 局所麻酔 kyokushomasui ( local anesthesia)
10.)  paraservikal blok = 頸部傍ブロック keibubouburokku ( paracervical block)
11.)  genital blok = 陰部ブロック inbuburokku ( genital block)
12.)  saddle blok anestezisi = サドルブロック麻酔 sadoruburokkumasui ( saddle block anesthesia)
13.)  spinal anestezi = 脊椎麻酔 sekitsuimasui ( spinal anesthesia)
14.)  alacakaranlık uykusu = トワイライトスリープ towairaitosuriipu ( twilight sleep)
15.)  ağrı = 疼痛 toutsuu ( pain)
16.)  tedavi = 治療 chiryou ( treatment)
17.)  Perioperatif dönem = 周術期 shuujutsuki ( Perioperative period)
18.)  sistemik yönetim = 全身管理 zenshinkanri ( systemic management)
19.)  sinir bloğu = 神経ブロック shinkeiburokku ( nerve block)
20.)  bronkoskopi = 気管支鏡 kikanshikagami ( bronchoscopy)
21.)  koroner arter bypass cerrahisi = 冠動脈バイパス術 kandoumyakubaipasujutsu ( coronary arte
22.)  postoperatif = 術後 jutsugo ( postoperative)
23.)  endotrakeal entübasyon = 気管内挿管 kikannaisoukan ( endotracheal intubation)
24.)  göğüs drenajı = 胸腔ドレナージ kyoukoudorenaaji ( chest drainage)
25.)  laparotomi = 開腹術 kaifukujutsu ( laparotomy)
26.)  median sternotomi = 胸骨正中切開 kyoukotsuseichuusekkai ( median sternotomy)
27.)  kolesistektomi = 胆嚢摘出術 tannoutekishutsujutsu ( cholecystectomy)
28.)  lokal anestezik = 局所麻酔薬 kyokushomasuiyaku ( local anesthetic)
29.)  intravenöz sedasyon = 静脈内鎮静法 joumyakunaichinseihou ( intravenous sedation)
30.)  kimyasal peeling = ケミカルピーリング kemikarupiiringu ( chemical peel)
31.)  kaza = 偶発症 guuhatsushou ( accident)
32.)  fentanil = フェンタニル fentaniru ( fentanyl)
33.)  enjeksiyon = 注射 chuusha ( injection)
34.)  hemostaz = 止血 shiketsu ( hemostasis)
35.)  damlama = 点滴 tenteki ( drip)
36.)  etkilenen bölge = 患部 kanbu ( affected area)
37.)  kateter  = カテーテル kateeteru ( catheter)
38.)  laparoskop = 腹腔鏡 fukkoukagami ( laparoscope)
39.)  kemoterapi = 化学療法 kagakuryouhou ( chemotherapy)
40.)  endoskop = 内視鏡 naishikyou ( endoscope)
41.)  radyoterapi = 放射線療法 houshasenryouhou ( radiotherapy)
42.)  cerrahi = 手術 shujutsu ( surgery)
43.)  harici kullanım = 外用 gaiyou ( external use)
44.)  analjezi = 鎮痛 chintsuu ( analgesia)
45.)  antikoagülan = 抗凝固薬 kougyoukokusuri ( anticoagulant)
46.)  lavman = 浣腸 kanchou ( enema)
47.)  müshil = 瀉下 shage ( purgative)
48.)  anestezi = 痲酔 masui ( anesthesia)
49.)  TIVA = TIVA tiva ( TIVA)
50.)  ağrı klinisyeni = ペインクリニシャン peinkurinishan ( pain clinician)
51.)  epidural  = エピデュラル epideyuraru ( epidural )
52.)  Kliobiyoz = クリオビオシス kuriobioshisu ( Cliobiosis)
53.)  İlaç = 投薬 touyaku ( Medication)
```

EK5: Model_v.1 Destek Modülü Kelime Çıktıları – "güneşenerjisi1.txt"

güneşenerjisi1.txt - Notepad

File Edit Format View Help

1.) güneş = 太陽 taiyou (sun)
2.) enerji = エネルギー enerugii (energy)
3.) güneş enerjisi = 太陽エネルギー taiyouenerugii (solar energy)
