Bilkent University

CS425: Algorithms for Web Scale Data

Turkish News Analytics

Selim Fırat Yılmaz - 21502736

Berk Mandıracıoğlu - 21501741

April, 2018

Table of Contents

Introduction

Online News articles are one of the channels that media companies connect with their users through Internet. Every day many news articles are produced and read. News articles are online texts that contain up-to-date information in many contexts. We have inspired from the website http://eventregistry.org/.

Datasets

News Dataset

We needed There is no real time data available for our purpose. However, we acquired the data scraping the news websites.

In short, we run our algorithms on ~180.000 news articles crawled by second crawler described below:

Breadth First Crawler

Our initial plan was to continuously crawl the following Turkish news websites:

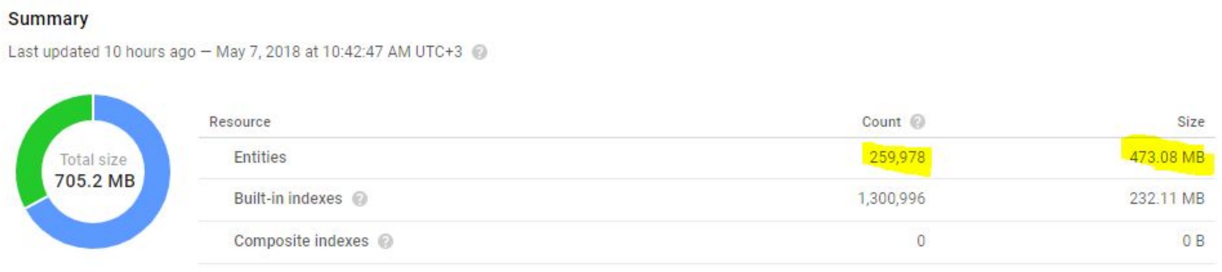
1. Cumhuriyet: www.cumhuriyet.com.tr
2. NTV: www.ntv.com.tr
3. Hurriyet: www.hurriyet.com.tr
4. Milliyet: www.milliyet.com.tr
5. Posta: www.posta.com.tr
6. Sabah: www.sabah.com.tr
7. Star: www.star.com.tr
8. Takvim: www.takvim.com.tr

We used started crawling milliyet.com.tr and takvim.com.tr. Our algorithm consist of two simple steps:

1. Store data if there are any significant (news story, news title, news description) content in the page via news-please library.

2. Visit all URLs in breadth first order targeted to source domain apply step 1 & 2 again.

We implemented the crawler and ran for 4 days and crawled 259.978 news articles.



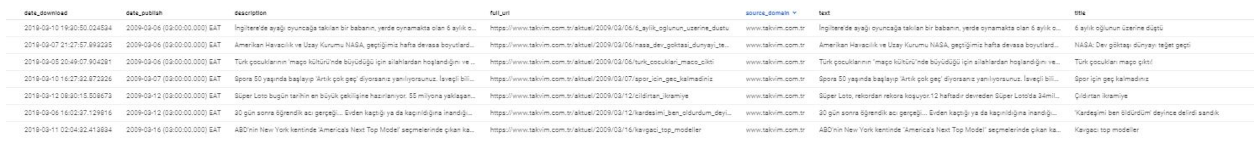
Summary

Last updated 10 hours ago — May 7, 2018 at 10:42:47 AM UTC+3

Total size 705.2 MB

Resource	Count	Size
Entities	259,978	473.08 MB
Built-in indexes	1,300,996	232.11 MB
Composite indexes	0	0 B

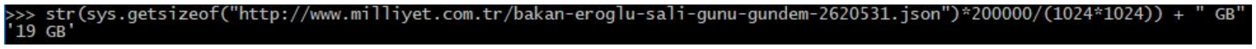
Title, description, publish date, article link are extracted for each news article.



date_download	date_publish	description	full_url	source_domain	text	title
2018-03-10 19:30:50.024534	2009-03-06 (03:00:00.000) EAT	İngiltere'de ayağı oyuncağa takılan bir babanın, yerde oynamakta olan 6 aylık o...	https://www.takvim.com.tr/aktuel/2009/03/06/6_aylik_oglunun_uzerine_dustu	www.takvim.com.tr	İngiltere'de ayağı oyuncağa takılan bir babanın, yerde oynamakta olan 6 aylık o...	6 aylık oğlunun üzerine düştü
2018-03-07 21:27:57.893235	2009-03-06 (03:00:00.000) EAT	Amerikan Havacılık ve Uzay Kurumu NASA, geçtiğimiz hafta devasa boyutlard...	https://www.takvim.com.tr/aktuel/2009/03/06/nasa_dev_goktasi_dunyayi_te...	www.takvim.com.tr	Amerikan Havacılık ve Uzay Kurumu NASA, geçtiğimiz hafta devasa boyutlard...	NASA: Dev göktaşı dünyayı teğet geçti
2018-03-05 20:49:07.904281	2009-03-06 (03:00:00.000) EAT	Türk çocuklarının 'maço kültürü'nde büyüdüğü için silahlardan hoşlandığını ve ...	https://www.takvim.com.tr/aktuel/2009/03/06/turk_cocuklari_maco_cikti	www.takvim.com.tr	Türk çocuklarının "maço kültürü"nde büyüdüğü için silahlardan hoşlandığını ve...	Türk çocukları maço çıktı!
2018-03-10 16:27:32.872326	2009-03-07 (03:00:00.000) EAT	Spora 50 yaşında başlayıp 'Artık çok geç' diyorsanız yanılıyorsunuz. İsveçli bili...	https://www.takvim.com.tr/aktuel/2009/03/07/spor_icin_gec_kalmadiniz	www.takvim.com.tr	Spora 50 yaşında başlayıp 'Artık çok geç' diyorsanız yanılıyorsunuz. İsveçli bili...	Spor için geç kalmadınız
2018-03-12 08:30:15.508673	2009-03-12 (03:00:00.000) EAT	Süper Loto bugün tarihin en büyük çekilişine hazırlanıyor. 55 milyona yaklaşan...	https://www.takvim.com.tr/aktuel/2009/03/12/cildirtan_ikramiye	www.takvim.com.tr	Süper Loto, rekordan rekora koşuyor.12 haftadır devreden Süper Loto'da 34mil...	Çıldırtan ikramiye
2018-03-06 16:02:37.129816	2009-03-12 (03:00:00.000) EAT	30 gün sonra öğrendik acı gerçeği... Evden kaçtığı ya da kaçırıldığına inandığı...	https://www.takvim.com.tr/aktuel/2009/03/12/kardesimi_ben_oldurdum_deyi...	www.takvim.com.tr	30 gün sonra öğrendik acı gerçeği... Evden kaçtığı ya da kaçırıldığına inandığı...	'Kardeşimi ben öldürdüm' deyince delirdi sandık
2018-03-11 02:04:32.413834	2009-03-16 (03:00:00.000) EAT	ABD'nin New York kentinde 'America's Next Top Model' seçmelerinde çıkan ka...	https://www.takvim.com.tr/aktuel/2009/03/16/kavgaci_top_modeller	www.takvim.com.tr	ABD'nin New York kentinde "America's Next Top Model" seçmelerinde çıkan ka...	Kavgacı top modeller

Crawled data are stored in Google Cloud Datastore which is NoSQL Document Database. Datastore ease our job to access our data while using other Google Cloud services to run our algorithms in.

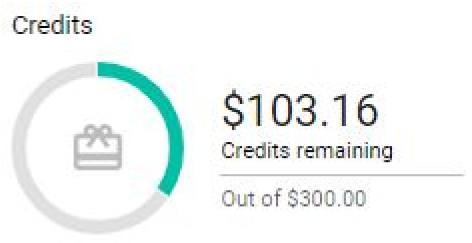
We want to filter already-visited URLs to speed up our data-crawling speed. However, when we try to hash 200.000 URLs(which is not a distant) to a regular hashmap, it takes 19 GB memory. This implementation uses so much memory due to Step 2 in the algorithm above.



```
>>> str(sys.getsizeof("http://www.milliyet.com.tr/bakan-eroglu-sali-gunu-gundem-2620531.json")*200000/(1024*1024)) + " GB"
'19 GB'
```

This image demonstrates that how much space 200.000 url strings take in Python3.

We needed our crawler to work continuously without crashing. However, to run our crawler on the Google Cloud Compute Instance, we needed expensive Compute instances with high memory to run our crawler. When that was the case, we were to run out of free credits provided by Google Cloud in only 4 days. We learned that crawling data is not so cheap.



To solve that problem, we implemented the Bloom Filters described in Bloom Filters section. However, we also implemented the crawler in a new way, as described in the following section:

Latest News Crawler

Since we realized the requirement for memory-efficient implementation for crawler and latest news would be better for us to make some inferences on data, we implemented this crawler.

RSS is a standard for distribution of contents such as news to online users. News websites distributes their latest news(only title, url, and description) through their RSS's.

The procedure for our latest news crawler is the following:

1. Visit RSS of 64 news websites.
2. Extract title, description, and URLs from RSS entities.
3. Visit URL of each entity and extract content via news-please library.
4. Store the all crawled information (title, description, content, url) for each news article into Google Datastore. (if URL is not visited before)
5. Repeat these steps every 15 minutes.

The news websites we are crawling are:

1. Anadolu Ajansı: https://aa.com.tr/tr/rss/default?cat=guncel
2. Cumhuriyet: http://www.cumhuriyet.com.tr/rss/son_dakika.xml
3. Anayurt Gazetesi: http://www.anayurtgazetesi.com/sondakika.xml
4. Dünya Gazetesi: https://www.dunya.com/rss?dunya
5. Balkan Günlüğü: https://www.balkangunlugu.com/feed/

6. Gerçek Gündem: http://www.gercekgundem.com/rss
7. Haberturk: http://www.haberturk.com/rss
8. Hurriyet: http://mix.chimpfeedr.com/07479-Hurriyet-Gazetesi
9. Milliyet Gazetesi: http://www.milliyet.com.tr/rss/rssNew/gundemRss.xml
10. Önce Vatan Gazetesi: http://www.oncevatan.com.tr/rss.php
11. Sabah Gazetesi: http://mix.chimpfeedr.com/d1bed-Sabah-Gazetesi
12. Star: http://www.star.com.tr/rss/rss.asp
13. Takvim: https://www.takvim.com.tr/rss/anasayfa.xml
14. Türkiye Gazetesi: http://www.turkiyegazetesi.com.tr/rss/rss.xml
15. Vatan Gazetesi: http://mix.chimpfeedr.com/68482-Vatan-Gazetesi
16. Yeniçağ: http://www.yenicaggazetesi.com.tr/rss
17. Yeni Mesaj: http://www.yenimesaj.com.tr/rss.php
18. Yeni Şafak: https://www.yenisafak.com/Rss
19. Yurt Gazetesi: http://www.yurtgazetesi.com.tr/rss.php
20. Sozcu: https://www.sozcu.com.tr/feed
21. Fotomaç: https://www.fotomac.com.tr/rss/anasayfa.xml
22. A Haber: https://www.ahaber.com.tr/rss/anasayfa.xml
23. CNNTurk: https://www.cnnturk.com/feed/rss/news
24. EuroNews: http://feeds.feedburner.com/euronews/tr/home?format=xml
25. NTV: http://mix.chimpfeedr.com/6ab71-NTV
26. TRT Haber: http://www.trthaber.com/sondakika.rss
27. Ulusal Kanal Haber Portalı: http://www.ulusal.com.tr/rss.php
28. Ajans Haber: http://www.ajanshaber.com/rss
29. Amerika Bülteni: http://feeds.feedburner.com/amerikabulteni?format=xml

30. Ankara Review: http://www.ankarareview.com/feed/
31. Ay Gazete: http://www.aygazete.com/rss/gundem-haberleri
32. Bas News: http://www.basnews.com/index.php/tr/detail/content/195-basnews-tr?format=feed&type=rss
33. BBC Türkçe: http://feeds.bbci.co.uk/turkce/rss.xml
34. Bianet: http://bianet.org/bianet.rss
35. Canli Haber: https://www.canlihaber.com/rss/
36. Dipnot: http://www.dipnot.tv/feed/
37. Diken: http://www.diken.com.tr/feed/
38. DW-World: https://rss.dw.com/rdf/rss-tur-all
39. En Son Haber: http://www.ensonhaber.com/rss/ensonhaber.xml
40. Eurovizyon: http://www.eurovizyon.co.uk/rss.php
41. F5 Haber: http://www.f5haber.com/rss/haberler.xml
42. Gazete Duvar: https://www.gazeteduvar.com.tr/feed/
43. Gazete Karınca: http://gazetekarinca.com/feed/
44. Girişim Haber: http://www.girisimhaber.com/rss.xml
45. Haber vaktim: https://www.habervaktim.com/rss/
46. Haberler.com: http://rss.haberler.com/rss.asp?kategori=sondakika
47. Haber7: http://sondakika.haber7.com/sondakika.rss
48. Halkin Habercisi: http://www.halkinhabercisi.com/feed
49. İleri Haber: http://ilerihaber.org/rss.xml
50. İnadına Haber: http://inadinahaber.org/feed/
51. İnternet Haber: http://www.internethaber.com/rss
52. Kampüs Haber: http://www.kampushaber.com/rss.xml

