CS 445 Natural Language Processing

Project 2: Text Exploration V.3

Cavit Çakır

23657

Table of Contents:

Environment Information and How to Run

Processor: 2,6 GHz 6-Core Intel Core i7
Memory: 16 GB 2667 MHz DDR4
OS: macOS Big Sur 11.0.1
To run: python3 main.py data_path
Python version: Python 3.8.3

Brief Introduction

In this project, I worked on a news dataset in order to implement language models, word embeddings and see if Zips's & Heap's Laws are hold or not. Also implemented word cloud to explore the dataset.

Dataset

We are provided 4 different sizes of the same dataset. In order to speed up the development I used the smallest dataset.

Examples from dataset:
['Kanal D'de yayımlanan 'Vatanım Sensin' dizisi, bu hafta Cumhurbaşkanı Erdoğan'ın konuşması sebebiyle geç başladı.\nKanal D'nin resmi Twitter hesabından yapılan paylaşımda, dizinin saat 21.00'da başlayacağı paylaşılmıştı. Ancak söylenen saatte dizinin başlamaması ve o saatlerde Kanal D ekranlarında Tayyip Erdoğan'ın konuşmasının yayımlanması çok sayıda vatandaş tarafından tepkiyle karşılandı.\nİşte Kanal D tarafından yapılan paylaşım ve üzerine gelen tepkilerden bazıları:']

["İstanbul Emniyet Müdürlüğü'ne bağlı Uyuşturucu ile Mücadele Şube Müdürlüğü ekipleri, Sarıyer'de önceden belirlenen evlere baskın yaptı.\nİstanbul Emniyet Müdürlüğü'nün Vatan Caddesi'ndeki yerleşkesinden konvoy halinde çıkan çok sayıda polis ekibi saat 06.00 sıralarında Sarıyer Çayırbaşı'na girdi.\nİstanbul Emniyet Müdürlüğü'ne bağlı Uyuşturucu ile Mücadele Şube Müdürlüğü ekipleri, bazı evlere operasyon düzenledi.\nUyuşturucu satıcılarına yönelik düzenlenen operasyona Özel Harekat Timleri de destek verdi. Özel Harekat polisleri bina dış kapısını kırıp içeri girdi. Ekiplerin baskın yaptığı adreste arama yapıldı."]

Zipf’s Law

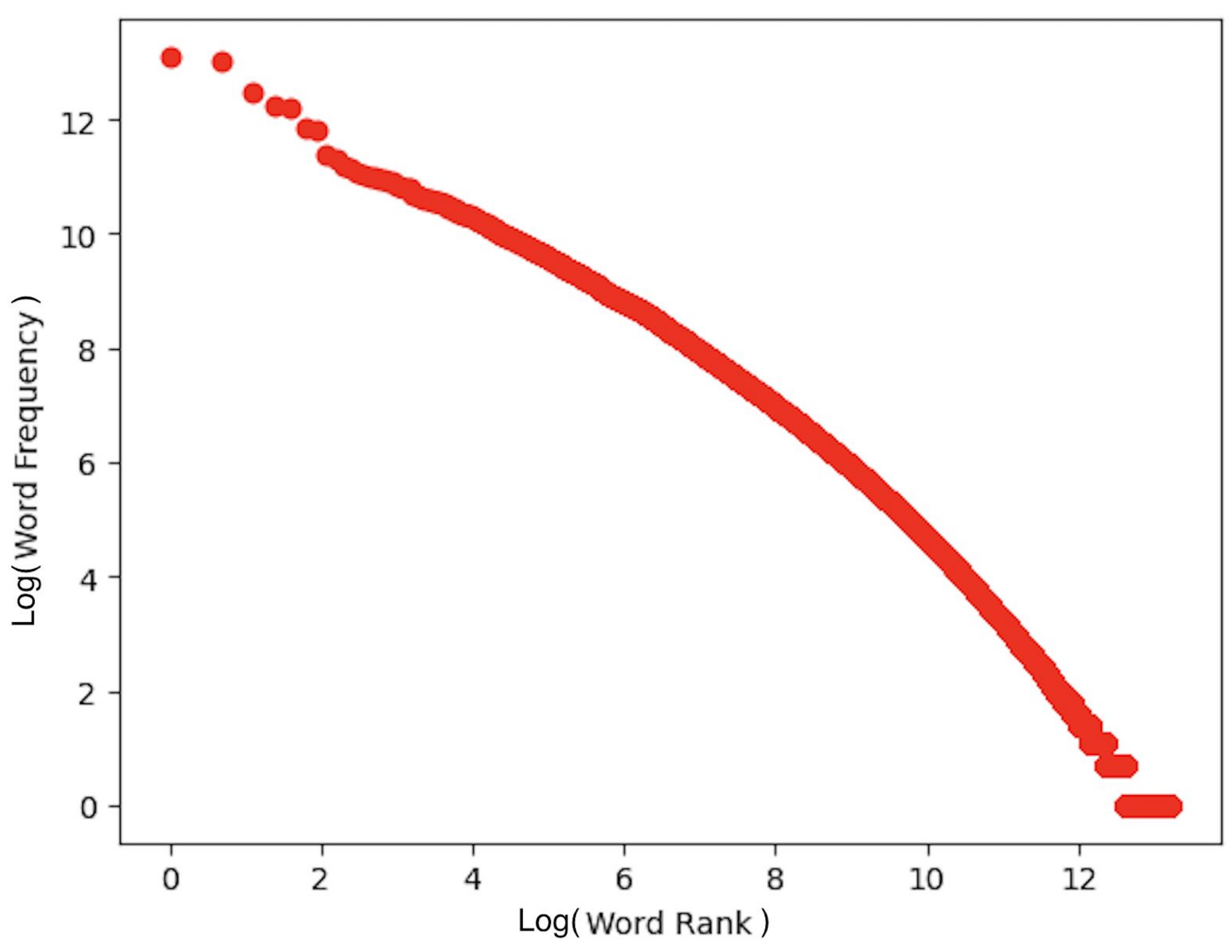


Figure 1

I used the T_sample100000.pkl dataset in Figure 1 in order to prove if Zipf’s Law holds or not.
George Kingsley Zipf suggested that “term frequency decreases rapidly as a function of rank!”.
Representation of this suggestion in mathematical way:
frequency of word = (specific constant) / rank of word
This equation is same as log(f) = log(k) − log(r)
So if we draw log(f) vs log(r) we should see a straight line with slope -1.
As we can see from Figure 1, there is almost a straight line with slope -1, and we can say that Zipf’s Law holds.
I added the T_sample5000.pkl plot to Appendix A. Even in a small dataset we can see that Zipf’s Law holds.