4.) enerji sektörü = エネルギー部門 enerugiibumon (energy sector)
5.) aktinik radyasyon = 化学線 kagakusen (actinic radiation)
6.) aktivasyon enerjisi = 活性化エネルギー kasseikaenerugii (activation energy)
7.) alfa radyasyonu = アルファ線 arufuasen (alpha radiation)
8.) alternatif akım = 交流電流 kouryuudenryuu (alternating current)
9.) alternatif enerji = 代替エネルギー daitaienerugii (alternative energy)
10.) atletizm = アスレチシズム asurechishizumu (athleticism)
11.) atom enerjisi = 原子エネルギー genshienerugii (atomic energy)
12.) atom gücü = 原子電力 genshidenryoku (atomic power)
13.) ışın = ビーム biimu (beam)
14.) beta radyasyonu = ベータ線 beetasen (beta radiation)
15.) bağlanma enerjisi = 結合エネルギー ketsugouenerugii (binding energy)
16.) kara cisim radyasyonu = 黒体放射 kokutaihousha (black body radiation)
17.) kimyasal enerji = 化学エネルギー kagakuenerugii (chemical energy)
18.) evren arka planı radyasyon = 宇宙背景放射 uchuuhaikeihousha (universe background radiat
19.) kozmik ışın = 宇宙線 uchuusen (cosmic ray)
20.) doğru akım = 直流 chokuryuu (direct current)
21.) elastik enerji = 弾性エネルギー danseienerugii (elastic energy)
22.) elektrik = 電気 denki (electricity)
23.) elektromanyetik radyasyon = 電磁放射 denjihousha (electromagnetic radiation)
24.) enerji seviyesi = エネルギー準位 enerugiijun'i (energy level)
25.) gama ışını = ガンマ線 ganmasen (gamma ray)
26.) jeotermal enerji = 地熱エネルギー chinetsuenerugii (geothermal energy)
27.) ısı = 熱 netsu (heat)
28.) yoğunlaşma ısısı = 凝縮熱 gyoushukunetsu (heat of condensation)
29.) ayrışma ısısı = 解離熱 kairinetsu (heat of dissociation)
30.) oluşum ısısı = 生成熱 seiseinetsu (heat of formation)
31.) füzyon ısısı = 融解熱 yuukainetsu (heat of fusion)
32.) çözünme ısısı = 溶液熱 youekinetsu (heat of solution)
33.) süblimleşme ısısı = 昇華熱 shoukanetsu (heat of sublimation)
34.) buharlaşma ısısı = 気化熱 kikanetsu (heat of vaporization)
35.) güneş radyasyonu = 日射 nissha (solar radiation)
36.) yakalama = 遮断 shadan (interception)
37.) iyonlaştırıcı radyasyon = 電離放射線 denrihoushasen (ionizing radiation)
38.) meyve suyu = ジュース juusu (juice)
39.) kinetik enerji = 運動エネルギー undouenerugii (kinetic energy)
40.) gizli ısı = 潜熱 sennetsu (latent heat)
41.) ışınlar = 線 sen (rays)
42.) ışık enerjisi = 光エネルギー hikarienerugii (light energy)
43.) mekanik enerji = 機械エネルギー kikaienerugii (mechanical energy)
44.) mikrodalga = マイクロ波 maikuronami (microwave)
45.) nötron radyasyonu = 中性子放射 chuuseishihousha (neutron radiation)
46.) potansiyel enerji = 潜在エネルギー senzaienerugii (potential energy)
47.) chi = 気 ki (chi)
48.) radyant enerji = 放射エネルギー houshaenerugii (radiant energy)
49.) radyasyon = 放射 housha (radiation)
50.) radyo dalgaları = 電波 denpa (radio waves)
51.) dinlenme enerjisi = 静止エネルギー seishienerugii (rest energy)
52.) ikinci rüzgar = 第二風 dainikaze (second wind)