53. Manset Haber: http://www.mansethaber.com/rss.xml

54. Senin Medyan: https://seninmedyan.org/feed/

55. National Turk: http://www.nationalturk.com/feed/

56. Objektif Haber: http://www.objektifhaber.com/sondakika.rss

57. Oda TV: https://odatv.com/rss.php

58. Pirha: https://www.pirha.net/feed/

59. Press Turk: http://www.pressturk.com/rss.xml

60. Haber Sol: http://haber.sol.org.tr/rss/tumicerik

61. Sonsöz: http://sonsoz.com.tr/feed/

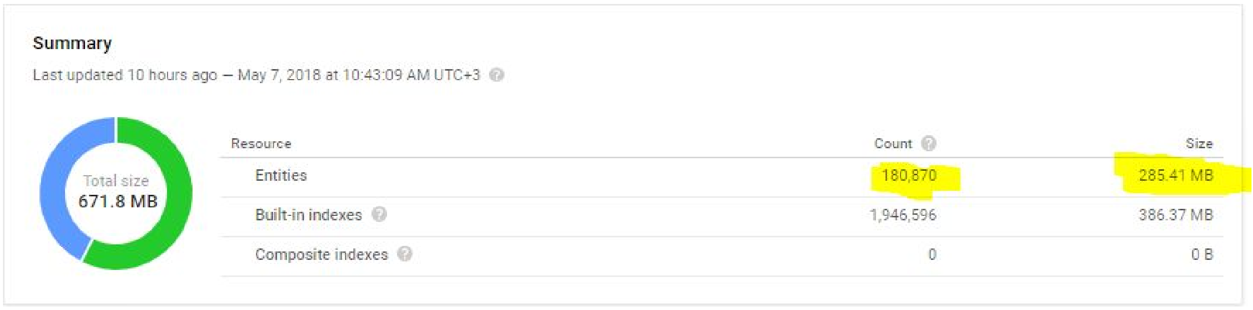
62. Sputnik News Türkçe: https://tr.sputniknews.com/export/rss2/archive/index.xml

63. Türkiye Haber Ajansı: http://www.turkiyehaberajansi.com/rss.xml

64. Yakin Plan: https://yakinplan.com/feed/

Our latest news crawler is running on cheapest Google Cloud Compute f1-micro (1 vCPU, 0.6 GB memory)] and crawling since Mar 13, 2018, 8:30:00 PM and crashed only 2 times since then. We achieved both stability and memory-efficiency through our new implementation.

Our crawler has crawled 180.870 news articles in total from Mar 13, 2018, 8:30:00 PM until now:



Summary

Last updated 10 hours ago – May 7, 2018 at 10:43:09 AM UTC+3

Total size 671.8 MB

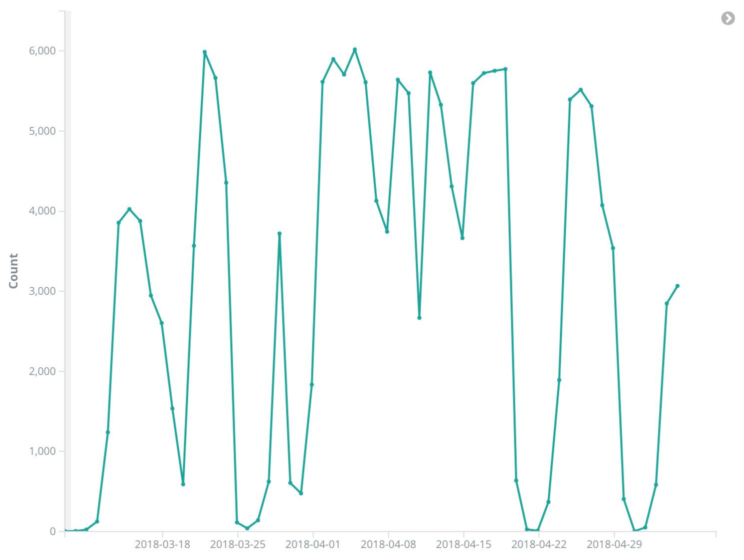
Resource	Count	Size
Entities	180,870	285.41 MB
Built-in indexes	1,946,596	386.37 MB
Composite indexes	0	0 B

Kibana for Data Visualization

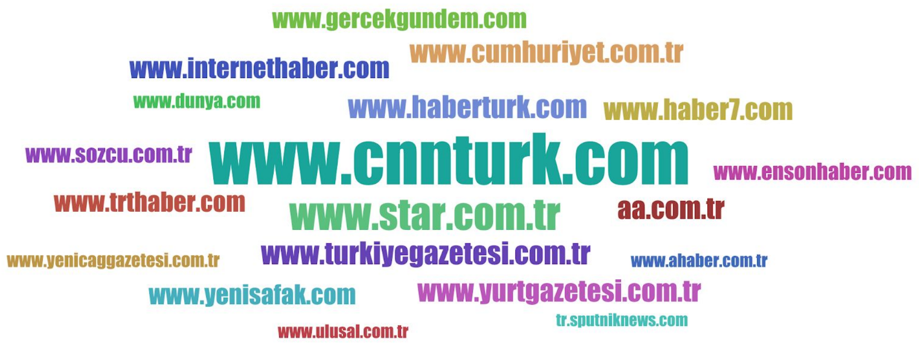
Kibana is a data visualization plugin for elasticsearch. Elasticsearch is a data analytics engine that enables fast queries on data. News dataset was converted to Elasticsearch indices to work with Kibana. Dataset was visualized and analysed in the following subsections.

General Analysis of Dataset

The news dataset was first analysed in terms of the number of news fetched with respect to time. It can be seen from figure below that thousands of news were fetched frequently on average. Moreover, fluctuations on news count can be observed over time, some days less number of news were fetched compared to others.

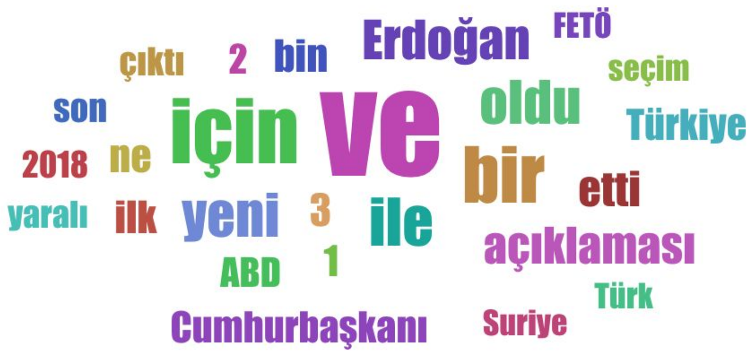


The dataset was examined in terms of counts of the distinct news sites. Domain cloud were plotted in order to view news sites proportional with their published news counts.

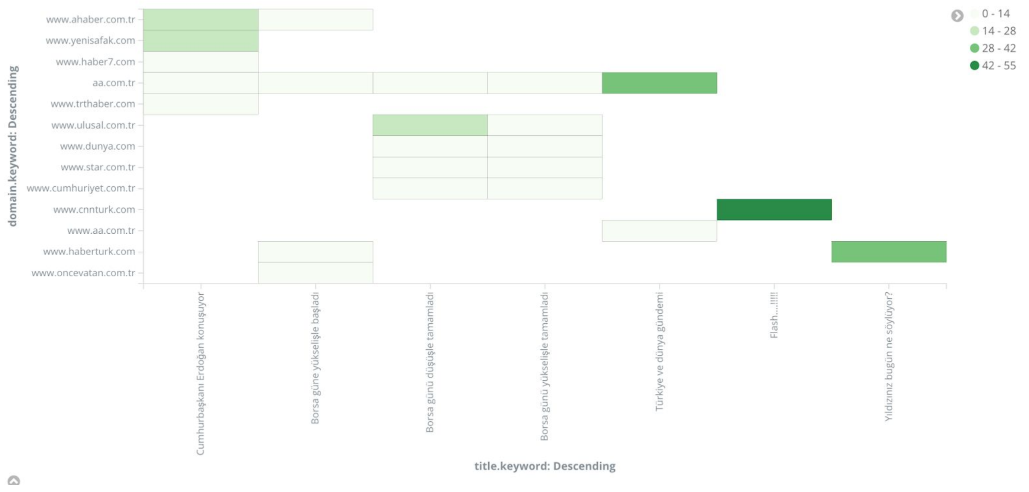


It is clear that our dataset contains more news from *cnnturk.com* and *star.com.tr* compared to other news sites. Moreover, the news title cloud were plotted to analyse the dataset in terms of popular titles.

It is clear that *Cumhurbaşkanı Erdoğan Konuşuyor* is the most popular news title among other news. It can also be observed that news titles related to stock market are popular as well. In addition, dataset was examined in terms of words that are used inside titles as shown in word cloud below:



It is clear that if we disregard the stop words, *Erdoğan, seçim* and *FETÖ* are the most used words among titles. The dataset was examined with respect to news sites and their published news titles. The heatmap below represents the most used titles versus their respective news site domain.



It is clear that ahaber.com.tr and yenisafak.com are biased towards Tayyip Erdoğan as they have the most news with having *Erdoğan* in their title. We can conclude that *Erdoğan* is the most used title and word which dominates other news titles.

Apache Beam for Fast and Parallel computations

Apache Beam was used to implement fast, and parallel working algorithms for our applications. Apache Beam is based on the idea of MapReduce article published in 2004 and it is model for defining data-processing pipelines. It adds features from open source distributed data processing environments such as Hadoop, Spark and Google's

Dataflow. Apache Beam works by defining pipeline to process the input data. Data are represented with PCollections and they can be both output and input as the data are pipelined. At each stage of pipeline, data are transformed by using PTransforms such as mapper, grouper, filter functions.

6000 Tweets Sentiment Classification Dataset

6000 tweets dataset consists of 3000 negative, 1552 positive and 1448 notr tweets. Test data is chosen as 20% of the dataset which is 1200. Validation splits are chosen as 20% of training set which are 960 tweets each.

The dataset is obtained from https://github.com/sercankulcu/twitterdata [19].

35000 Sentences Named Entity Recognition Dataset

Algorithms

Bloom Filtering

Task Definition

As we discussed in "Breadth First Crawler" section, we encountered memory problems while crawling websites. The purpose of bloom filtering is to lookup for visited websites for our crawlers and not to visit websites that are visited before. The bloom filters are to check a membership in a set for an element.

Algorithm Definition

The section below is taken from https://github.com/selimfirat/bloom-filtering [1] which is implemented and written by Selim Fırat Yılmaz(one of our group members).

In data streaming, we may need to apply some filtering process to accept some elements that meet a criterion. Rejected elements are dropped whereas accepted elements are proceeded to the stream.

For the case of Bloom Filtering, the criterion is to lookup for a membership of a set. Bloom filters eliminate most of the elements that are not in the set and rejected elements are certainly not in the set. In other words, the false negative is always 0 whereas bloom filters can only say the element is "maybe" in the set.

Procedure of the bloom filtering is the following:

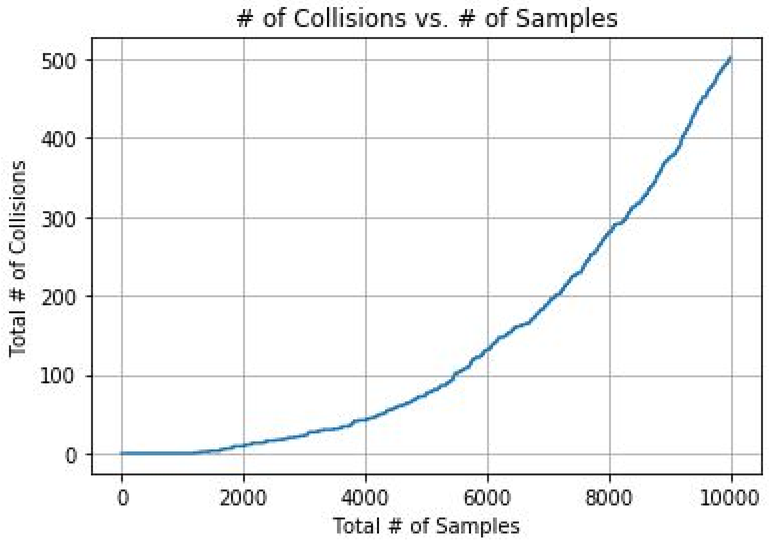
1. Init the array of bits a, initially all 0.
2. Set 2 hash functions h1(x), h2(x).
3. ones(x) gives the the # of bits in the binary representation of x.
4. zeros(x) gives the the # of bits in the binary representation of x
5. For element el to be added to the set,
    - a[h1(zeros(el))] = 1
    - a[h2(ones(el)) = 1
6. To check whether an element might be in the set,
    - a[h1(zeros(el))] == 1 and a[h2(ones(el)) == 1
7. To check whether an element is not in the set,
    - not (a[h1(zeros(el))] == 1 and a[h2(ones(el)) == 1)

More general version of the bloom filtering can be found on the book[2] below in references section.