Heap’s Law

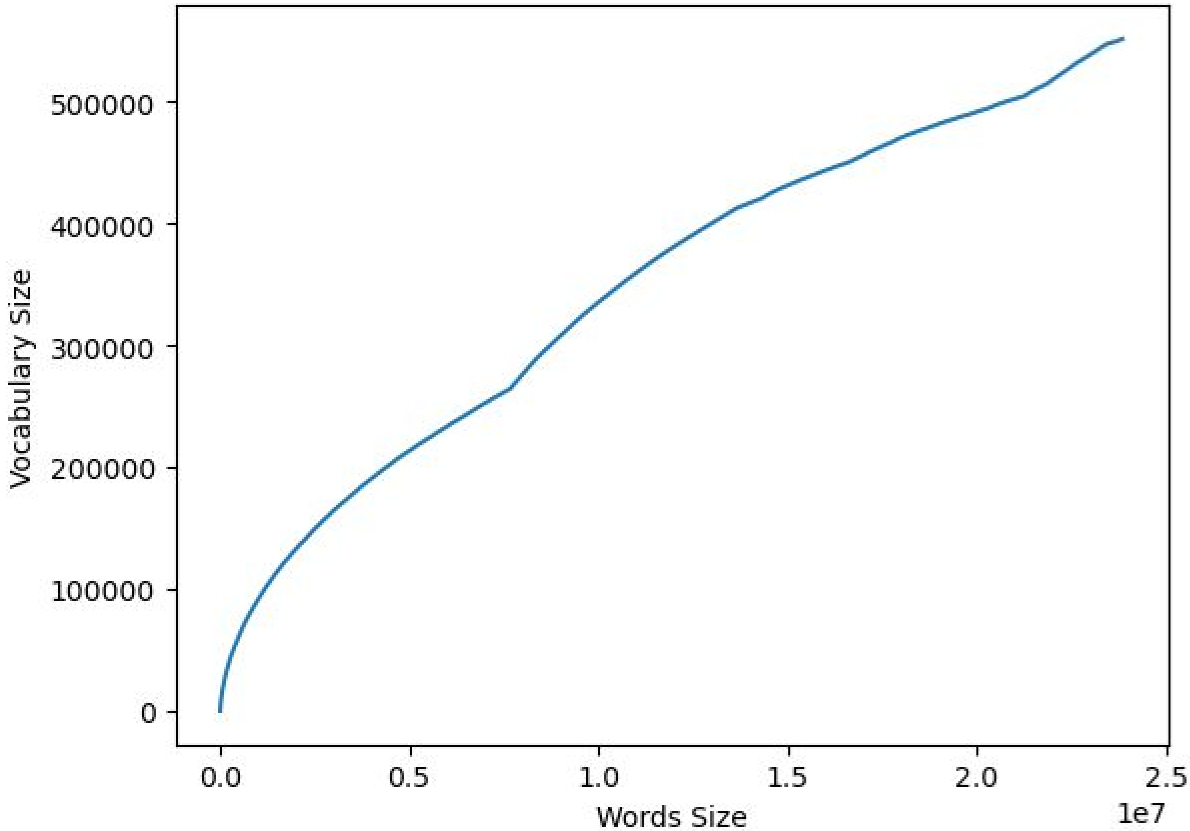


Figure 2

I used the T_sample100000.pkl dataset in Figure 2 in order to prove if Heap’s Law holds or not. Heap’s Law suggests that “The number of new words decreases as the size of the corpus increases”. In order to check if this law is held or not I plotted a graph which is vocabulary size versus words seen so far.

We can observe that, as the number of words seen so far increases, the size of vocabulary increases fast at first, but then increases at a slower rate but it never stops increasing because there is no end of vocabulary and people invents mails etc.

But there are some unexpected parts in the graph which do not fit the expected Heap’s curve. As the reason for that, we can say that maybe there are new corpus contexts or different topics in those documents.

To sum up, if we look at the overall curve, it holds the Heap’s Law.

I added the T_sample5000.pkl plot to Appendix B. Even in a small dataset we can see that Heap’s Law holds.

Wordcloud

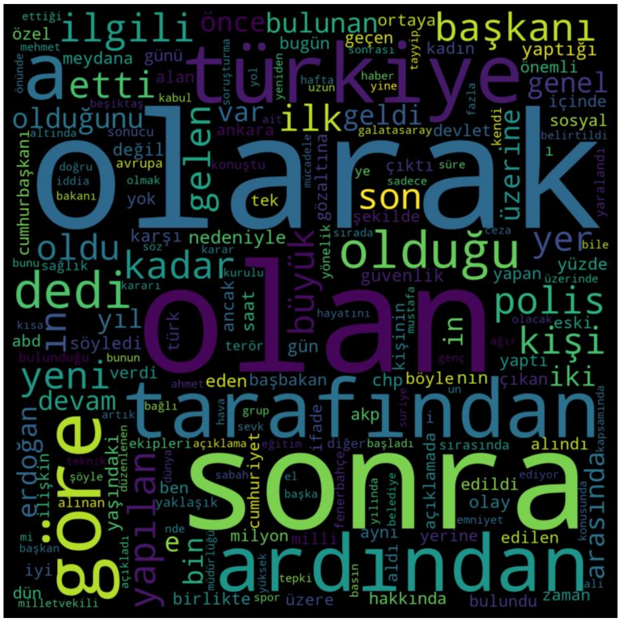


Figure 3

Figure 4

Figure 5

Figure 6

T_sample100000.pkl used as a dataset in Figure 3, 4, 5 and 6.
Figure 3 is TF without the stopwords and Figure 4 is TF with the stopwords. TF is basically a term frequency in the dataset, as we can see in Figure 4 is dominated by stopwords like “bir”, “ve”, “bu”, “da” and etc. If we remove those stopwords, we can see that in Figure 3, common stopwords are not in wordcloud but now those words are not meaningful and do not give much information about the dataset.

