İSTANBUL TECHNICAL UNIVERSITY
FACULTY OF COMPUTER AND INFORMATICS

# COMPANY RELATION EXTRACTION

**Graduation Project Report**

**Baran Kaya**
**150130032**
**Mehmet Ali Osman Atik**
**150140804**

**Department: Computer Engineering**
**Division: Computer Engineering**

**Advisor: Yrd. Doç. Dr. Gülşen Cebiroğlu Eryiğit**

July 2018

# Statement of Authenticity

We hereby declare that in this study

1. all the content influenced from external references are cited clearly and in detail,
2. and all the remaining sections, especially the theoretical studies and implemented software that constitute the fundamental essence of this study is originated by our individual authenticity.

İstanbul, July 27, 2018

Baran Kaya

Mehmet Ali Osman Atik

# COMPANY RELATIONS EXTRACTION (SUMMARY)

This project focuses on one of the most crucial concepts of the banks known as "Know your customer". Within an agreement with ITU, Yapı Kredi Technology started this project as a graduation project. Main purpose of the project is to automate finding company relations with other companies from websites. Banks' are tracking their company customers' actions and decisions of company management. For that purpose, several bank employees spend a lot of time on internet for searching the news related to the bank's company customers. This project aims to automate this job with using NLP (Natural Language Processing) and Machine Learning (ML) techniques on Turkish texts.

Finding clean and high-quality web sources was the first step of the project. Project's initial web sources were Ticari Sicil Gazetesi (TSG), Ticaret Odası Kayıtları (TOK), Kamuyu Aydınlatma Platformu (KAP), Rekabet Kurulu (RK), newspaper web pages and social media. Due to the non-formal language in the social media, these sources were held out of scope. Other websites were researched and found out that TSG and TOK platforms contains only the general company informations (ie. address, phone) and these two websites do not preserve any company relation informations. On the other hand, KAP contains company relations, so for extracting relation data KAP was the best option. Nevertheless, KAP only contains companies that are in the stock market BİST (Borsa İstanbul). On RK website informations about mergers and acquisitions (takeovers) are usually mentioned within a sentence. Yet informations in RK website will already be published in KAP therefore, KAP seems to be enough as a data source for extracting company relations. As a next step web crawler was written for collecting newspaper data from their web sites in addition to the KAP. The purpose of adding news as a source was collecting information about companies that are not in BIST. Also, adding news supplied more data for training and testing sets.

This project consists of two phases. The first one is analyzing text and finding company names. On NLP, usually Conditional Random Fields [1] are used for segmenting and labeling sequential data. There are various implementations of CRF and within these, CRF++ [2] have been chosen for labeling company names. The second one is extracting the relations in the text. This can be done by classifying data by looking the features in it. A well-known technique for classification is to use Support Vector Machines [3]. In this project a popular SVM implementation LibSVM [4] have been used for classifying relations.

In this project, KAP data used for training models and newspaper data that obtained from web used for testing these models. For language processing part, data were transformed to the CoNLL 2003 format which CRF++ needs for train and test processes. After transforming the data, company names were tagged with search algorithms and CRF++ was used for generating models for the tagged data. Then these models were used to test KAP and newspaper data.

But for extracting relations between companies, the formal language of KAP was not good for classifying the casual language of newspaper data. So, more newspaper data were added and tagged for increasing newspaper data size to use with LibSVM to extract these relations.

After trying different approaches and testing with various parameters, project accuracy was reached to 95% for tagging company names and 82% for labeling relations on our test sets.

# ŞİRKET İLİŞKİLERİ ÇIKARMA (ÖZET)

Bu proje bankacılığın en önemli konseptlerinden biri olan “Müşterini tanı” üzerinedir. Bir Yapı Kredi Teknoloji projesi olarak başlayan bu proje daha sonra İTÜ’ye bitirme projesi olarak verilmiştir. Projenin asıl amacı banka şirket müşterilerinin ilişkilerini internet kaynaklarından bulmayı insan elinden çıkarıp otomatik hale getirmektir. Bankalar şirket müşterilerinin faaliyetlerini ve şirket hakkındaki kararlarını takip etmektedir. Bu amaçla bankada sadece bu iş üzerine çalışan insanlar internette şirketler hakkında haberleri takip etmek için çok fazla zaman harcamaktadır. Bu proje bu işi otomatik bir hale getirmek için Türkçe metinler üzerinde Doğal Dil İşleme (Natural Language Processing) ve Makine Öğrenmesi (Machine Learning) tekniklerini kullanmaktadır.