Experimentation Methodology

We applied the described algorithm for bit array with length 2^16, when distinct random samples are introduced and tested to the bloom filter.

Experimentation Results



Number of collisions here indicates the false positive rate which is 0.05 in this analysis.

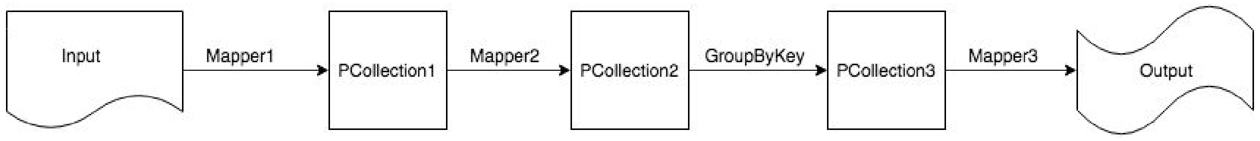
Discussion

Thus, we can say that we have not visited a certain url before with probability of 0.95 if an url is not hashed to same locations before. If an URL is not marked as visited, we can say that we already visited that url with 0.05 probability. These are good rates for us to use this algorithm for our task which is not to re-visit any visited url.

Clustering News

LSH with Apache Beam

Regular serially executing LSH algorithm is not fast enough for news dataset to find similar news articles. Therefore, Apache Beam was used to implement LSH in parallelized manner and find similar news fast. The pipeline of the LSH implementation can be seen below:



This pipeline is explained in the following subsections.

Mapper1:

- **Input:** <key = documentId, value = line representation of news>
- **Output:** <key = documentId, value = 32-bit int shingle(of defined size) >

Firstly, Mapper1 seperates news into shingles of size k, where k is given. Redundant shingles are discarded. Shingles are hashed to 32-bit integers and added to list of shingles for the news. The doucumentId, list of shingles pair is output.

Mapper2:

- **Input:** <key = documentId, value = list of shingles>
- **Output:** <key = band, value = documentId>

Firstly, h number of hash functions are applied to each shingle,, where h is given. For each hash function minimum value is calculated and signature row for the news is found. Then, for each band in signature row (band, documentId) is output

GroupByKey:

The output of Mapper2 is grouped by keys and pipelined to Mapper3.

Mapper3:

- Mapper3 input: <key = band, value = list of documentId >
- Mapper3 output: <key = pair of documentId, value = 1 >

For each pair in value_list which is list of documentIds :

- If pair1 > pair2, then emit([pair2,pair1],1)
- Else, emit([pair1, pair2], 1)

GroupByKey:

The output of Mapper3 is grouped by keys and final output is written.

Experimentation Methodology and Results

Plagiarism Corpus Dataset was used to experiment with LSH implementation [21]. In the procedure of making this dataset, participants were asked to :

- Copy and Paste (cut) ,
- Lightly Revise(light),
- Heavily Revise(heavy) a source,

Moreover, for cut category, they did not specify the start and end positions to copy the source. Therefore,some copy and pasted works are not very similar as their intersection can be small for jaccard similarity metric.

In the experiments the aim was to find articles that have the cut category.

In the first experiment, number of bands was 25 and number of hash functions were 100 so number of rows in every band was 4. The results are as follows:

- True positive = 0.3157
- False positive = 0.13157
- True negative = 0.86842
- False negative = 0.6842

In the second experiment, number of bands was 20 and number of hash functions were 100 so number of rows in every band was 5. The results are as shown:True positive = 0.210

- False positive = 0.2108
- False positive = 0.0395
- True negative = 0.9605
- False negative = 0.7892

False negative is high because start and end places of copying a document is not specified so not all cuts may be highly similar. The result confirm that some copy and pasted documents have around 45% jaccard similarity because of the reason mentioned. Moreover, similarity of lightly revisioned and cut documents can be same because the amount of revision is very slight. Therefore, our LSH implementation found lightly revised articles candidate pairs which are false positive. It is clear that lack of specific labels that denote exact similarities of documents, makes this dataset not a very well evaluator. However, we have been able to conclude that LSH with Apache Beam works accurately considering the properties of corpus plagiarism dataset.

Running LSH Apache Beam On Our News Dataset

LSH Apache Beam implementation was run on news dataset which is 300 MB in total. LSH was run on Google Cloud Dataflow because of the demanding memory usage (8 GB) of LSH-Apache Beam implementation on 180.173 news articles. It is observed that before Mapper3 stage, the pipeline executes in 10 - 15 minutes and outputs <key = unique bands, value = list of candidate news>. This output represents the clusters of similar news.

Unfortunately, starting from Mapper3, to output <key = news pairs, value = number of similar news> it took around 2.5 hours and still did not complete. Therefore, the program was shut down due to billing concerns on Google Cloud Dataflow because Dataflow charges for computation usage and we were not able to estimate the billing. The reason is that there are at most c(180.000,2) = **324.000.000** pairs of news published. Approximately starting from the Mapper3 it will take 3.5 hours to output all similar pairs.

The output of GroupByKey was 100 MB which was the clusters of similar news. Although we have shut down rest of the pipeline and could not get all pairwise similarity counts of news sites, we could analyse this output in general. This set of clusters shows that cnnturk.com, star.com.tr and yenisafak.com have the most similar news with count 2198. This can be because of the fact that our dataset has most news from cnn.com.tr. Moreover, star.com.tr is in the top 5 in terms of news count in our dataset.

Future Work

Google Cloud Dataflow billing estimation will be made and the rest of the pipeline (Starting from Mapper3) will be executed once again if it is affordable.

Sentiment Classification

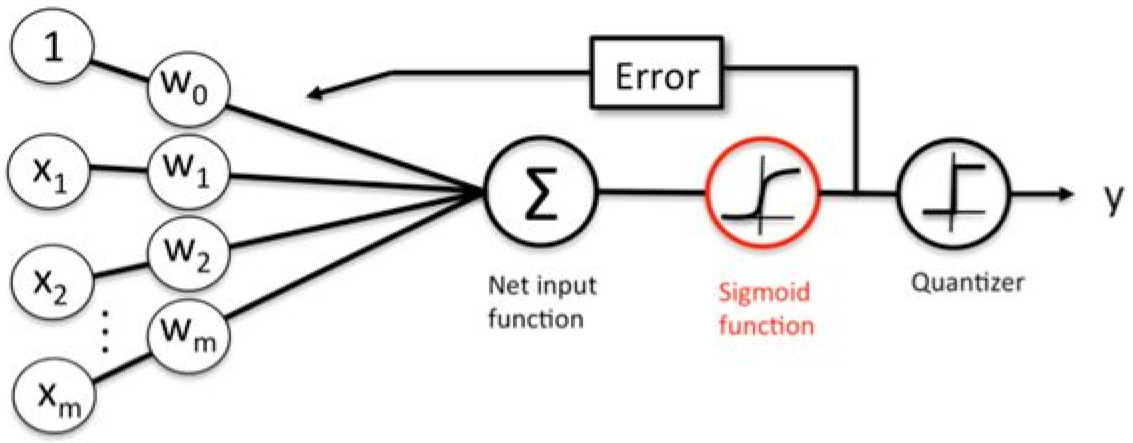
Task Definition

As people express their opinions via their writings or tweets, news Sources express their support/criticism via their articles. Although journals have to be objective, many of news sources are biased and do not follow this rule. Our target is to classify supportive articles as positive criticizing articles as negative and others as notr. Simply, we want to map a news article to the labels POSITIVE, NEGATIVE, and NOTR.

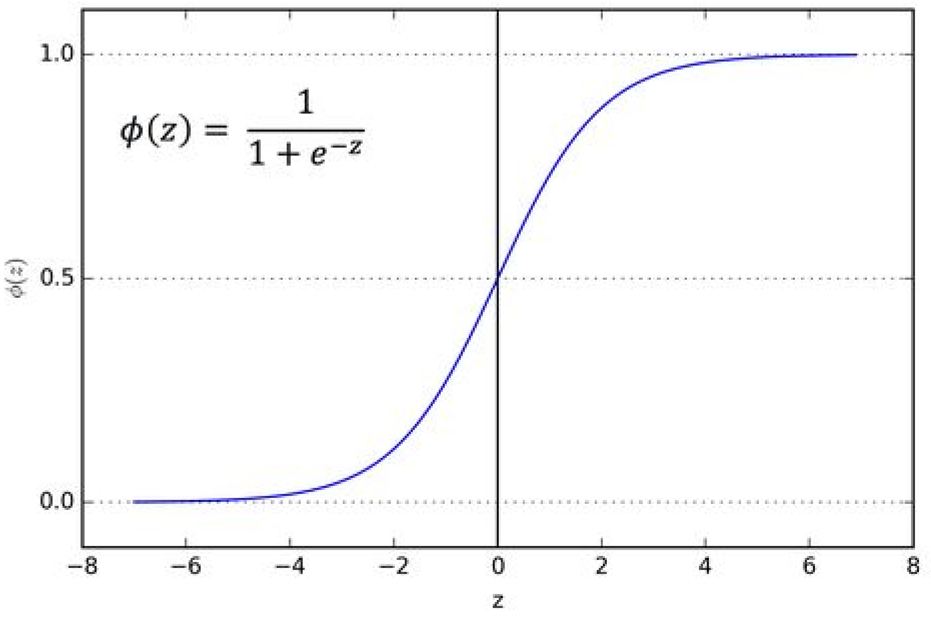
Turkish is a morphologically rich and an agglutinative language. In such languages data sparsity is a huge problem. This problem will be detailed Word2Vec subsection below.

Also, lack of enough data and lack of available NLP tools for Turkish makes this task harder.

Logistic Regression Algorithm



In logistic regression, the feature vector is multiplied with the transpose of normal vector of the fitting line. Then, a non-linearity which is the sigmoid function is applied to that result.



Since logistic regression function is continuous, it can easily be optimized via calculating and moving towards its gradient.

Stochastic Gradient Descent

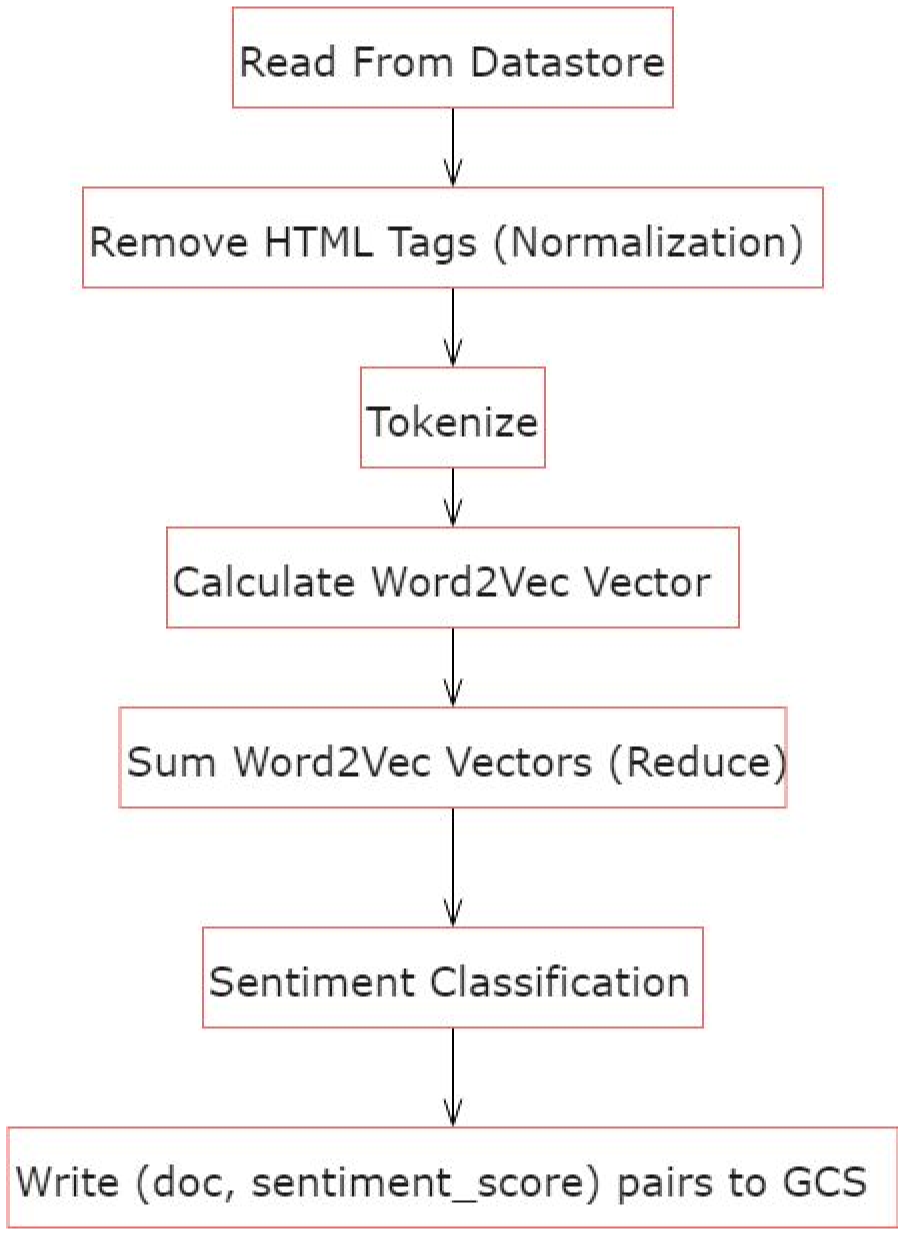
Normal vector of fitting line is moved at each iteration of introducing samples.

$$\begin{aligned}
g'_{\text{logistic}}(z) &= \frac{\partial}{\partial z}\left(\frac{1}{1+e^{-z}}\right) \\
&= \frac{e^{-z}}{(1+e^{-z})^2}(\text{chain rule}) \\
&= \frac{1+e^{-z}-1}{(1+e^{-z})^2} \\
&= \frac{1+e^{-z}}{(1+e^{-z})^2} - \left(\frac{1}{1+e^{-z}}\right)^2 \\
&= \frac{1}{(1+e^{-z})} - \left(\frac{1}{1+e^{-z}}\right)^2 \\
&= g_{\text{logistic}}(z) - g_{\text{logistic}}(z)^2 \\
&= g_{\text{logistic}}(z)(1 - g_{\text{logistic}}(z))
\end{aligned}$$

The derivative of sum of multiplication of w transpose and x vectors are appended to that by the chain rule.