Figure 5 is TF-IDF without the stopwords and Figure 6 is TF-IDF with the stopwords. TF-IDF highlights terms that are frequent in the document, but not frequent in general which means that TF-IDF shows the characteristics of the dataset. As we can see “polis”, “turkiye” and etc. are the most important words in our dataset.
Also I added T_sample5000.pkl dataset versions of those wordclouds in Appendix C, results are almost the same.

Language Models

Following models are trained on T_sample20000.pkl dataset.

As we can see from trials, perplexity scores have an inverse ratio with the number of ngram. This result is expected because while generating sentences if we remember history well then we will predict the upcoming word better but if we do not know the history then our prediction will be bad.
In theory KneserNeyInterpolated should work better but in my trials vanilla MLE and KneserNeyInterpolated worked almost the same.
I also did the same trials with the T_sample5000.pkl dataset and the results are similar. You can check the results in Appendix D.

MLE

1gram: not feasible running time for my computer specs.

2gram with input “milli”;
Sentence: Milli stoperi takımın belirleneceği play-off finaline yükselmiş olabilir.
Perplexity Score: 26.837047119895157
Running Time: 166 seconds

3gram with input “milli”;
Sentence: milli eğitim bakanı'na yakışacak bir üslupla betimledi.
Perplexity Score: 2.2039445754429603
Time: 230 seconds

4gram with input “milli”;
Sentence: milli savunma bakanı olması gerektiğine aldırış etmeden, olayın bir uzmanı gibi konuşabilmiştir!

Perplexity Score: 1.2589254117941673
Running Time: 356 seconds

5gram with input “milli”;
Sentence: milli savunma bakanlığı olarak ne askerliğin süresinin düşürülmesi ne de yeni bir bedelli çalışmamız yok"dedi.
Perplexity Score: 1.0
Running Time: 697 seconds

KneserNeyInterpolated

1gram: not feasible running time for my computer specs.

2gram: not feasible for my computer specs.

3gram with input “milli”;
Sentence: milli piyango İdaresince düzenlenen sayısal loto'nun 909. hafta çekilişinde kazandıran numaralar 22, zoran erceg ve jamont gordon 19, anthony ve j.r. smith 17'şer sayılık katkı yaptı mı?
Perplexity Score: 2.1493390943757995
Running Time: 830 seconds

4gram with input “milli”;
Sentence: milli atlet murat kaçar, danimarka'nın başkenti kopenhag'da karşı karşıya gelmişti.
Perplexity Score: 1.2150994473214862
Running Time: 606 seconds

5gram with input “milli”;
Sentence: milli eğitim bakanlığı da “ katılan ” sıfatıyla yer aldı.
Perplexity Score: 1.0046998229702124
Running Time: 496 seconds

Word Embeddings

I created the following embedding models with T_sample50000.pkl dataset.
In my trials, I got the best result from a skip-gram model with 100 dimensions and 3 window sizes(Example 1). It has the highest prediction score and worked fastest.

In a paper named "Dependency-Based Word Embeddings" [1] they claimed that:
Larger windows tend to capture more topic/domain information: what other words (of any type) are used in related discussions?
Smaller windows tend to capture more about the word itself: what other words are functionally similar?
So as I experienced smaller window size is better for our case because we care about the word itself.

In a paper name "Towards Lower Bounds on Number of Dimensions for Word Embeddings" [2] they argued that:
"We discussed the importance of deciding the number of dimensions for word embedding training by looking at the corpus. We motivated the idea using abstract examples and gave an algorithm for finding the lower bound."
So, dimension size is a task based hyperparameter and in my trials I found 100 is better in our task.

In an article [3], the following is claimed:
Skip-gram: works well with small amounts of the training data, represents well even rare words or phrases.
CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words
So, in our country-capital example we are dealing with words which are rare, I expected skip-gram to work better and in my trials, skip-gram worked better.

I also added results of same trials with T_sample5000.pkl dataset. Results are not good due to small dataset. Results could be found in Appendix E.

Parameters that I tried in my trials:
Window size: 3,5,11.
Dimensions: 100, 200, 300
Model: skip-gram and cbow
Relation Tuples:
[('italya','roma'),('almanya','berlin'),('ispanya','madrid'),('hollanda','amsterdam'),('ingiltere','londra'),('suriye','şam'),('ırak','bağdat'),('çin','pekin'),('fransa','paris'),('ukrayna','kiev'),('abd','washington'),('kanada','toronto')]
Input Tuples: ('rusya','') and ('','moskova')
At total I got 32 results as follows:

Example 1:

Dimension: 100
Model: skip-gram,
Window Size: 3
Time in seconds: 216.797182391
Input: ('rusya','')
Results:

('moskova', 0.847544252872467)
('washington', 0.7931650280952454)
('vaşington', 0.7854709625244141)
('tahran', 0.7776059508323669)
('riyad', 0.7774473428726196)

Input: ('','moskova')
Results:

('rusya', 0.8769800662994385)
('ukrayna', 0.8635997772216797)
('almanya', 0.7765350341796875)
('avusturya', 0.7701860666275024)
('fransa', 0.7662272453308105)

Example 2:

Dimension: 100
Model: skip-gram,
Window Size: 5
Time in seconds: 271.176971327
Input: ('rusya','')
Results:

('moskova', 0.8268367052078247)
('tahran', 0.7902897000312805)

('soçi', 0.7868655920028687)
('washington', 0.7806432843208313)
('vaşington', 0.7623207569122314)

Input: ('','moskova')
Results:
('rusya', 0.8485007286071777)
('ukrayna', 0.8342123627662659)
('hollanda', 0.7437592148780823)
('almanya', 0.7407880425453186)
('fransa', 0.7255274057388306)

Example 3:
Dimension: 100
Model: skip-gram,
Window Size: 11
Time in seconds: 333.24031867499997
Input: ('rusya','')
Results:
('moskova', 0.8058627247810364)
('soçi', 0.7723041772842407)
('ukrayna', 0.7612508535385132)
('rus', 0.754408061504364)
("rusya'nın", 0.7538105249404907)

Input: ('','moskova')
Results:
('rusya', 0.8167217969894409)
('ukrayna', 0.7931294441223145)
('dmitri', 0.7038598656654358)
('fransa', 0.6949672698974609)
('rus', 0.6786647439002991)