Temiz ve kaliteli bir internet kaynağı bulmak projenin ilk adamıydı. Projenin ilk internet kaynakları Ticari Sicil Gazetesi (TSG), Ticaret Odası Kayıtları (TOK), Kamuyu Aydınlatma Platformu (KAP), Rekabet Kurulu (RK), web haberleri ve sosyal medya idi. Sosyal medyada kullanılan dilin resmi ve düzgün olmaması nedeniyle ilk elenen kaynak sosyal medya oldu. Ayrıca diğer internet siteleri üzerinde bir miktar araştırma yapıldıktan sonra TSG ve TOK sitelerinin sadece şirketlerin adres ve telefon gibi genel bilgilerini içerdiklerini ve bu sitelerin herhangi bir şirket ilişkisi içermediği görüldü. Ancak KAP sitesi şirket ilişkilerini bulunduruyordu ve bu iş için en iyi kaynağın KAP olduğu anlaşıldı. Öte yandan KAP sadece Borsa İstanbul (BİST) içerisinde olan şirketlerin bilgilerini içeriyordu bu sebeple diğer kaynaklar da incelenmeye devam edildi. Rekabet Kurulu sitesinde şirket birleşme ve satın alma haberleri bulunuyordu fakat bu ilişkiler sadece 1 cümle içinde geçmekteydi. Ayrıca Rekabet Kurulu’ndaki bilgiler KAP sitesinde de yayınlanıyordu. Bundan dolayı KAP, şirket ilişkilerini çekebilmek için en iyi kaynak olarak gözüküyordu. Projenin ilerlemesiyle gazete haberleri de KAP verilerinin yanına ek kaynak olarak eklendi. Gazete haberlerinin eklenme nedeni BİST dışında bulunan şirketlerin bilgileri ancak gazetelerden bulunabiliyor olmasıydı. Ayrıca gazete haberlerini eklemek, model eğitmek ve test etmek için projeye daha fazla veri sağladı.

Bu proje iki ana aşamadan oluşmaktadır. İlk aşama metin analizi yapıp şirket isimlerini bulmaktır. DDİ yöntemleri içinde Koşullu Rastgele Alanlar (CRF) [1] yöntemi veriyi bölümlemek ve etiketlemek için sıklıkla kullanılır. Çeşitli CRF uyarlamaları arasından projemizde kullanılmak üzere CRF++ [2] seçilmiştir. İkinci aşama ise metindeki ilişkileri çıkarmaktadır. Bunun için, verinin içindeki özelliklerine göre sınıflandırılması yapılabilir. Destek Vektör Makineleri (SVM) [3] sınıflandırma için iyi bilinen tekniklerden biridir. Bu projede, ilişkileri sınıflandırmak için popüler bir SVM uygulaması olan LibSVM [4] kullanılmıştır.

KAP ve web gazete haberlerinden bir miktar veri toplandıktan sonra KAP haberleri modeli eğitmek ve gazete haberleri modeli test etmek için kullanılmaya karar verildi. Dil işleme bölümü için eldeki KAP ve gazete verileri CoNLL 2003 biçimine çevrildi. Bu formatın

seçilme nedeni CRF++'ın model oluşturup test edebilmesi için bu formatta veriye ihtiyaç duymasıydı. Daha sonra bu veriler değişik arama algoritmaları kullanılarak etiketlendi. Ardından CRF++ kullanılıp bu verilerden bir model oluşturuldu ve bu model elde kalan KAP ve gazete haber verileri üzerinde test edildi.

Ancak şirketler arasındaki ilişkilerin çıkarılması için, KAP verisinin resmi dili, gazete verilerinin gündelik dilini sınıflandırmak için uygun değildi. Bu nedenle, daha fazla gazete haberi üzerinde etiketleme yapılarak, LibSVM ile kullanılmak üzere haber verisinin boyutu artırıldı.

Farklı yaklaşımlar ve çeşitli parametrelerle yapılan denemeler sonucunda, test kümeleri üzerinde şirket isimlerini etiketlemede %95 ve ilişki çıkarımında %82 başarıma ulaşıldı.