53.) sinyal = 信号 shingou (signal)
54.) güneş patlaması = 太陽フレア taiyoufurea (solar flare)
55.) güneş belirginliği = 太陽プロミネンス taiyoupurominensu (solar prominence)
56.) güneş radyasyonu = 太陽放射 taiyouhousha (solar radiation)
57.) güneş rüzgarı = 太陽風 taiyoufuu (solar wind)
58.) parlaklık = 輝き kagayaki (brilliance)
59.) özgül ısı = 比熱 hinetsu (specific heat)
60.) canlılık = 活力 katsuryoku (vitality)
61.) rüzgar üretimi = 風の発生 kazenohassei (wind generation)
62.) iş = 仕事 shigoto (work)
63.) röntgen = X線 xsen (x-ray)
64.) moleküler simetri = 分子の対称性 bunshinotaishousei (molecular symmetry)
65.) lazer = レーザー reezaa (laser)
66.) alfa ışını = α線 alphasen (alpha ray)
67.) özgül dönüş = 比旋光度 hisenkoudo (specific rotation)
68.) işitme siniri = 聴神経 choushinkei (auditory nerve)
69.) hissedilir ısı = 顕熱 kennetsu (sensible heat)
70.) sekiz ışın = 8線 8sen (eight rays)
71.) her zaman = いつも itsumo (always)

Ln 1, Col 1 100% Windows (CRLF) UTF-8

güneşenerjisi1.txt - Notepad

File Edit Format View Help

72.) akı = 流束 ryuusoku (flux)
73.) Elektromanyetik dalgalar = 電磁波 denjiha (Electromagnetic waves)
74.) Patlayıcı olaylar = 爆発現象 bakuhatsugenshou (Explosive phenomena)
75.) Aşırı ultraviyole ışınları = 極紫外線 kyokushigaisen (Extreme ultraviolet rays)
76.) Io volkanizması = イオの火山活動 ionokazankatsudou (Io volcanism)
77.) Parlama = 輝く kagayaku (Glowing)
78.) Nem = 潤い juni (Moisture)
79.) Yapılan iş = した仕事 shitashigoto (Work done)
80.) -0,5℃ = -0.5℃ -0.5℃ (-0.5℃)
81.) Enerji kaynakları = エネルギー資源 enerugiishigen (Energy resources)
82.) Helioseismology = 日震学 nichishingaku (Ilseismology)
83.) DC elektrifikasyonu = 直流電化 chokuryuudenka (DC electrification)
84.) β-saç tokası = βヘアピン betaheapin (Beta hairpin)
85.) Mirko = Mirko mirko (Mirko)
86.) Güneş ışığı = 太陽光 taiyoukou (Sunlight)
87.) Parçalanma = 寸断 sundan (Fragmentation)
88.) Limon = レモン remon (Lemon)
89.) Hidrojen sülfit = 硫化水素 ryuukasuiso (Hydrogen sulfide)
90.) 2 Sıra = 2号線 2gousen (2 Line)
91.) Haki = は気 haki (Haki)
92.) Bamberga = バンベルガ banberuga (Bamberga)
93.) NMR = NMR nmr (NMR)
94.) Manyetosfer = 磁気圏 jikiken (Magnetosphere)
95.) Isı kapasitesi = 熱容量 netsuyouryou (Heat capacity)
96.) Çalışma = 勉強 benkyou (Study)
97.) Kurşun Aşındırma = バレットイロージョン barettoiroojon (Bullet Erosion)
98.) Yutma = のみ込め nomikome (Swallow)
99.) Enerji = エネルギ enerugi (Energy)
100.) Ruhsal enerji = スピリチュアルエネルギー supirichuaruenerugii (Spiritual energy)
101.) β ışını = β線 betasen (β ray)
102.) Kılıç = サーベル saaberu (Saber)
103.) γ ışını = γ線 gammasen (γ ray)
104.) AC/DC = 交直 kouchoku (AC/DC)
105.) DC formu = 直流形 chokuryuukatachi (DC form)
106.) Amma = アンマ· anma· (Amma)
107.) Elektrik ışığı = 電灯 dentou (Electric light)
108.) Yüksek ateş = 高熱 kounetsu (High heat)