Here, L loss function is L2, namely the sum of square of distances between actual values and estimates.

Running via Apache Beam Mapreduce


Read From Datastore
Remove HTML Tags (Normalization)
Tokenize
Calculate Word2Vec Vector
Sum Word2Vec Vectors (Reduce)
Sentiment Classification
Write (doc, sentiment_score) pairs to GCS

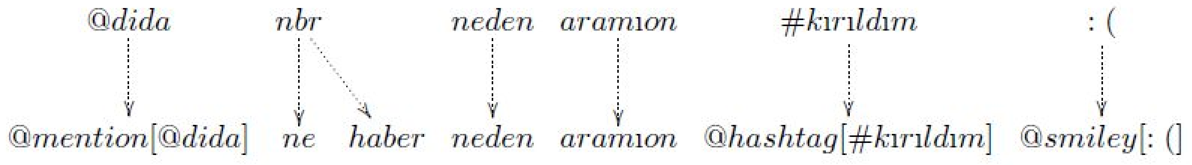
Note: Apache beam redirects some steps to the disks only when it is required. Thus, there is not much difference between map step and yielding foreach loop in redundant mappings. Reactive programming style is used to improve the quality and understandability of the implementation.

Pre-processing

Normalization

The labeled data for sentiment analysis are tweets which are very dissimilar in many ways. For example, tweets are very short in terms of text length compared to news articles. Tweets are much more informal than news articles mostly written by individuals, not journalists. Converting an informal text to a formal one with proper language usage is called normalization. Also, many tweets are not written with proper turkish characters since most people use English layout keyboard on their mobile phones.

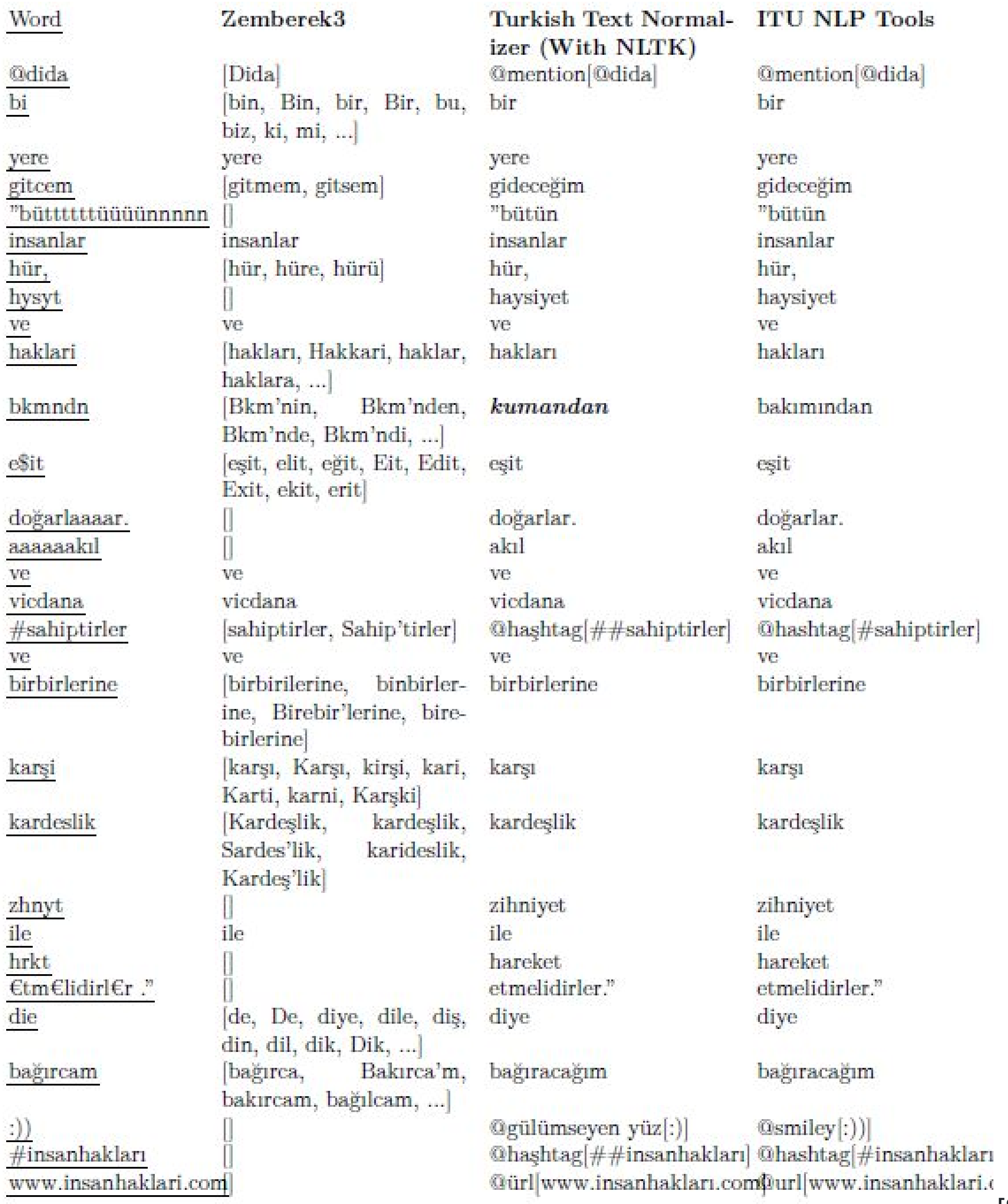
The normalization example for the tweet "@dida what’s up, why don’t you call #offended :(" is below:



[8]

To normalize turkish tweets in our 6000 tweets dataset, we use Turkish Tweet Normalizer library [11].

As we see no improvement on validation step when tweets are normalized, we wondered whether our text normalizer. To research this issue, we compared with popular Turkish NLP tools for normalization which are Zemberek3 [9] and ITU Normalization Service [10] as following:



Word	Zemberek3	Turkish Text Normalizer (With NLTK)	ITU NLP Tools
@dida	[Dida]	@mention[@dida]	@mention[@dida]
bi	[bin, Bin, bir, Bir, bu, biz, ki, mi, ...]	bir	bir
yere	yere	yere	yere
gitcem	[gitmem, gitsem]	gideceğim	gideceğim
"bütttttttüüüünnnnn	[]	"bütün	"bütün
insanlar	insanlar	insanlar	insanlar
hür,	[hür, hüre, hürü]	hür,	hür,
hysyt	[]	haysiyet	haysiyet
ve	ve	ve	ve
haklari	[hakları, Hakkari, haklar, haklara, ...]	hakları	hakları
bkmndn	[Bkm'nin, Bkm'nden, Bkm'nde, Bkm'ndi, ...]	***kumandan***	bakımından
e$it	[eşit, elit, eğit, Eit, Edit, Exit, ekit, erit]	eşit	eşit
doğarlaaaar.	[]	doğarlar.	doğarlar.
aaaaaakıl	[]	akıl	akıl
ve	ve	ve	ve
vicdana	vicdana	vicdana	vicdana
#sahiptirler	[sahiptirler, Sahip'tirler]	@haştag[##sahiptirler]	@hashtag[#sahiptirler]
ve	ve	ve	ve
birbirlerine	[birbirilerine, binbirler-ine, Birebir'lerine, bire-birlerine]	birbirlerine	birbirlerine
karşi	[karşı, Karşı, kirşi, kari, Karti, karni, Karşki]	karşı	karşı
kardeslik	[Kardeşlik, kardeşlik, Sardes'lik, karideslik, Kardeş'lik]	kardeşlik	kardeşlik
zhnyt	[]	zihniyet	zihniyet
ile	ile	ile	ile
hrkt	[]	hareket	hareket
€tm€lidirl€r ."	[]	etmelidirler."	etmelidirler."
die	[de, De, diye, dile, diş, din, dil, dik, Dik, ...]	diye	diye
bağırcam	[bağırca, Bakırca'm, bakırcam, bağılcam, ...]	bağıracağım	bağıracağım
:))	[]	@gülümseyen yüz[:)]	@smiley[:))]
#insanhakları	[]	@haştag[##insanhakları]	@hashtag[#insanhakları
www.insanhaklari.com	[]	@ürl[www.insanhakları.com	@url[www.insanhaklari.c

[8]

These results do not indicate any problem with the library we use(Turkish Text Normalizer (With NLTK)). Thuss, we believe that normalization removes some information about sentiments. However, for our task we will apply normalization because normalized tweets are obviously more like the news data.

Sentence Tokenization

Sentence tokenization is simply the separating different sentences in a text from each other. In medium-sized writings such as news articles, sentences are usually seperated with dots. However, there are many abbreviations that use dot (.) to indicate abbreviation in Turkish language.

Thus, we use pre-trained Punkt sentence tokenizer model [13] for Turkish language and trained it more largest Turkish abbreviations corpus used by Zemberek3 [9].

Word Tokenization

Word tokenization is simply the separating different words/punctuations etc. in a text from each other. Word tokenization is rather simple. Splitting via spaces are enough. Also, punctuations, negating suffix(-me, -ma) are considered as separate tokens in our implementations. NLTK library [12] handles such cases.

Stemming

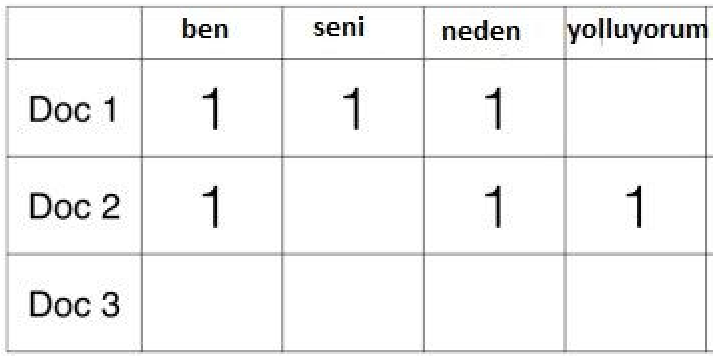
Stemming is to obtain the surface form of the words. Since turkish is an agglutinative language, this process would reduce the vocabulary size and may result in better results.

To stem a word, Turkish Stemmer library [15] is used.

Features

Bag-of-Words

Bag-of-words is a matrix containing the number of occurrences of a token in text.



	ben	seni	neden	yolluyorum
Doc 1	1	1	1	
Doc 2	1		1	1
Doc 3				

Tokens may be words as well as n word shingles. n-shingles are the consecutively occurred n words in text.

Tokens may also be stemmed or normalized as stated in any experimentation detail.