# Contents

# 1 Introduction and Project Summary

Banks track their customers for better predictions about them. Especially they have to track their company customers because they are the biggest clients for them. So, banks check their current or new company customers' financial status and management decisions. Generally, the biggest operations for companies are mergers, takeovers and split-ups. Hence, banks have to follow the news about company relations. Banks have employees for tracking company relations from news and web sources. These employees have to check every news related to the selected company in order to understand the situation about the company. This process takes long investigation times; therefore, it costs a lot. For decreasing the time and the cost of this job, our project aims to automate this investigation process.

Company Relation Extraction project has been started by Yapı Kredi Technology as a Turkish natural language processing (NLP) graduation project. After some steps it was seen that in addition to the NLP techniques, project needed some machine learning (ML) techniques. Also, a web crawling part was required for getting relevant company news data from web sources.

Company Relation Extraction project consists of 2 main parts. First of them is acquiring the data from the selected web sites. The second one is the implementing natural language processing and machine learning methods to the data that were gotten from the web. Also, data processing section has 2 parts. One of them is finding and tagging company names in the Turkish text. The other part is obtaining the relations among the company names that was found by the machine learning techniques.

For the data acquiring/extraction part different frameworks were researched. At the end Python's web crawling framework that named Scrapy was selected. It is a simple framework for extracting data from web sites. Scrapy spiders were coded for selected data sources of the project.

Project's primal data sources for company relations were Ticari Sicil Gazetesi (TSG), Ticaret Odası Kayıtları (TOK), Kamuyu Aydınlatma Platformu (KAP), Rekabet Kurulu (RK), daily newspapers' financial news pages and social media. Because of the language usage and grammar, the social media was eliminated at the first step. Also, since all newspaper sites do not have the same structure for news articles' parts, each newspapers' site needed a different data crawling program. Hence, only the top 5 newspaper sites have been chosen for extraction. After the research on all data sources, it was found out that TSG and TOK sites contains only general company informations like phone numbers and addresses but do not have any company relation information. Thus, these two sources been eliminated.

On the other hand, KAP and RK have company relation information. RK contains company mergers and takeovers within a sentence long declaration. KAP website contains company relation data as well as other kinds of data like financial reports. Nevertheless, KAP contains companies in the stock market BIST (Borsa Istanbul) so other companies that are not in the stock market informations do not stored in KAP. Yet, companies in the stock market have to inform public so that they have to share information in KAP. These shared and published informations hold them in declaration format. Also, KAP has a declaration search function in it and this search function can find requested information on the website but the page that shows the search result is a dynamic page that written in JavaScript. Hence, before and after search function run page, link doesn’t change so searched declarations could not be extracted via this search. However, KAP data have declaration type so a scrapy spider have been coded that can check every declaration since 2016 and if declaration’s type is merger or takeover, it crawls that page.

KAP stores company relations’ data in the table format (ie. Company 1, Company 2, relation, declaration summary). So, achieving data was relatively easy and almost no language processing needed for extracting relations from KAP. Thus, the daily newspaper websites were added as additional data sources. However, KAP declarations summaries were main data for training a model. Extracting data from different newspaper sites were difficult due to the different structure of each newspaper’s website which needed a different data crawling spider. So following top 5 newspaper sites were selected. These are Hürriyet, Milliyet, HaberTürk, Cumhuriyet and Dünya websites. For crawling newspaper sites Google search was used with KAP’s company list, relation keywords (ie. birleşmesine) and 5 newspaper sites as a parameter of search site. After spiders finished crawling and storing Google search results, there were nearly 4000 news texts, yet most of them were irrelevant so relevant 57 articles were selected.

After web crawling part, some KAP and newspaper article were obtained to train a model for finding company names inside a text. But first these texts have to be tagged by hand for a model. For that, text format was changed to CoNLL format and all of them were tagged. After tagging, CRF++ was used for generating a model. Different models with different features were tested and test results of them are going to be discussed later.

Last mission of the project was to determine relations between companies. For that purpose, bag-of-words approach was used as a feature list and LibSVM was used to classify the news. Early trials were at low accuracy because of limited number of data. So more crawled news was added for more data and total of 155 news were tagged, including non-related ones this time. Later tests on new data were much more promising.