109.) Ukase = うかされ ukasare (Haze)
110.) Güneş Işığı = 日照 nisshou (Sunshine)
111.) Doğrudan Kazanç = ダイレクトゲイン dairekutogein (direct gain)
112.) Kapatma = シャットアウト shattoauto (shut out)
113.) Bloke = 遮ら saegira (blocked)
114.) İçki = 飲み物 nomimono (drink)
115.) Meyve Suyu = ジュース ju゛su (juice)
116.) Sınır = 境界 kyoukai (boundary)
117.) Hatta = 線上 senjou (on line)
118.) Gönder = 遣っ tsukatsu (send)
119.) Gönder = 遣う tsukau (send)
120.) Alev = 火炎 kaen (flame)
121.) Radyoaktivite = 放射能 houshanou (radioactivity)
122.) Denpa = デンパ denpa (Denpa)
123.) Prototipler = プロトタイパーズ purototaipaazu (Prototypers)
124.) Shin Yukawa = 新油川 shinaburakawa (Shin Yukawa)
125.) Higashikawa = 東川井 higashikawai (Higashikawai)
126.) Işıltı = 煌めき koumeki (sparkle)
127.) Entalpi = エンタルピ entarupi (enthalpy)
128.) Genki = 元気 genki (energy)
129.) Dinçlik = 気力 kiryoku (energy)
130.) Çalışma = シゴト shigoto (work)
131.) İşletme = 業務 gyoumu (business)

EK6: Kullanılan Japonca Yazınlar

1. masui1.pdf
   https://www.mhlw.go.jp/file/06-Seisakujouhou-10800000-Iseikyoku/0000057041.pdf

2. masui2.pdf
   https://core.ac.uk/download/pdf/229702208.pdf

3. masui3.pdf
   https://www.med.jrc.or.jp/Portals/0/images/hospital/clinic/department/masui/pamph_anesthesia.pdf

4. Nagai_çeviri.pdf
   https://www.jstage.jst.go.jp/article/johokanri/14/5/14_335/_pdf

5. Oda_çeviri.pdf
   https://repository.tku.ac.jp/dspace/bitstream/11150/11398/1/jinbun145-03.pdf

6. Senoue_çeviri.pdf
   https://dept.sophia.ac.jp/g/gs/wp-content/uploads/2020/08/Segami.pdf

7. Saito_dilbilim.pdf
   https://www.researchgate.net/profile/Motoki-Saito/publication/314238716_yanyuyanyuzhishiyanyunengli-leishigainiannozhenglikarajianruyanyuguannoyinari-_Language_linguistic_knowledge_and_linguistic_ability_Similar_but_different_technical_concepts_illuminating_different_assu/links/58bd18df45851591c5e4016c/yanyuyanyuzhishiyanyunengli-leishigainiannozhenglikarajianruyanyuguannoyinari-Language-linguistic-knowledge-and-linguistic-ability-Similar-but-different-technical-concepts-illuminating-different-as.pdf

8. Takubo1_dilbilim.pdf
https://www.academia.edu/38664978/%E8%A8%80%E8%AA%9E%E5%AD%A6%E3%81%A8%E3%81%AF

9. Takubo2_dilbilim.pdf
https://www.academia.edu/11362532/%E7%94%9F%E6%88%90_%E5%A4%89%E5%BD%A2_%E6%96%87%E6%B3%95_%E6%98%8E%E6%B2%BB%E6%9B%B8%E9%99%A2_%E6%97%A5%E6%9C%AC%E8%AA%9E%E5%AD%A6%E7%A0%94%E7%A9%B6%E4%BA%8B%E5%85%B8_%E9%A0%85%E7%9B%AE%E5%9F%B7%E7%AD%86_2007_1

10. Udagawa_günenrj.pdf
https://annex.jsap.or.jp/photonics/kogaku/public/41-06-kantougen.pdf

11. Nedo_günenrj.pdf
https://www.enecho.meti.go.jp/category/saving_and_new/attaka_eco/reference/pdf/sekkei/sekkei_1.pdf

12. Odagaki_günenrj.pdf
http://mercury.yukawa.kyoto-u.ac.jp/~bussei.kenkyu/pdf/01/1/0007-011801.pdf