TF-IDF

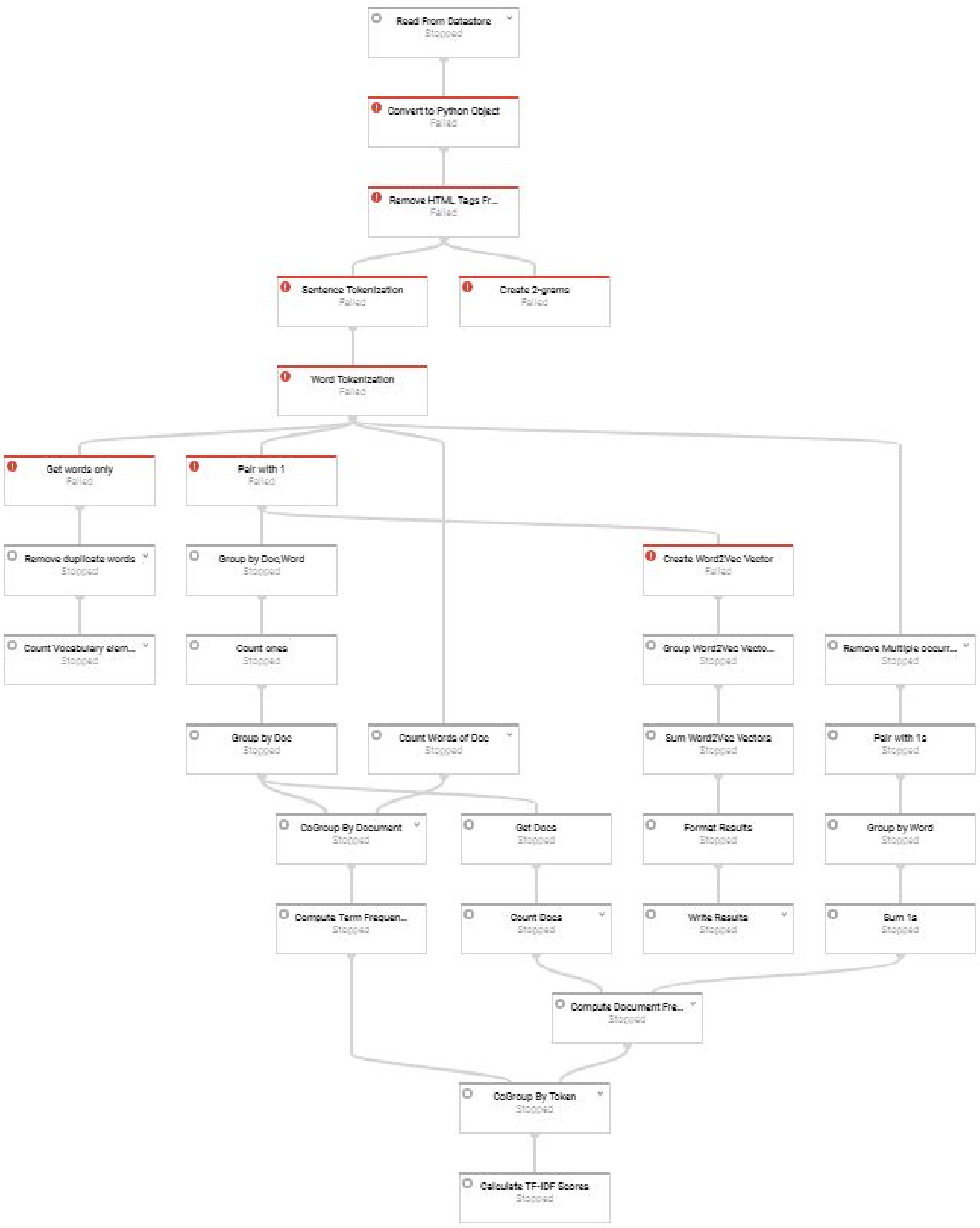
TF-IDF is an information retrieval technique that measures the relative importance of tokens. The metrics used are term frequency(tf) and inverse document frequency(idf).

$$TF(i,j) = \frac{\text{Term i frequency in document j}}{\text{Total words in document j}}$$

$$IDF(i) = \log_2\left(\frac{\text{Total documents}}{\text{documents with term i}}\right)$$

TF-IDFs are useful on retrieving the tokens that determine the context of a particular text as well as eliminating stop words.

Below is the mapreduce implementation of tf-idf:



Here, we tried to run via dataflow but some steps seems failed since the error caused by word2vec related stuff. In fact, later we switched to Google Cloud Compute Instance to run Mapreduce implementation of TF-IDF.Note: Apache beam redirects some steps to the disks only when it is required. Thus, there is not much difference between map step and yielding foreach loop in redundant mappings. Reactive programming style is used to improve the quality and understandability of the implementation.

Note: Apache beam redirects some steps to the disks only when it is required. Thus, there is not much difference between map step and yielding foreach loop in redundant mappings. Reactive programming style is used to improve the quality and understandability of the implementation.

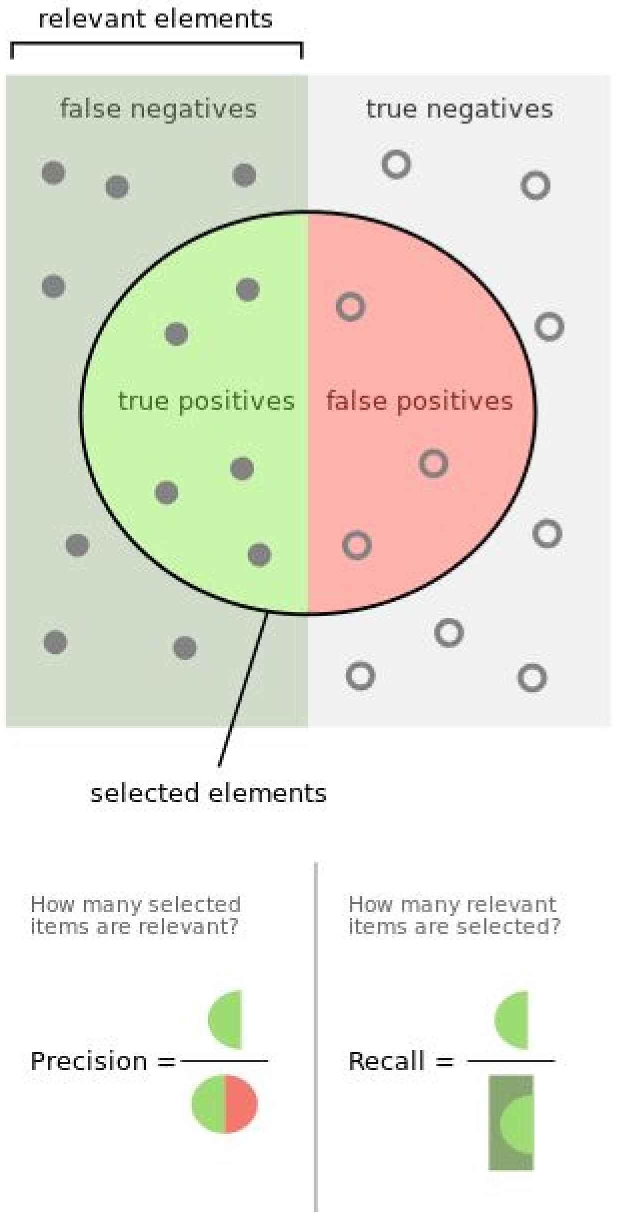
Word2vec

As we discussed in the task definition of sentiment classification, morphological richness property of Turkish language causes data sparsity problem. The matrices we obtained for bag of words and tf-idf are very sparse. For example, 6000 tweets sentiment classification dataset has 26000 words in total. This is a large number to compute and also very low number for a model to be able to classify. It is not possible to classify out-of-vocabulary words using bag-of-words and tf-idf features. We can decrease the out-of-vocabulary words in any text via stemming and normalization. However, a real solution to this problem is using pre-trained Word2Vec models as nearly all researchers do.

Thus, we prefered to switch Word2Vec model [14] pre-trained on 800k words wikipedia data. This provides us much larger corpus containing 800000.

Experimentation Methodology

To make experiments on, 6000 tweets dataset is used. We first made experiments using sklearn [6] library in order to be sure about the implementations we tested are correct and easily validate them. Later, we implement the best fitting model in matrix library numpy [7] and try to get same results with this.Decision tree, logistic regression, nearest neighbor, SVM classifiers are evaluated on this dataset using SKLearn library.



High precision means most of the items classified as a label, this label is most probably the correct one.

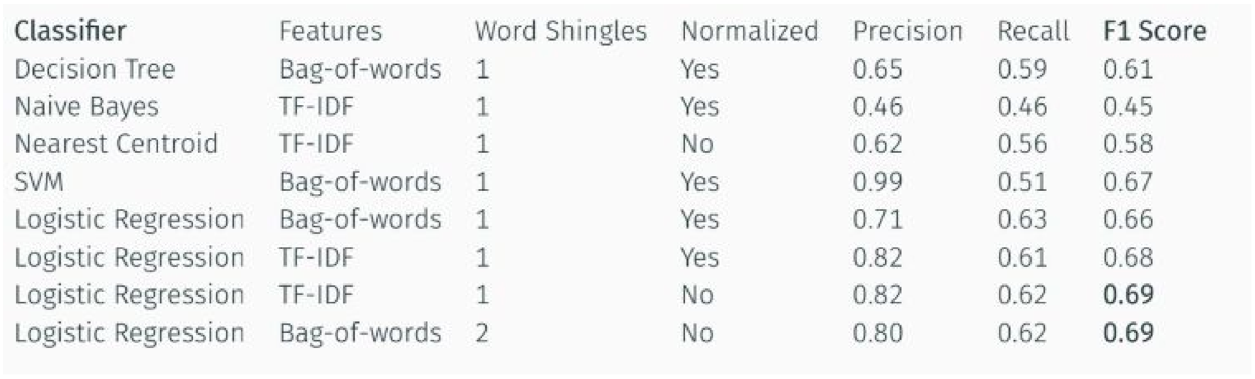
High recall means most of the items that needed to be labelled are labelled correctly.

The only reason why we use the harmonic mean is because we're taking the average of ratios (percentages), and in that case the harmonic mean is more appropriate than the arithmetic mean.

To compare our results with state-of-art results, we needed to use F1 scores as is in these research papers.

In our classifications, for each class, the true positive, true negative, false positive, false negative results are calculated and F1 score is computed. Then, F1 scores of all classes is averaged.

Experimentation Results

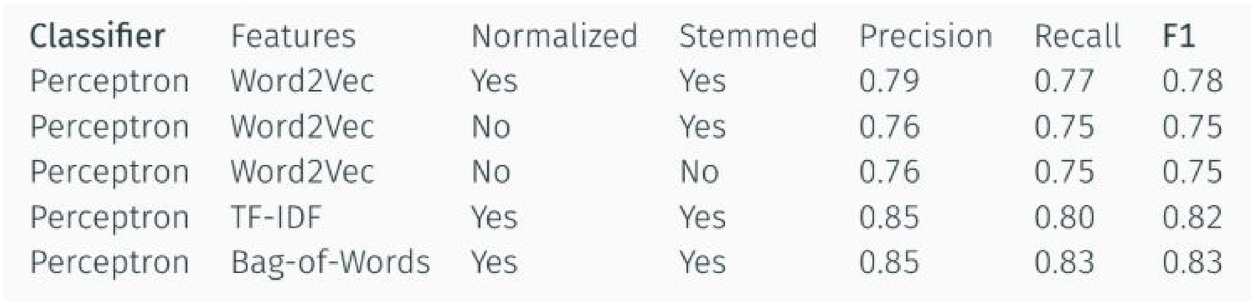


Classifier	Features	Word Shingles	Normalized	Precision	Recall	**F1 Score**
Decision Tree	Bag-of-words	1	Yes	0.65	0.59	0.61
Naive Bayes	TF-IDF	1	Yes	0.46	0.46	0.45
Nearest Centroid	TF-IDF	1	No	0.62	0.56	0.58
SVM	Bag-of-words	1	Yes	0.99	0.51	0.67
Logistic Regression	Bag-of-words	1	Yes	0.71	0.63	0.66
Logistic Regression	TF-IDF	1	Yes	0.82	0.61	0.68
Logistic Regression	TF-IDF	1	No	0.82	0.62	**0.69**
Logistic Regression	Bag-of-words	2	No	0.80	0.62	**0.69**

These models are biased over negative tweets to increase their accuracy. Negative tweets are easier to classify for all type of classifiers whereas notr tweets are the hardest to classify.

We get most accurate results with SVM and logistic regression. However, SVM is too slow to be work on larger data such as our news dataset.

Thus, we chose to implement Logistic Regression using numpy matrix computation library [7]. Below is the logistic regression experimentation results



Classifier	Features	Normalized	Stemmed	Precision	Recall	F1
Perceptron	Word2Vec	Yes	Yes	0.79	0.77	0.78
Perceptron	Word2Vec	No	Yes	0.76	0.75	0.75
Perceptron	Word2Vec	No	No	0.76	0.75	0.75
Perceptron	TF-IDF	Yes	Yes	0.85	0.80	0.82
Perceptron	Bag-of-Words	Yes	Yes	0.85	0.83	0.83

Here, perceptron classifier is the name for logistic regression classifier in neural networks literature. Although we get highest scores with Bag-of-words model, we do not use it in production. Because it requires more memory and time consuming computations than Word2Vec model.

Related Work

There are not many work done in this area. Kaya et. al. [5] made many experiments on political news 82.01 F1 score at best. However, our results are not comparable with theirs.

Future Work

There is definitely lack of labeled data on news articles for sentiment analysis. Labeling new datasets for Turkish news context would be the best contribution to the field. This process would also help for generalization of the sentiment classification task in news domain.Eliminating objective sentences would indicate the objectivity of news sources. Namely, objectivity-subjectivity of a news source can be determined.

Named Entity Recognition

Task Definition

In this task we want to classify named entities.

Sequence to sequence labeling is a common problem in Natural Language Processing literature. Named entities are real world object with proper names such as people, locations, organizations.