# 2 Comparative Literature Survey

While researching the literature, different methods and different programs for similar projects were checked for understanding the concept and seeing different kind of applications for the same techniques. 5 sample projects [5, 6, 7, 8, 9] were found. The differences between these projects are data sources and methods that they used. Most of them avoided the social media as a source because of the non-formal language, but they used web-based formal data sources. After extracting data from sources, researchers have to know the subject in order to work on selected topic. Yamamoto states out that in order to understand the relations between companies, financial and social knowledge is necessary [8].

Company relations or other relation extraction projects' data sources are generally websites. Some projects use one or two web-based data sources while the others use several sources. F. Naumann's similar project [9] focused on a single newspaper (New York Times) to extract relationship types, such as *ownership_of*, *partnership_of*, *competitor_of*, and *supplier_of* with a semi-supervised strategy. In our project 4 different kinds of company relations were focused. These relation types are mergers, takeovers, split-ups and other relations. Yamamoto states that in order to extract company relations from news, news articles should contain both macro-viewpoint (companies) and micro-viewpoint (relations) [6]. If an article doesn't have any of the viewpoints that mentioned above, then this article was irrelevant. Other relations were used for this kind of irrelevant articles.

Extracting data was the first part of the project. After that, the finding company names and relations were focused on. Finding and tagging informal company names is not a simple task even for people. Raymond said that instead of full company names, abbreviations or pronouns can be used [7]. That makes even more difficult to find the company names in articles or in any text.

All the project that are mentioned above are using natural language processing or machine learning on English language texts with web-based data sources. English NLP parsers are not suitable or adaptable for Turkish language. Some other projects [10, 11] are used Dynamic Conditional Random Fields (DCRF) for language processing on Chinese and Turkish. Some CRF projects were come across however, it used CRF for Named Entity Recognition on Malay language [10]. As mentioned in another similar project [13], finding proper nouns in Turkish is not simple. In the same project Kazkılınç used CRF with several features for finding proper nouns. Similarly, different features for finding company names were used in our project. So, after a little survey a CRF implementation CRF++ have been selected to use in this project.

Bunescu said that information extraction (IE) has 3 subproblems and these are coreference resolution, named entity recognition and relation extraction [14]. 2 of them (NER & RE) were used in this project. After name entity recognition (NER) projects that were searched for tagging company names, research topic was changed to relation extraction (RE). There are several projects on relation extraction [14, 15, 16] and most of them have used kernel methods for that purpose. Support vector machines were the most used technique in relation extraction projects that used kernel methods. In a similar project [15] Bunescu and Mooney used SVM for binary relation extraction but in our project, relations could be between 2 or more companies. However, as Zelenko stated, relations can be between people, organizations and locations [16]. Nevertheless, in this project, only the relations between organizations/companies were focused. Hence, SVM should be used for this purpose; classifying news articles for 4 different company relations. LibSVM, one of the most popular SVM implementations have been chosen for this project.

# 3 Developed Approach and System Model

Projects' initial step and requirement was to find data and, after checking the possible data sources KAP had been selected as a primal source. Accordingly, first spider for KAP was coded. KAP has declarations about any company related news therefore only the company relation related ones had to be selected. First KAP's search function was tried; it was working for finding related declarations. However, result page of the search function was a dynamic JavaScript page, so data could not be extracted from that page by Scrapy spider. Then method for finding related declarations was changed. In KAP every declaration has their own unique declaration number. (ie. https://www.kap.org.tr/tr/Bildirim/625464). These declaration numbers were used for checking every declaration later than 2016. Even though their database contains declarations since 2013, the year 2016 was selected. That is because KAP's declaration format was changed on June 2016 and the data could not be extracted from declarations before 2016 with the same spider. Also, there were at most 30 declarations before that which may be subject to our purpose and even those were containing a very short information. With this basis Scrapy spider was coded. It checked every declaration up to date and crawled the ones relevant to the company relations by saving each instance as web page and storing the extracted declarations in a JsonList (.jl) file. After getting KAP data, newspaper's data had to be extracted from daily news sites. But there are tons of news on the web and selecting related ones was hard. For that reason, Google search engine was used as a news searcher by using company names and relation words as parameters. KAP was used for getting company names list and relation words that consists of 512 company names and 52 keywords with their frequencies for relations (ie. birleşmesine, bölünmüştür). After getting all necessary data a new spider was coded for using these data and Google's some search parameters.