Example 4:
Dimension: 200
Model: skip-gram,
Window Size: 3
Time in seconds: 299.68789133999996
Input: ('rusya','')
Results:
('moskova', 0.7123792171478271)
('soçi', 0.7052896022796631)

('vaşington', 0.6980272531509399)
('tahran', 0.6905953884124756)
('washington', 0.6871846914291382)

Input: ('','moskova')
Results:
('ukrayna', 0.791801393032074)
('rusya', 0.7623082399368286)
('almanya', 0.7112134099006653)
('hollanda', 0.6907719969749451)
('polonya', 0.684969425201416)

Example 5:
Dimension: 200
Model: skip-gram,
Window Size: 5
Time in seconds: 328.7669415130001
Input: ('rusya','')
Results:
('moskova', 0.6999872922897339)
('riyad', 0.6990851759910583)
('kremlin', 0.694652259349823)
('soçi', 0.684712827205658)
('tahran', 0.6846414804458618)

Input: ('','moskova')
Results:
('ukrayna', 0.7634921669960022)
('rusya', 0.7377407550811768)
('almanya', 0.6443852782249451)
('fransa', 0.6372957229614258)
('slovakya', 0.6337270736694336)

Example 6:
Dimension: 200
Model: skip-gram,
Window Size: 11
Time in seconds: 394.917785716
Input: ('rusya','')
Results:

('moskova', 0.7359180450439453)
("rusya'nın", 0.7260714769363403)
('soçi', 0.7051738500595093)
('putin', 0.6993057131767273)
('rus', 0.6964919567108154)

Input: ('','moskova')
Results:
('rusya', 0.7629435062408447)
('ukrayna', 0.7029561400413513)
('lokomotiv', 0.6192625761032104)
('dmitri', 0.6098300218582153)
('rus', 0.6000860929489136)

Example 7:
Dimension: 300
Model: skip-gram,
Window Size: 3
Time in seconds: 328.15018214500014
Input: ('rusya','')
Results:
('vaşington', 0.6347262859344482)
('moskova', 0.6345219612121582)
('riyad', 0.6246436238288879)
('tahran', 0.6208360195159912)
("rusya'nın", 0.6135982275009155)
Input: ('','moskova')
Results:
('ukrayna', 0.7262554168701172)
('rusya', 0.6944918632507324)
('almanya', 0.6649573445320129)
('finlandiya', 0.6556897163391113)
('slovakya', 0.6547004580497742)

Example 8:
Dimension: 300
Model: skip-gram,
Window Size: 5
Time in seconds: 366.1793840719997
Input: ('rusya','')

Results:

('moskova', 0.6646968126296997)
('soçi', 0.6356329917907715)
("rusya'nın", 0.6302217245101929)
('sergey', 0.6249292492866516)
('minsk', 0.6177489757537842)

Input: ('','moskova')
Results:

('rusya', 0.7049394249916077)
('ukrayna', 0.7032980918884277)
("moskova'nın", 0.6095453500747681)
('cska', 0.6093893051147461)
('kiev', 0.5966991782188416)

Example 9:

Dimension: 300
Model: skip-gram,
Window Size: 11
Time in seconds: 475.75232137700004
Input: ('rusya','')
Results:

('moskova', 0.6727989315986633)
("rusya'nın", 0.6469591856002808)
('kremlin', 0.6461992263793945)
('soçi', 0.6461039781570435)
('dmitri', 0.6389012932777405)

Input: ('','moskova')
Results:

('rusya', 0.7069600820541382)
('ukrayna', 0.6457012295722961)
('dmitri', 0.5533719062805176)
('donetsk', 0.5496358871459961)
('rus', 0.5472872853279114)

Example 10:

Dimension: 100
Model: cbow,
Window Size: 3
Time in seconds: 350.2314897790002

Input: ('rusya','')
Results:
('moskova', 0.738727331161499)
('washington', 0.7288508415222168)
('ukrayna', 0.721463680267334)
('katar', 0.6847485303878784)
('başika', 0.6777483224868774)

Input: ('','moskova')
Results:
('ukrayna', 0.9113376140594482)
('hollanda', 0.8781981468200684)
('rusya', 0.8655756711959839)
('almanya', 0.8550446033477783)
('i̇talya', 0.8521261811256409)

Example 11:
Dimension: 100
Model: cbow,
Window Size: 5
Time in seconds: 365.5781529740002
Input: ('rusya','')
Results:
('ukrayna', 0.7695176601409912)
('moskova', 0.7451097965240479)
('washington', 0.7117413878440857)
('katar', 0.7064324021339417)
('lavrov', 0.7053736448287964)

Input: ('','moskova')
Results:
('ukrayna', 0.8980440497398376)
('rusya', 0.8699040412902832)
('hollanda', 0.8516157269477844)
('almanya', 0.8303980231285095)
('fransa', 0.8256067037582397)

Example 12:

Dimension: 100
Model: cbow,
Window Size: 11
Time in seconds: 396.93826542199986
Input: ('rusya','')
Results:
('ukrayna', 0.7414493560791016)
('moskova', 0.7113135457038879)
('tahran', 0.6503350734710693)
('ermenistan', 0.6413894295692444)
('kırım', 0.6396092772483826)

Input: ('','moskova')
Results:
('ukrayna', 0.9062513113021851)
('rusya', 0.8437826633453369)
('hollanda', 0.825083315372467)
('almanya', 0.8109331130981445)
('polonya', 0.7913578748703003)

Example 13:
Dimension: 200
Model: cbow,
Window Size: 3
Time in seconds: 392.2430402689997
Input: ('rusya','')
Results:
('ukrayna', 0.6871805191040039)
('moskova', 0.6834495663642883)
('rakka', 0.6760140657424927)
('washington', 0.667595624923706)
('katar', 0.6607035398483276)

Input: ('','moskova')
Results:
('ukrayna', 0.8981389999389648)
('hollanda', 0.8730639219284058)
('i̇talya', 0.8442262411117554)
('almanya', 0.833430826663971)
('rusya', 0.8322193622589111)

Example 14:

Dimension: 200

Model: cbow,

Window Size: 5

Time in seconds: 424.6886728320005

Input: ('rusya','')

Results:

('ukrayna', 0.7244968414306641)
('moskova', 0.7128095626831055)
('washington', 0.6677026152610779)
('lavrov', 0.6590718626976013)
('rakka', 0.6465096473693848)