Named entity recognition is challenging task. Named entities can be defined differently in different human labelers. Also, substrings of some named entity may also be another named entity. For example, "Mustafa Kemal Caddesi" is a location whereas "Mustafa Kemal" is a person. Also, context awareness is required since "İpek" may be a person name as well as a product name.

For NER Task we implement a similar architecture to the one proposed by Kuru et al. [20] which is discussed in Related Work section of NER. In this project, we only extract PERSON, LOCATION, ORGANIZATION named entities.

Character Embeddings

One-hot character vectors are converted to distributed representation of characters. Embeddings are learned during training of the network. This is simply a dense layer of neurons that learn the distributed representation of characters. This kind of embedding would highly reduce data sparsity

LSTM

Recurrent neural networks(RNNs) achieved state-of-art results in many languages.

However, original RNNs are One type of them are Long Short Term Memory which is originally created to overcome exploding gradient problem during training.

In LSTM cells, the following equations are defined:

$$\begin{aligned}
f_t &= \sigma(\boldsymbol{W}_{fx}x_t + \boldsymbol{W}_{fh}h_{t-1} + w_{fc} * c_{t-1} + b_f) \\
i_t &= \sigma(\boldsymbol{W}_{ix}x_t + \boldsymbol{W}_{ih}h_{t-1} + w_{ic} * c_{t-1} + b_i) \\
c_t &= f_t * c_{t-1} + i_t * \tanh(\boldsymbol{W}_{cx}x_t + \boldsymbol{W}_{ch}h_{t-1} + b_c) \\
o_t &= \sigma(\boldsymbol{W}_{ox}x_t + \boldsymbol{W}_{oh}h_{t-1} + w_{oc} * c_t + b_o) \\
h_t &= o_t * \tanh(c_t)
\end{aligned}$$

Left hand sides of these equations are called forget gate, input gate, output gate, cell vector, and hidden state vector, respectively.

σ is the sigmoid function. Uppercase W letters are matrices, indicating weights subscripted with target and , respectively. These equations can be optimized via stochastic gradient descent we described in Sentiment Classification section.

Bidirectional LSTM

Success of Bidirectional LSTMs in NER tasks show that being a named entity depends on future character sequences as well as future character sequences. Simply, bidirectional LSTMs are the way to consider future LSTM sequences.

$$i_t = \sigma(\boldsymbol{W}_{ix}x_t + \boldsymbol{W}_{ih}h_{t+1} + w_{ic} * c_{t+1} + b_i)$$

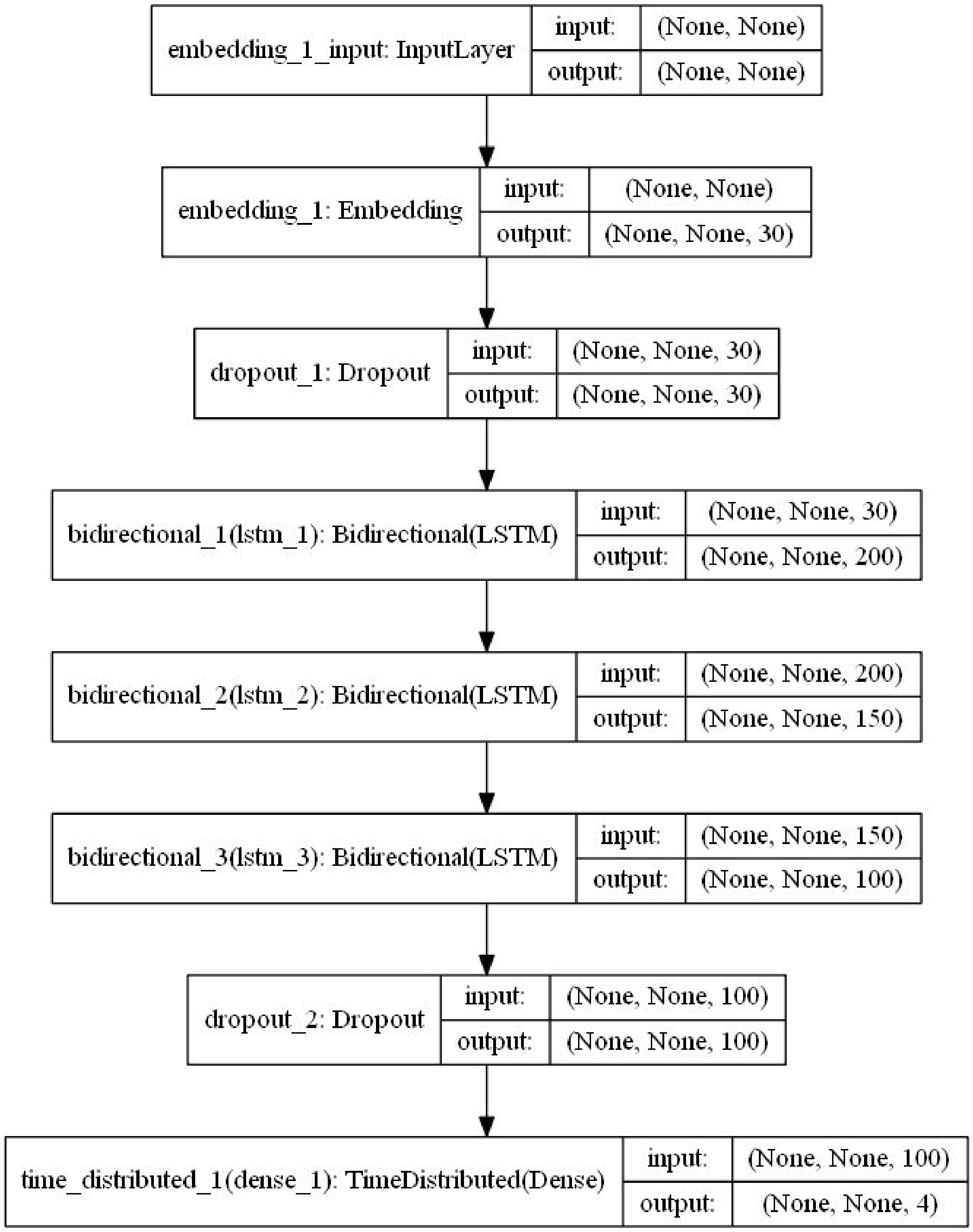
Here, we add the features from timestamp (t+1) to the input gate to make our LSTM bidirectional.

Softmax

We have labels which PERSON, ORGANIZATION, LOCATION, and NONE. This is why the last dimension of softmax layer is 4. For each label, we want to assign a probability to that label. In this case, we replace an activation function such as logistic function with softmax function.

$$P(y = j \mid z^{(i)}) = \phi_{softmax}(z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{j=0}^{k} e^{z_k^{(i)}}},$$

Our Architecture



Nones in model plot indicate variable lengths.