- Google Search Parameters
  - "Company name" in quotes for exact search
  - Keywords for relations with 'OR' between them
  - News sites, 5 different site URLs
  - Language: Turkish, in Turkish texts
  - Sequence: Relevant
  - Source: News

- An example Google search URL

https://www.google.com.tr/search?q="YAPI+KREDİ+FAKTORİNG+A.Ş."+birleşmesine+OR+birleşme+OR+…+site:www.hurriyet.com.tr&lr=lang_tr&tbs=lr:lang_1tr,sbd:0&tbm=nws

- Company name: q="YAPI+KREDİ+FAKTORİNG+A.Ş."
- Keywords: birleşmesine+OR+birleşme+OR+…
- Selected news site: site:www.hurriyet.com.tr
- Language: lr=lang_tr&tbs=lr:lang_1tr

- Relevant order: sbd:0
- In news: tbm=nws

Turkish language parameter, site parameter for top 5 news websites, news parameter and relevant order parameters were used for better search results. After selecting and applying parameters the company name was added as exact search word and relations were added with “OR” between them. After getting search results from Google, result URLs were extracted. Then fuzzy-wuzzy similarity algorithm was used for matching these links with the 5 news spiders. If the link’s similarity is 80% or higher to the top 5 news URLs then spider crawls that page and stores only the news article part inside a JsonList file.

Google search has a character limit so every keyword could not be searched at once. Therefore, keyword list was splitted into 3 parts. For each company, spider has to search 5 different news website, and for every site it has to search for 3 keyword parts in the list. So, the spider searches 15 times for every company. That means for every company 15 requests were sended to Google. But, Google has a request limit for static IP’s. If this spider was run without any delays, Google blocks our static IP’s for hours. So, 15-minute delays were added between each search requests. It took long time but at least Google did not block our IP’s with this method. Also, at some time dynamic IP spider was tried to implement with torrent or any other method. While searching for non-static IP, spider was near the end of the crawling. That is why searching for dynamic IP spider subject was stopped.

Later, nearly 4000 news were held in a file. Nevertheless, most of them are not relevant to the company relations. So, some of them were needed to be eliminated by hand. 57 news were selected for 3 different relations (Merger, takeover and split up). After cleaning and converting data to the CoNLL format, data had to be tagged in order to train a model. Therefore, some similar or exact search algorithms were tried to use for that job. Firstly, Rabin Karp search algorithm and KAP’s company names list were used for tagging the company names automatically. This algorithm uses hashes to search and it is very fast, however Rabin Karp is an exact search algorithm. Therefore, it did not work well with KAP’s name list on news. Because KAP is a formal platform for Borsa Istanbul (BIST) and companies have to use formal language in this site. Yet language that is used in news are not as formal as in KAP. Some company names are not exact and “A.Ş.” can be “A.Ş” or “AŞ” in news. Hence, another algorithm that can work with similarity search was needed. For that reason, bitap algorithm implementation in python was searched and after some research an implementation was found but it was not perfect for project’s process. This bitap implementation was finding a single word but company names consisting of several words. Also, it only returns the found word yet the location of the word was needed for tagging. Therefore, the implementation was changed for project’s purposes. Nevertheless, neither Rabin Karp nor bitap algorithms were perfect for project’s goal. Bitap tagged around 70% of the companies. Later, the rest of the companies were tagged by hand. This was the last part of the collecting and preparing data for the rest of the project.

To find company names automatically in new texts, CRF++ was used. It is a conditional random field program written in C++. It generates the model within seconds, that is why it was good for this project. First plan for CRF++ was to use KAP data for training a model and to use newspaper data for testing that model. So, the data was needed to be converted to CoNLL format for using them with CRF++. Python script was coded for that however it was not enough for CRF++. Data needs tags for training a model therefore both KAP and news data had to be tagged with the company names. First KAP's company list was used for tagging the company names automatically but it was not enough. Hence, rest of them was tagged by hand. Then, data files were ready for CRF++. Data was tested and results will be discussed in the next section.