Input: ('','moskova')

Results:

('ukrayna', 0.8992433547973633)
('rusya', 0.8513813018798828)
('hollanda', 0.8397420048713684)
('almanya', 0.8171616792678833)
('japonya', 0.8096118569374084)

Example 15:

Dimension: 200

Model: cbow,

Window Size: 11

Time in seconds: 480.58151548600017

Input: ('rusya','')

Results:

('ukrayna', 0.7346456050872803)
('moskova', 0.6919870972633362)
('lavrov', 0.6432904601097107)
('i̇ran', 0.6395341753959656)
("rusya'nın", 0.6331853866577148)

Input: ('','moskova')

Results:

('ukrayna', 0.8950284123420715)

('rusya', 0.8345697522163391)
('hollanda', 0.7803235054016113)
('japonya', 0.7634563446044922)
('polonya', 0.7600893974304199)

Example 16:

Dimension: 300
Model: cbow,
Window Size: 3
Time in seconds: 436.87147439099954
Input: ('rusya','')
Results:

('moskova', 0.7104461789131165)
('ukrayna', 0.7017318606376648)
('washington', 0.6935857534408569)
('rakka', 0.6870666146278381)
('tahran', 0.6551743745803833)

Input: ('','moskova')
Results:

('ukrayna', 0.8993306159973145)
('hollanda', 0.8780252933502197)
('rusya', 0.8442966938018799)
('i̇talya', 0.8423678874969482)
('fransa', 0.8358744382858276)

Example 17:

Dimension: 300
Model: cbow,
Window Size: 5
Time in seconds: 485.9772616950004
Input: ('rusya','')
Results:

('ukrayna', 0.7066778540611267)
('moskova', 0.6997084617614746)
('rakka', 0.6565234661102295)
('washington', 0.6499203443527222)
('i̇ran', 0.6434057950973511)

Input: ('','moskova')

Results:

('ukrayna', 0.888090968132019)
('hollanda', 0.8417341709136963)
('rusya', 0.8404217958450317)
('almanya', 0.8180384039878845)
('japonya', 0.8083571195602417)

Example 18:

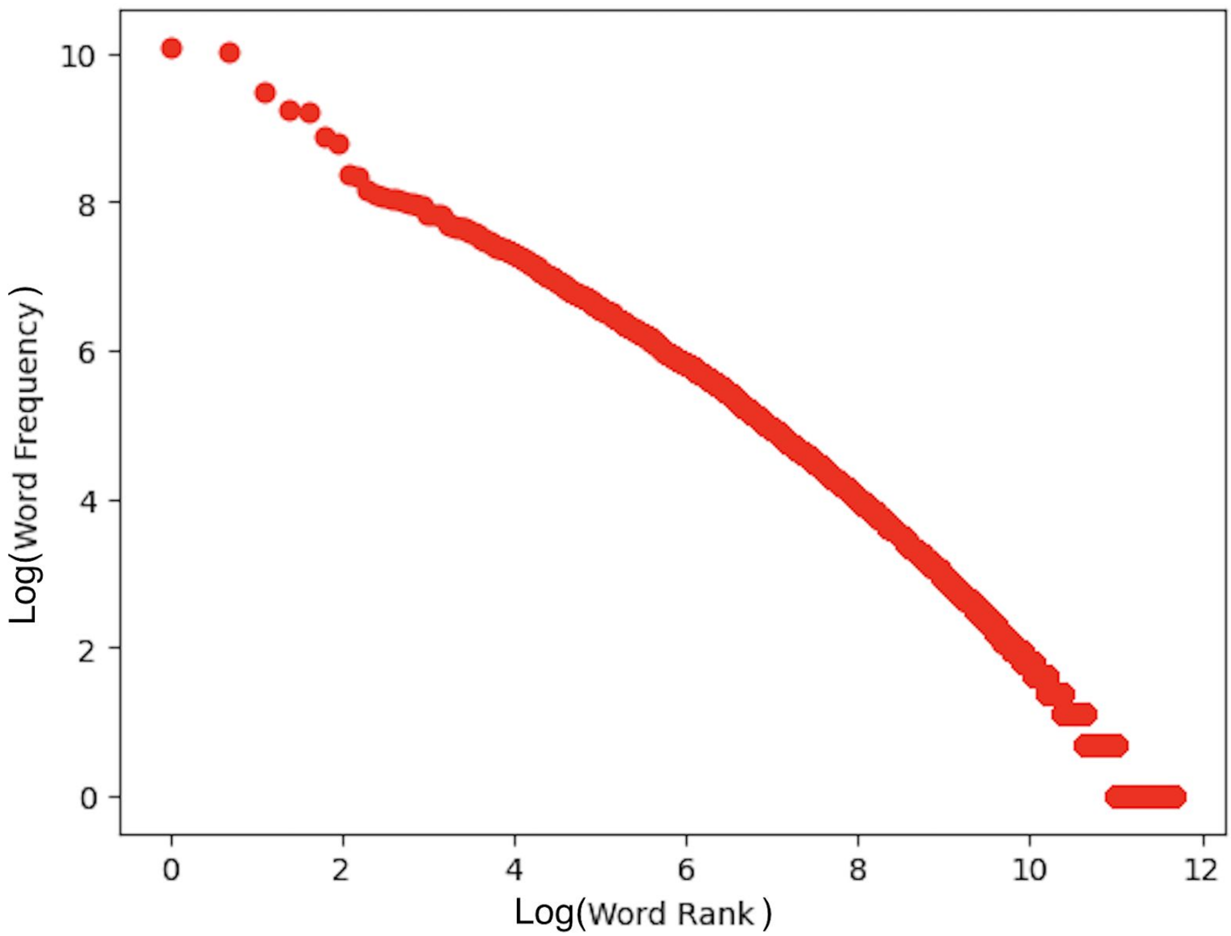
Dimension: 300
Model: cbow,
Window Size: 11
Time in seconds: 549.1320686680001
Input: ('rusya','')
Results:

('ukrayna', 0.7092050909996033)
('moskova', 0.6917195320129395)
('kırım', 0.6395009756088257)
('lavrov', 0.6221363544464111)
('ermenistan', 0.6206549406051636)