Character embeddings are connected to stacked 3 bidirectional lstm layers and at the end Softmax layer gives the probabilities of sequence of named entities for the words of sentence.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 30)          900
_________________________________________________________________
dropout_1 (Dropout)          (None, None, 30)          0
_________________________________________________________________
bidirectional_1 (Bidirection (None, None, 200)         104800
_________________________________________________________________
bidirectional_2 (Bidirection (None, None, 150)         165600
_________________________________________________________________
bidirectional_3 (Bidirection (None, None, 100)         80400
_________________________________________________________________
dropout_2 (Dropout)          (None, None, 100)         0
_________________________________________________________________
time_distributed_1 (TimeDist (None, None, 4)           404
=================================================================
Total params: 352,104
Trainable params: 352,104
Non-trainable params: 0
_________________________________________________________________
```

Experimentation Methodology

Since the network is time consuming to train, we could not make many experiments. However, Kuru et al. [20] achieves state-of-art results with similar neural network architecture. We implemented similar neural network to and experimented on same dataset as Kuru et al. [20]

To make experiments 35000 sentences NER dataset High precision means most of the items classified as a label, this label is most probably the correct one.

High recall means most of the items that needed to be labelled are labelled correctly.

The only reason why we use the harmonic mean is because we're taking the average of ratios (percentages), and in that case the harmonic mean is more appropriate than the arithmetic mean.

To compare our results with state-of-art results, we needed to use F1 scores as is in these research papers.

$$F_1 = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Experimentation Results

We could not exactly reproduce the results in CharNER [20] architecture probably because we do not have Viterbi decoder. However, we obtained significant results for Named Entity Recognition.

Training is stopped at 89.61 F1 score on validation set When we test that model on test dataset F1 is 82.37.

Lower than original implementation of CharNER [20] which is 91.30 but seems sensible for our project. Nevertheless, the reasons for that result are discussed in "Related Work" section below.

Related Work

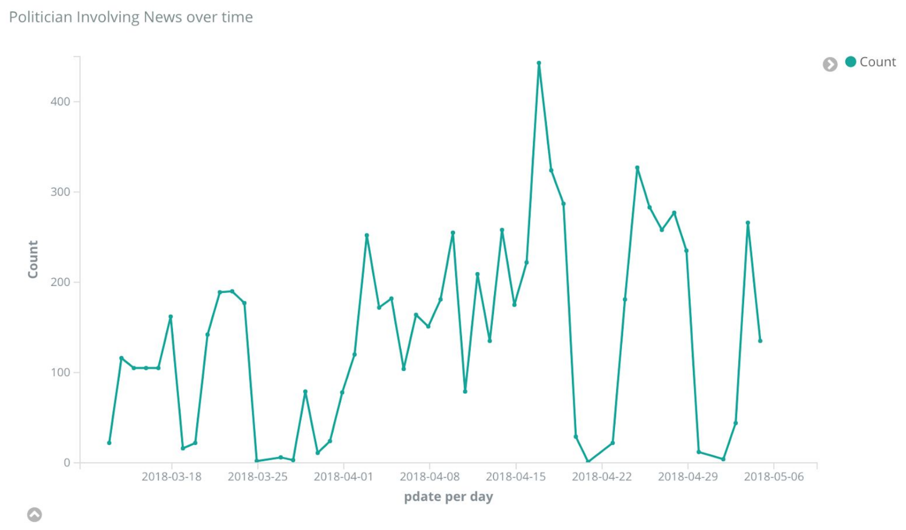
In CharNER architecture proposed by Kuru et al. [20], they mapped each character to a label. Then, they extracted the most probable Named Entity sequence for words of the sentence from these labels via Viterbi Decoder. In our implementation we used a softmax layer instead of the Viterbi decoder. As Kuru et al. [20] discusses we got lower results than the CharNER implementation proposed by them.

Future Work

Viterbi decoder, or Random Forest, or SVMs can be used after softmax layer. In published paper of Kuru et al. [20], they use Viterbi decoder to make some rule based decisions. For example, the continuation of a named entity cannot be at the beginning of that named entity. This decoder increases the F1 scores by ~2% as they indicate.

In Depth Analysis of Our Results:

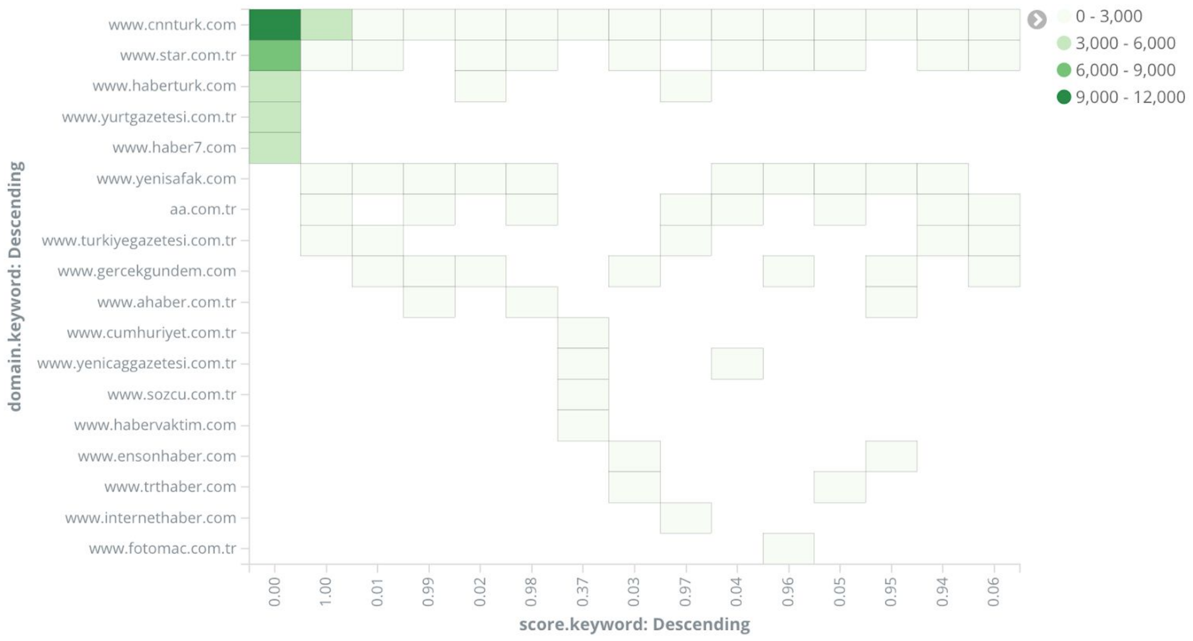
In order to analyse our results in terms of time, we have visualized politician related news(Kılıçdaroğlu, Erdoğan, Bahçeli, Akşener, etc.) with respect to time as below:



It is clear that there is a peak after the decision for early election in Turkey. However, on 22th of April, our scraper was sleeping so it seems like there is a decrease but it is not true. Overall the increase in the amount of news related with politicians is obvious.

Sentiment Score Analysis

Moreover, we have analyzed our sentiment scores for news articles. In order to analyse better, we have visualized the heatmap for news sites with respect to their sentiment scores as below:



It is clear that cnnturk.com and star.com.tr have more negative news compared to other news sites because the count of their sentiment score having 0 is very high. This can be because of the fact that cnnturk.com and star.com.tr have the most number of news in the dataset. Moreover, the count of negative news dominates count of positive news as it can be viewed from the heatmap.

Named Entity Analysis

In addition, we have analyzed our results for named entities in the following subsections.

Named Entity Clouds

In order to view investigate which entities were more used we have plotted tag clouds of named entities. The following clouds show the named entities proportionally bigger with respect to their counts.

Location Entity Cloud:

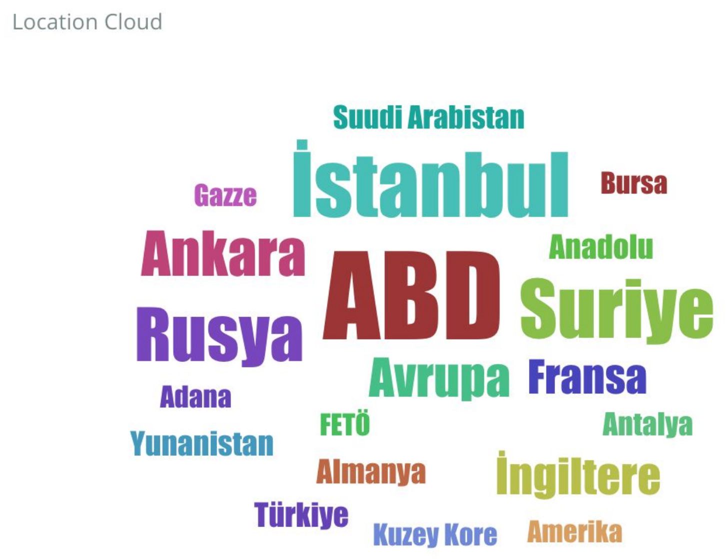


Figure: Location Entity Cloud

It is clear that *ABD*, *İstanbul*, *Rusya* and *Suriye* dominates other locations in terms of count. This can be because of the fact that news related to heat between ABD and Rusya regarding Suriye.

There also some false positives such as *FETÖ* is found to be a location but these false positives are negligible.

Organization Entity Cloud:

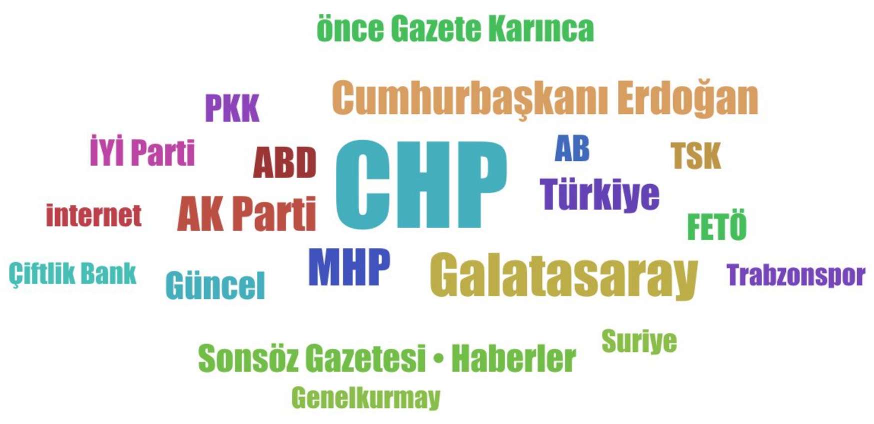


Figure: Organization Entity Cloud

It is clear that *CHP* dominates other organization in terms of news count. This can be because the early election is approaching and *CHP* is trending in the news. Moreover, *Galatasaray* seems to be dominating other sports organizations, this can be because galatasaray is the leader in the Super Lig so there are more news about them. Ak Parti is not as popular as CHP probably because Recep Tayyip Erdoğan dominates the place of Ak Parti as seen in the person entity cloud below.

There are also some false positives such as *Cumhurbaşkanı Erdoğan* but it is negligible as most of them are correct

Person Entity Cloud:

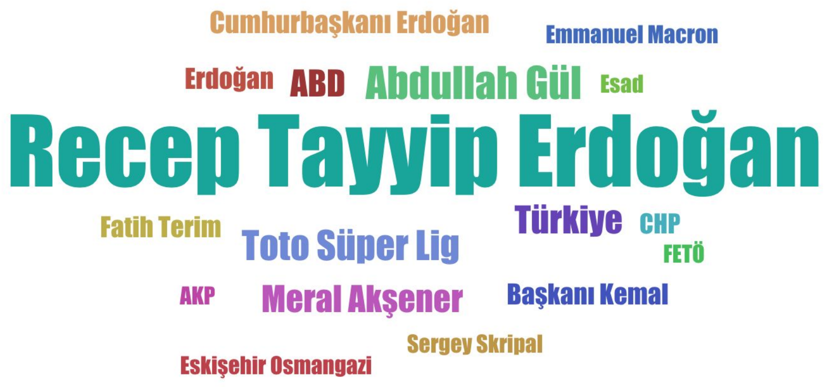


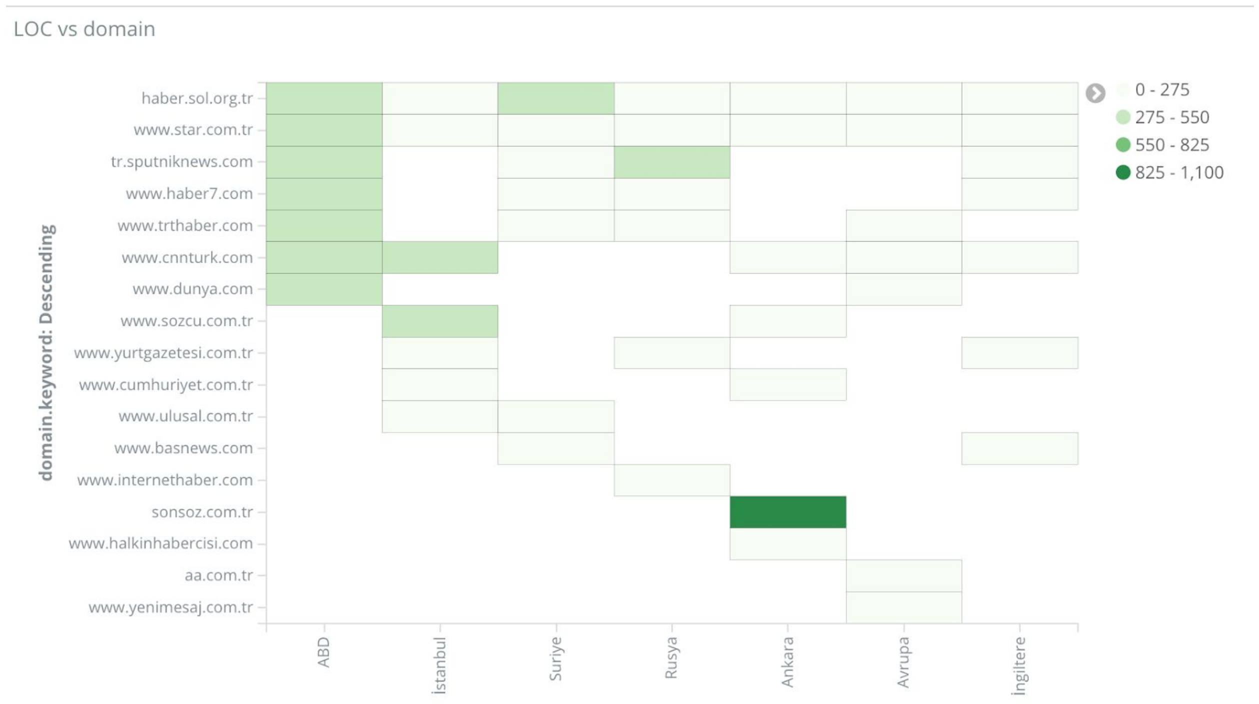
Figure: Person Entity Cloud

It is clear that *Recep Tayyip Erdoğan* dominates other people in terms of number of news related to them. We can definitely conclude that news sites are not publishing equally distributed news about people and they are biased. Moreover, Meral Akşener seems to be trending among other politicians. This can be because of the fact that she is president candidate of newly-established İYİ parti and there are many news related to her.