CRF++ runs on Windows command prompt. Also train and test commands are different, and they need to run separately. Picture 1 and 2 shows the running CRF++ train and test commands respectively.

```
B:\Courses\BLG 492 (Bitirme)\CRF++\CRF++>crf_learn ykt3\template5 ykt2\trainKarışık.txt ykt2\modelKarışık
CRF++: Yet Another CRF Tool Kit
Copyright (C) 2005-2013 Taku Kudo, All rights reserved.

reading training data: 100.. 200.. 300.. 400.. 500.. 600.. 700.. 800.. 900.. 1000..
Done!1.30 s

Number of sentences: 1030
Number of features:  329400
Number of thread(s): 8
Freq:                1
eta:                 0.00010
C:                   1.00000
shrinking size:      20
iter=0 terr=0.89005 serr=1.00000 act=329400 obj=39887.84765 diff=1.00000
iter=1 terr=0.10995 serr=0.60388 act=329400 obj=25678.04742 diff=0.35624
iter=2 terr=0.10995 serr=0.60388 act=329400 obj=20677.74283 diff=0.19473
iter=3 terr=0.11102 serr=0.71262 act=329400 obj=14776.58343 diff=0.28539
```

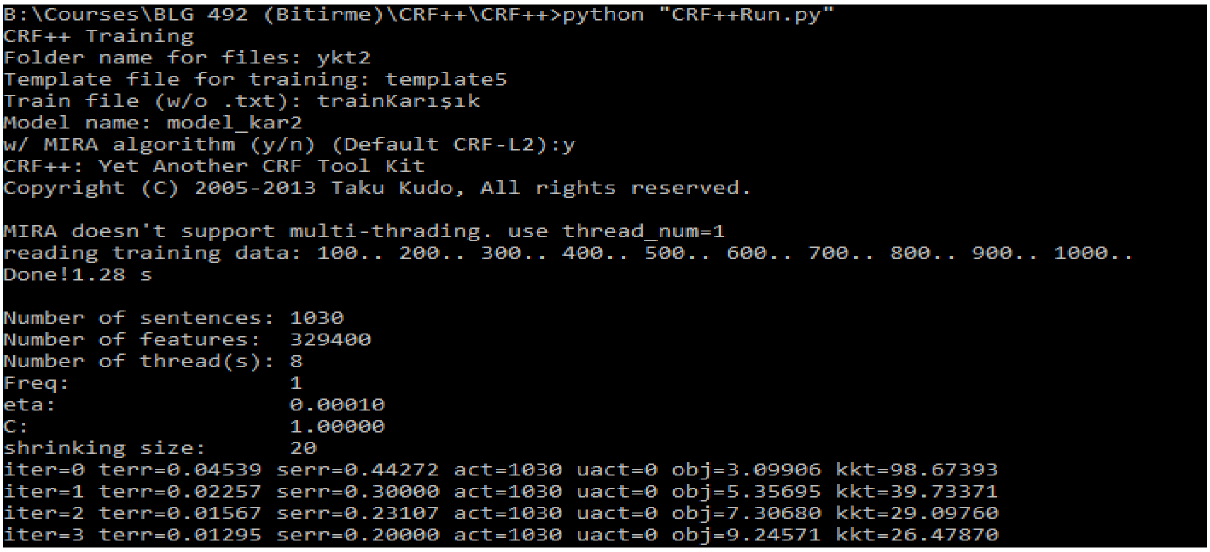
Picture 1: CRF++ training a model

```
iter=142 terr=0.00012 serr=0.00388 act=329400 obj=363.94502 diff=0.00008
iter=143 terr=0.00012 serr=0.00388 act=329400 obj=363.92735 diff=0.00005
iter=144 terr=0.00012 serr=0.00388 act=329400 obj=363.90849 diff=0.00005

Done!5.44 s


B:\Courses\BLG 492 (Bitirme)\CRF++\CRF++>crf_test -m ykt2\modelKarışık ykt2\testKarışık.txt > ykt2\outKarışık.txt

B:\Courses\BLG 492 (Bitirme)\CRF++\CRF++>
```