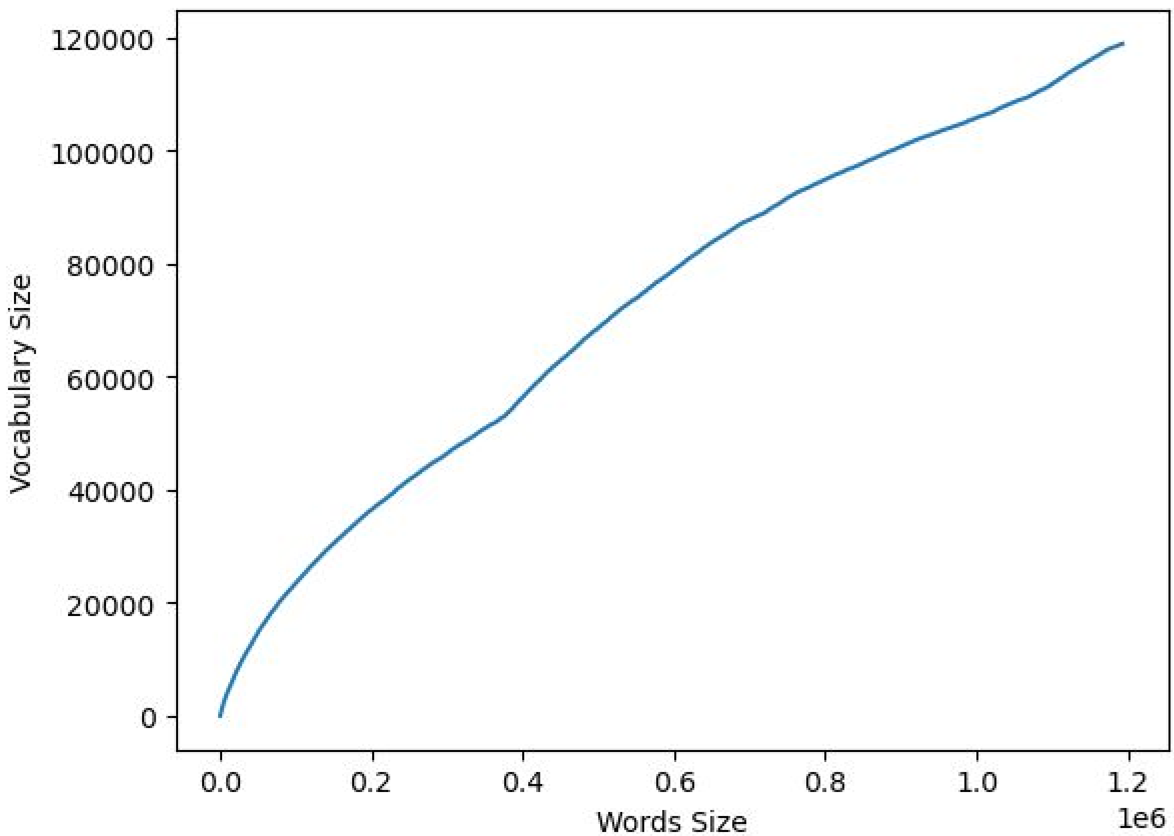
Input: ('','moskova')
Results:

('ukrayna', 0.8945426940917969)
('rusya', 0.8316028118133545)
('hollanda', 0.7860684394836426)
('japonya', 0.7701008319854736)
('polonya', 0.7627453804016113)

Appendix A

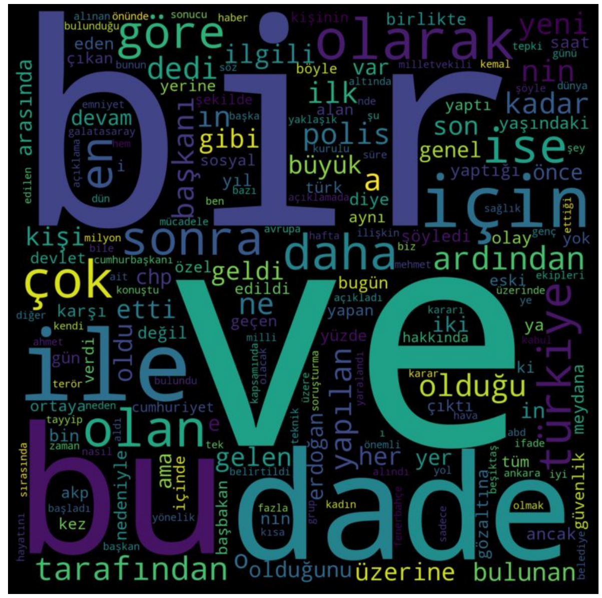


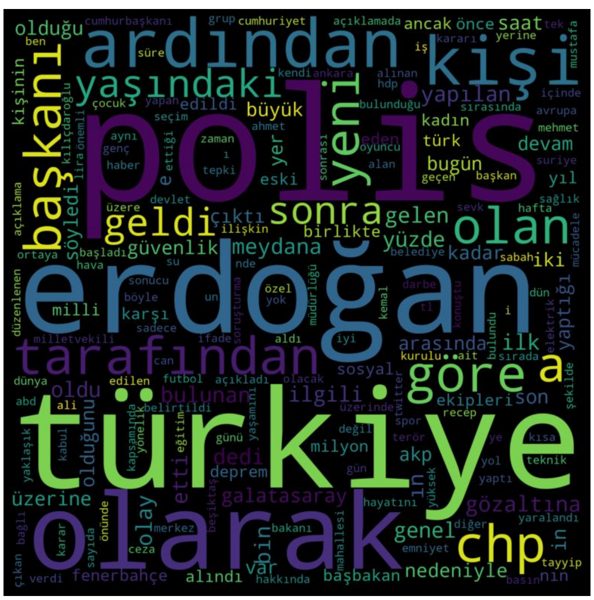
Appendix B

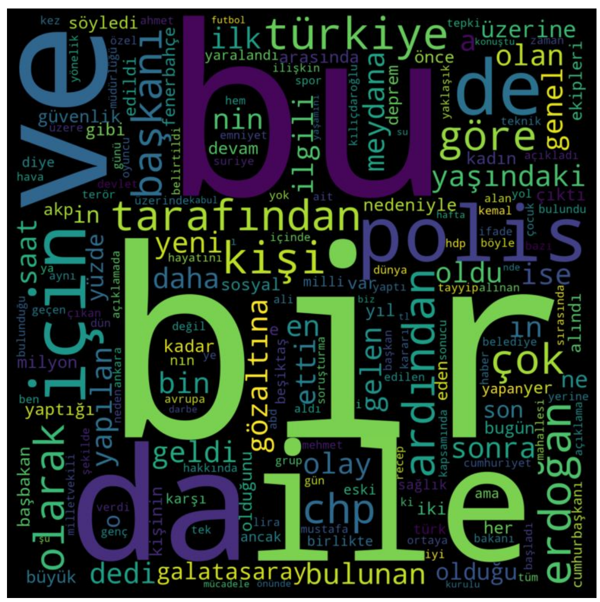

120000
100000
80000
60000
40000
20000
0
Vocabulary Size
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1e6
Words Size