Named Entity Heatmaps

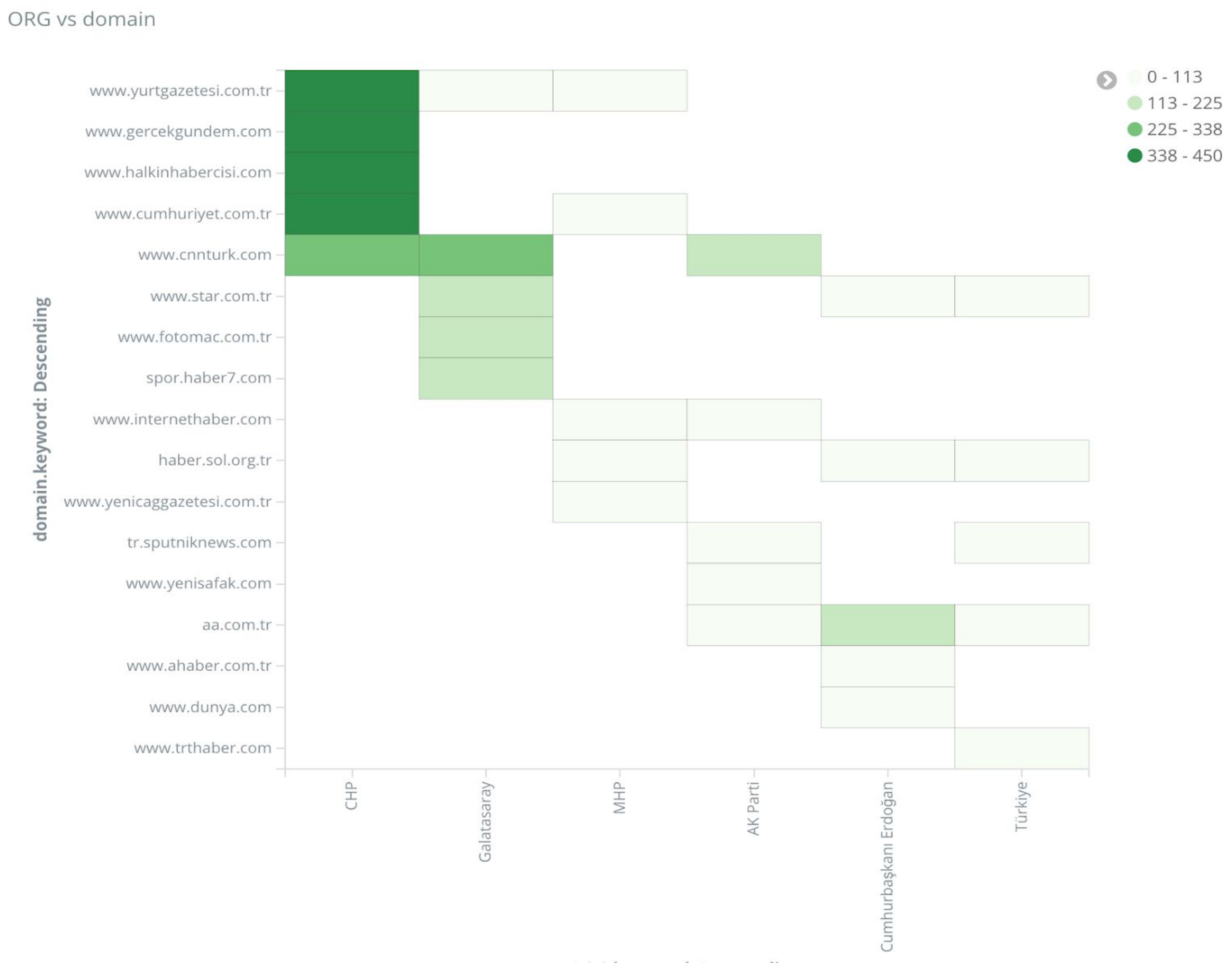
In order to analyse named entities better, we have plotted the heatmap of named entities with respect to news sites. These heatmaps plot the number of news having the specified named enitity(location, organization, etc). This visualization makes it easier to analyze our results in terms of news sites.

Location Named Entities and News Sites Heatmap



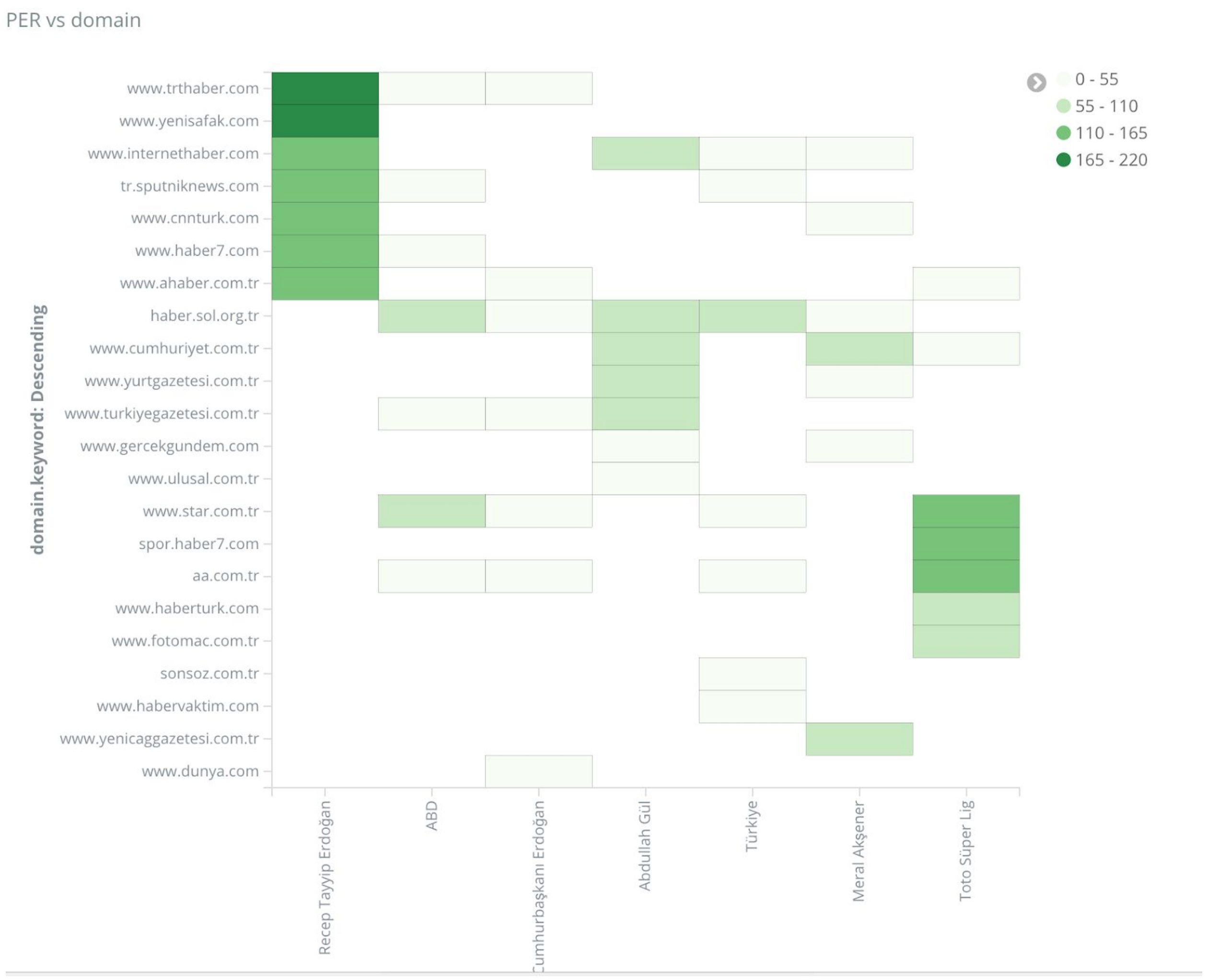
It is clear that ABD dominates other locations in the news. News that had ABD as location were generally published by news sites that publish international news as well. As united states is one of the biggest countries in the world, this result seems to be logical. Moreover, Habersol.org have made more news related with Suriye compared to other news sites which can indicate that either they are biased ether towards or against Suriye.

Location Named Entities and News Sites Heatmap



It is clear that yurtgazetesi, gercekgundem, halkinhabercisi and cumhuriyet are biased towards *CHP* as they have published many news related with chp. Moreover, cnn seems to be biased towards *Ak Parti* as they have more news compared to other news sites. Some false positives in location named entities can be observed as *Cumhurbaşkanı Erdoğan* is labeled as location entity.

Person Named Entities and News Sites Heatmap:

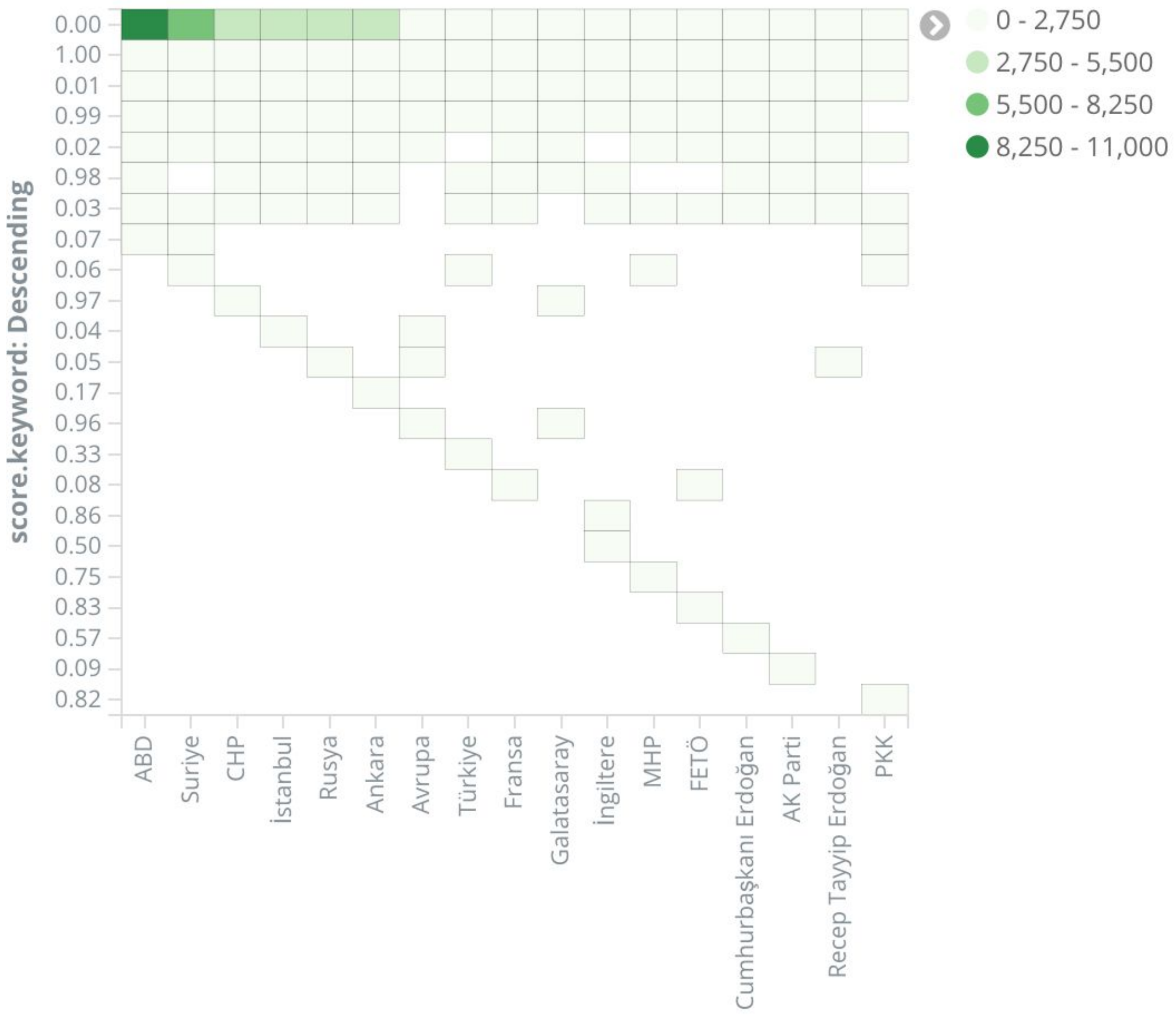


It is obvious that trthaber, yenisafak, internethaber, cnnturk, haber7 are biased towards *Recep Tayyip Erdoğan.* Moreover, it seems like news site that are generally supporting main opposition party preferred to use *Cumurbaşkanı Erdoğan* instead of Recep Tayyip Erdoğan. In addition, the same news sites also published news related to *Abdullah Gül* more than other sites.

There also some false positives in our person named entities such as *Spor Toto Süper Lig* and *Türkiye.*

Named Entity vs Sentiment Scores

The heatmap of named entities with respect to sentiment scores of their related news is plotted in order to analyse the relation between named entities and sentiment scores. Notice that the sentiment scores are not sorted, it could not be sorted in Kibana. Moreover, sentiment score 0 denotes most negative content whereas 1 denotes most positive.



It is clear that ABD and Suriye has the most number of negative content news as the heat map shows dark green for sentiment value 0. This can be because of the war in Suriye. Moreover, CHP has the most number of negative content news. This shows that generally news sites are biased against CHP compared to other parties such as MHP, Ak Parti and etc.

Technologies & Environment

- Language: Python 3.5.4 & Python 2.7 (For Apache Beam & Google Cloud Dataflow)
- Environment: Anaconda 4.4.7
- Jupyter Notebook [18]
- Google Cloud Datastore
- Google Cloud Compute
- Google Cloud Dataflow
- Apache Beam [3]
- news-please [4]
- numpy [7]
- sklearn [16]
- keras [17]

Related Work

Kaya et. al. [5] states that as a future work, determining which columnists write about which party/person and determine their sentiment(supportive or criticisive behavior). This is what we did in this project.

Future Work

Initially, our plan was to build real time version of this project. Apache beam is also capable of working on data streams such as stream of news like ours. In order to build that kind of architecture Google Cloud Pub/Sub can be used. When a new item is written to Google Cloud Datastore, a new event is published. Then, apache beam runners listens that event and indexes these results to the ElasticSearch to be visualized by Kibana.

Work Distribution

Our group was initially consist of 4 people: Berk Mandıracıoğlu, Selim Fırat Yılmaz, Ömer Kurun, Arda Kıray. However, Ömer Kurun withdrew the course. Later, Arda Kıray was not communicating with us and he was not coming to the classes.

Therefore, as Selim Fırat Yılmaz, and Berk Mandıracıoğlu we created this project as a group of 2 people. Berk Mandıracıoğlu implemented the LSH with Apache Beam and Kibana Data Visualization parts. Selim Fırat Yılmaz implemented the Data Crawling, Sentiment Classification and Named Entity Recognition parts. For the report and presentation, we both prepared & presented our individual parts separately and merged them.

Bibliography

[1] https://github.com/selimfirat/bloom-filtering

[2] Leskovec, J., Rajaraman, A. and Ullman, J. (2016). Mining of massive datasets. Delhi: Cambridge University Press, pp.140-141.

[3] https://github.com/fhamborg/news-please

[4] Beam, A. An advanced unified programming model. 2016-12-02]. https://beam.apache.org.

[5] Kaya, M., Fidan, G., & Toroslu, I. H. (2012, December). Sentiment analysis of turkish political news. In Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01 (pp. 174-180). IEEE Computer Society.

[6] Cournapeau, D. (2015). Sci-kit Learn. Machine Learning in Python. online,[cit. 8.5. 2017]. URL http://scikit-learn.org.

[7] Walt, S. V. D., Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. Computing in Science & Engineering, 13(2), 22-30.

[8] Torunoğlu, D., & Eryiğit, G. (2014). A cascaded approach for social media text

normalization of Turkish. In Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM) (pp. 62-70).

[9] Akın, A. A., & Akın, M. D. (2007). Zemberek, an open source nlp framework for turkic languages. Structure, 10, 1-5.

[10] Eryiğit, G. (2014). ITU Turkish NLP web service. In Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (pp. 1-4).

[11] https://github.com/uozcan12/Twitter-Data-Normalization

[12] Bird, S., & Loper, E. (2004, July). NLTK: the natural language toolkit. In Proceedings of the ACL 2004 on Interactive poster and demonstration sessions (p. 31). Association for Computational Linguistics.

[13] https://www.kaggle.com/nltkdata/punkt

[14] https://github.com/akoksal/Turkish-Word2Vec

[15] https://github.com/otuncelli/turkish-stemmer-python

[16] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

[17] Chollet, F. (2015). Keras.

[18] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., ... & Ivanov, P. (2016, May). Jupyter Notebooks-a publishing format for reproducible computational workflows. In ELPUB (pp. 87-90).

[19] https://github.com/sercankulcu/twitterdata

[20] Kuru, O., Can, O. A., & Yuret, D. (2016). Charner: Character-level named entity recognition. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers (pp. 911-921).

[21]https://ir.shef.ac.uk/cloughie/resources/plagiarism_corpus.html#Download