Picture 2: CRF++ running test

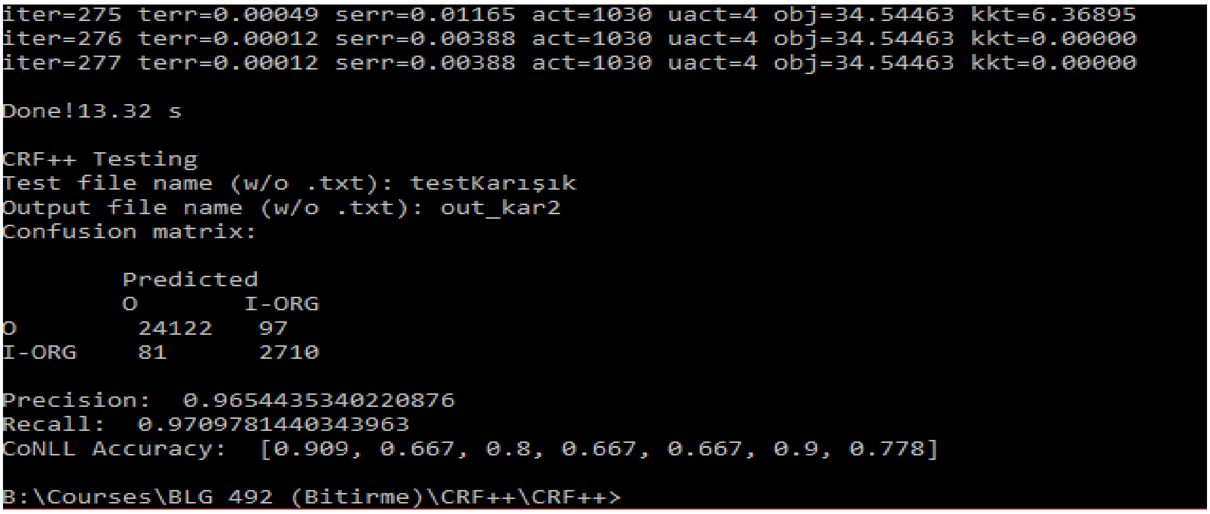
2 Python scripts were coded for running CRF++ train and test. One of them is for calculating accuracy with already tagged data and the other one is for generating new test results for new news articles. Both of them require the folder name of the train and the test files, names of the files and algorithm selection. Then CRF++ starts to generate model for that train file and asks the test and result files' names. After that, first script calculates the precision and recall rates for that result and lastly shows the CoNLL truth rate of the same words. It uses tags (tagged by hand) to calculate accuracy. The second Python script also works like that, the only difference is, it does not show the accuracy part. That's because second script takes simple text format and it does not have any tagged words so it cannot calculate the accuracy of it.

However, it converts simple text to CoNLL format and adds features to it then it will test the model with this file.



```
B:\Courses\BLG 492 (Bitirme)\CRF++\CRF++>python "CRF++Run.py"
CRF++ Training
Folder name for files: ykt2
Template file for training: template5
Train file (w/o .txt): trainKarışık
Model name: model_kar2
w/ MIRA algorithm (y/n) (Default CRF-L2):y
CRF++: Yet Another CRF Tool Kit
Copyright (C) 2005-2013 Taku Kudo, All rights reserved.

MIRA doesn't support multi-thrading. use thread_num=1
reading training data: 100.. 200.. 300.. 400.. 500.. 600.. 700.. 800.. 900.. 1000..
Done!1.28 s

Number of sentences: 1030
Number of features:  329400
Number of thread(s): 8
Freq:                1
eta:                 0.00010
C:                   1.00000
shrinking size:      20
iter=0 terr=0.04539 serr=0.44272 act=1030 uact=0 obj=3.09906 kkt=98.67393
iter=1 terr=0.02257 serr=0.30000 act=1030 uact=0 obj=5.35695 kkt=39.73371
iter=2 terr=0.01567 serr=0.23107 act=1030 uact=0 obj=7.30680 kkt=29.09760
iter=3 terr=0.01295 serr=0.20000 act=1030 uact=0 obj=9.24571 kkt=26.47870
```