Appendix C


olarak
sonra
göre
olan
türkiye
tarafından
yapılan
başkanı
kadar
yeni
dedi
olduğu
ardından
ilgili
etti
bulunan
geldi
polis
kişi
gelen


bir
ve
bu
da
de
ile
için
olarak
göre
sonra
daha
çok
türkiye
olan
tarafından


polis
erdoğan
türkiye
olarak
ardından
kişi
başkanı
yaşındaki
tarafından
sonra
yeni
geldi
olan
göre
chp


bir
bu
ve
ile
da
için
polis
türkiye
de
tarafından
olarak
kişi
göre
erdoğan
galatasaray
bulunan

Appendix D

MLE-2gram: milli şefimiz sasha goetzel ve adalet arayışı (adanaspor ile sarık ve kaçırılan elektrik direğine çarptı. 37.16241718789977 **Time**: 33.947981704
MLE-3gram: milli piyango idaresinin düzenlediği on numara oyununun 609. hafta çekilişinde , 5 yıl 4'er aya kadar vade seçenekleri mevcuttur. 2.129756262171067 **Time**: 47.11840513100001
MLE-4gram: milli takım'ın kaptanı arda turan da kalya ’ yı unutmadı ve çelenk gönderdi. 1.1161231740339044 **Time**: 65.916011379
MLE-5gram: milli eğitim bakanlığı , 8 yılı dolduran öğretmenlere il içinde rotasyon çalışması ile ilgili internet üzerinden görüşler aldı. 1.0 **Time**: 87.036331899

KneserNey-2gram: not feasible for my computer specs.
KneserNey-3gram: milli piyango idaresi genel müdürlüğünce hassasiyetle takip edilmektedir"ifadeleri kullanıldı beyaz saray web sitesinde bile yeralmıyor. 2.7113993314983977 **Time**: 331.652879827
KneserNey-4gram: milli eğitimin yozlaştırılması ise, çocuklarımız üzerinden geleceğimizi ipotek altına almaktadır. 1.0403880446142055 **Time**: 123.90566256299996
KneserNey-5gram: milli piyango idaresinin düzenlediği on numara oyununun 609. hafta çekilişinde 10 bilen 4 kişi 205 bin 400 lira doksanar kuruş ikramiye kazandı. 1.0062109491311964 **Time**: 124.61906385599997

Appendix E

WE = project02.create_WordVectors(Docs,100,'skip',3) Time: 22.397833523000003

('washington', 0.8616781830787659)
('kürdistanı', 0.8595083951950073)
('bulgaristan', 0.8542243838310242)
('macaristan', 0.852027416229248)
('lübnan', 0.849829912185669)

('afrika', 0.9301445484161377)
('ingiltere', 0.9243122339248657)
('almanya', 0.9236325025558472)
('irlanda', 0.9152437448501587)
('ukrayna', 0.9129889011383057)

WE = project02.create_WordVectors(Docs,100,'skip',5) Time: 22.423015414

('putin', 0.8133633136749268)
('küba', 0.80672687292099)
('kürdistanı', 0.8034670948982239)
('lübnan', 0.8023445010185242)
('beşar', 0.7985395193099976)

('ingiltere', 0.9002585411071777)
('almanya', 0.8962581157684326)
('ukrayna', 0.8919341564178467)
('afrika', 0.8859708309173584)
('romanya', 0.8849648237228394)

WE = project02.create_WordVectors(Docs,100,'skip',11) Time: 24.04649277

('medvedev', 0.781139612197876)
('vladimir', 0.7675997018814087)
('incirlik', 0.7615451812744141)
("putin'in", 0.7560871839523315)
('uçağını', 0.754576563835144)

('polonya', 0.886718213558197)
('belçika', 0.8763113021850586)
('romanya', 0.8694121837615967)
('ukrayna', 0.8673723340034485)

('japonya', 0.8545383810997009)

WE = project02.create_WordVectors(Docs,200,'skip',3) Time: 26.510754906000003
('küba', 0.8573517203330994)
('lübnan', 0.8540811538696289)
('brüksel', 0.8532541990280151)
('angela', 0.8531876802444458)
('kürdistanı', 0.8524071574211121)

('ukrayna', 0.9189229011535645)
('fransa', 0.9174784421920776)
('afrika', 0.9146996736526489)
('almanya', 0.9063720107078552)
('ingiltere', 0.9045618772506714)

WE = project02.create_WordVectors(Docs,200,'skip',5) Time: 26.888804416
('putin', 0.8237704634666443)
('kanada', 0.8202865123748779)
('muhafızları', 0.8191971778869629)
('hizbullah', 0.8184604048728943)
("rusya'nın", 0.8171793222427368)

('almanya', 0.9033888578414917)
('ukrayna', 0.899726152420044)
('japonya', 0.8951320648193359)
('polonya', 0.8945754766464233)
('afrika', 0.8933375477790833)

WE = project02.create_WordVectors(Docs,200,'skip',11) Time: 30.486573115
("putin'in", 0.7909150123596191)
('medvedev', 0.7899088859558105)
('gürcistan', 0.7892126441001892)
('tahran', 0.7873342037200928)
('astana', 0.7804709672927856)

('polonya', 0.8865137696266174)
('ukrayna', 0.8603984117507935)
('macaristan', 0.8603954315185547)

('belçika', 0.8524558544158936)
('romanya', 0.850975513458252)

WE = project02.create_WordVectors(Docs,300,'skip',3) Time: 29.894074308
('brüksel', 0.861997127532959)
('azerbaycan', 0.8603747487068176)
('macaristan', 0.8512738347053528)
('washington', 0.849533200263977)
('kaynaklar', 0.849419891834259)

('ukrayna', 0.9292564392089844)
('afrika', 0.9206398725509644)
('almanya', 0.9127621650695801)
('fransa', 0.9110527038574219)
('i̇rlanda', 0.9072998762130737)

WE = project02.create_WordVectors(Docs,300,'skip',5) Time: 26.541177248999986
('putin', 0.8335407972335815)
('bulgaristan', 0.8326401114463806)
('i̇ncirlik', 0.8295810222625732)
('tahran', 0.8290204405784607)
('kürdistanı', 0.8281960487365723)

('i̇ngiltere', 0.9002046585083008)
('almanya', 0.8913196921348572)
('polonya', 0.8910276293754578)
('amerika', 0.8880304098129272)
('fransa', 0.887525200843811)

WE = project02.create_WordVectors(Docs,300,'skip',11) Time: 30.65090209799999
("putin'in", 0.767113447189331)
('vladimir', 0.7643730640411377)
('putin', 0.757083535194397)
('s-400', 0.7551090717315674)
('medvedev', 0.7546549439430237)

('polonya', 0.8952550888061523)
('ukrayna', 0.8823429346084595)
('belçika', 0.877269983291626)

('İtalya', 0.8747043609619141)
('İngiltere', 0.8512288928031921)

WE = project02.create_WordVectors(Docs,100,'cbow',3) Time: 31.385930804000026
('washington', 0.9478369951248169)
('azerbaycan', 0.9390254020690918)
('urfa', 0.936279833316803)
('ala', 0.9325066804885864)
('obama', 0.9308460354804993)

('batı', 0.9749982357025146)
('almanya', 0.9737517237663269)
('fransa', 0.9659305214881897)
('rusya', 0.9636110067367554)
('kuzey', 0.963318943977356)

WE = project02.create_WordVectors(Docs,100,'cbow',5) Time: 32.573340277
('muş', 0.9283466339111328)
('İtalya', 0.9150819182395935)
('başika', 0.9142926335334778)
('almanya', 0.9132579565048218)
('washington', 0.9113235473632812)

('almanya', 0.9676097631454468)
('rusya', 0.9639055728912354)
('amerika', 0.9605140089988708)
('kore', 0.9557536840438843)
('batı', 0.9551022052764893)

WE = project02.create_WordVectors(Docs,100,'cbow',11) Time: 33.753386863
('İtalya', 0.9144419431686401)
('zirvesi', 0.9053218364715576)
('beyoğlu', 0.9037870764732361)
('temsilcisi', 0.9017254114151001)
('hatay', 0.9003499746322632)

('almanya', 0.9651260375976562)
('rusya', 0.9643968343734741)
('fransa', 0.9601719379425049)

('kore', 0.9548153281211853)
('kuzey', 0.9522536993026733)

WE = project02.create_WordVectors(Docs,200,'cbow',3) Time: 33.60746136399996
('washington', 0.9421255588531494)
('fransa', 0.9420766234397888)
('anadolu', 0.9410465955734253)
('almanya', 0.9387519955635071)
('başbakanı', 0.9361427426338196)

('almanya', 0.9720592498779297)
('ingiltere', 0.9714273810386658)
('tv', 0.9712228178977966)
('amerika', 0.9711527228355408)
('batı', 0.9689205884933472)

WE = project02.create_WordVectors(Docs,200,'cbow',5) Time: 33.989291277999996
('muş', 0.9408857822418213)
('italya', 0.9380991458892822)
('ala', 0.9373815655708313)
('azerbaycan', 0.9342039823532104)
('kıbrıs', 0.9317049980163574)

('almanya', 0.9758681654930115)
('rusya', 0.9728187322616577)
('kore', 0.9643262624740601)
('amerika', 0.9608231782913208)
('ingiltere', 0.9586565494537354)

WE = project02.create_WordVectors(Docs,200,'cbow',11) Time: 37.400677492
('sincan', 0.9227784276008606)
('yaptırımları', 0.9188225865364075)
('italya', 0.9141092300415039)
('lübnan', 0.913769006729126)
('çağlayan', 0.900581955909729)

('almanya', 0.9684951305389404)
('rusya', 0.9626773595809937)
('fransa', 0.9548932909965515)

('amerika', 0.951496958732605)
('kuzey', 0.9486826658248901)

WE = project02.create_WordVectors(Docs,300,'cbow',3) Time: 33.795446760000004
('başbakanı', 0.9343094229698181)
('arabistan', 0.9338981509208679)
('afrika', 0.931601881980896)
('hatay', 0.9313607811927795)
('mısır', 0.9297446012496948)

('almanya', 0.9765951037406921)
('batı', 0.9719436168670654)
('tv', 0.9694924354553223)
('rusya', 0.9687789678573608)
('amerika', 0.9683988690376282)

WE = project02.create_WordVectors(Docs,300,'cbow',5) Time: 35.33960305100004
('washington', 0.9363430738449097)
('azerbaycan', 0.934694766998291)
('hollanda', 0.9316296577453613)
('kaynaklar', 0.9301724433898926)
('s-400', 0.9298504590988159)

('amerika', 0.9784935116767883)
('rusya', 0.9700212478637695)
('almanya', 0.9672857522964478)
('i̇ngiltere', 0.961216390132904)
('fransa', 0.9574841260910034)

WE = project02.create_WordVectors(Docs,300,'cbow',11) Time: 40.530123344
('i̇talya', 0.922166109085083)
('beyoğlu', 0.920951247215271)
('temsilcisi', 0.9186640977859497)
('birecik', 0.9154621362686157)
('belgeseli', 0.912002682685852)

('rusya', 0.9616332054138184)
('almanya', 0.9606381058692932)
('fransa', 0.9516079425811768)

('amerika', 0.9515043497085571)
('çin', 0.9402384161949158)

References

[1]
https://levyomer.files.wordpress.com/2014/04/dependency-based-word-embeddings-acl-2014.pdf
[2] https://www.aclweb.org/anthology/I17-2006.pdf

[3] https://www.quora.com/What-are-the-continuous-bag-of-words-and-skip-gram-architectures

[4]
https://towardsdatascience.com/natural-language-processing-feature-engineering-using-tf-idf-e8b9d00e7e76