Picture 3: Python script for running CRF++ (Train)



```
iter=275 terr=0.00049 serr=0.01165 act=1030 uact=4 obj=34.54463 kkt=6.36895
iter=276 terr=0.00012 serr=0.00388 act=1030 uact=4 obj=34.54463 kkt=0.00000
iter=277 terr=0.00012 serr=0.00388 act=1030 uact=4 obj=34.54463 kkt=0.00000

Done!13.32 s

CRF++ Testing
Test file name (w/o .txt): testKarışık
Output file name (w/o .txt): out_kar2
Confusion matrix:

        Predicted
        O        I-ORG
O         24122   97
I-ORG     81       2710

Precision:  0.9654435340220876
Recall:  0.9709781440343963
CoNLL Accuracy:  [0.909, 0.667, 0.8, 0.667, 0.667, 0.9, 0.778]

B:\Courses\BLG 492 (Bitirme)\CRF++\CRF++>
```

Picture 4: Python script for running CRF++ (Test & Results)

LibSVM can also run from the command line once the required training and testing data files supplied as parameters. And as mentioned before these data files are needed to be in a digitally represented form. For that purpose a python script was written "_trainFileGenerator_4l.py" to convert our existing news data in .jl format containing four different labels. The 1k set is hardcoded into the script and if no external feature set supplied with -w parameter, script uses this set as default. Also, if no output file specified with parameter -o, script write result to "trainingData" file by default. Full usage of our script is as follows:

trainFileGenerator_4l.py -w <words-weightsSet> -i <inputFile> -o <outputFile> -l <letterLength> -s <scope>

```
C:\Windows\System32\cmd.exe

C:\Users\aliosmanatik\Desktop\Bitirme\DATA\newGoogleData>python _trainFileGenerator_4l.py
-i testData_th_4l_sorted.jl -l 5 -s 0

 Warning, words-weightsSetFile not supplied, using hardcoded default set!

 > Word count = 996

 Using 5-5 letter length features!

 > Reduced feature count = 441

 Writing translated results to "trainingData" file...
```

Picture 5: Script running with default values

```
C:\Windows\System32\cmd.exe

C:\Users\aliosmanatik\Desktop\Bitirme\DATA\newGoogleData>python _trainFileGenerator_4l.py
-w kw_w_124_sorted.txt -i trainData_th_4l_sorted.jl -o trainingData_th_4l_l6_s2 -l 6 -s 2

 Reading words-weights set from "kw_w_124_sorted.txt" file...

 > Word count = 124

 Using 4-6 letter length features!

 > Reduced feature count = 15

 Writing translated results to "trainingData_th_4l_l6_s2" file...
```

Picture 6: Script running with supplied parameters

After generating data files for LibSVM, models were easily trained and predicted test data with this model. To train models “svm-train.exe” was used (ie. svm-train.exe trainingData_th_4l_l5_s0 model_th_4l_l5_s0), this program also can make a cross-validation on given data by using -v parameter (ie. svm-train.exe -v 5 allData_th_4l_l5_s0). To make a prediction by using a created model, “svm-predict.exe” was used and it writes the predicted classes for test file to an output file, and simply calculates accuracy.

```
C:\Windows\System32\cmd.exe

C:\Users\aliosmanatik\Desktop\libSVM\YKT_NEW>..\windows\svm-predict.exe testData_th_4l_l4_s0
model_th_4l_l4_s0 output_th_4l_l4_s0
Accuracy = 82.5% (33/40) (classification)
```

Picture 7: LibSVM predicting a test data with a pretrained model

## 3.1 Data Model

JsonList (.jl) was used for storing KAP and news data and KAP’s company names list. CRF++ needs text files (.txt) for both train and test data. LibSVM requires digital representation of data for classification in the format:

<label 1> <feature 1>:<value 1> <feature 2>:<value 2> ...
<label 2> <feature 1>:<value 1> <feature 2>:<value 2> ...
...

KAP declarations stored in JsonList files. These files do not have only the summary part of the declaration, they also have different parameters of the declarations. These parameters are declaration number, company code, name of the company, relation type, second company name, declaration date, meeting date and operation date. The below one is an example declaration in file.

{"bNo": "642236", "kod": "BOSSA", "isim": "BOSSA TİCARET VE SANAYİ İŞLETMELERİ T.A.Ş.", "yontem": "Devralma Şeklinde Birleşme", "birlesen": "Akkardan Sanayi ve Ticaret A.Ş.", "kapTarih": "21.11.2017 17:33:57", "ykkTarih": "09.11.2017", "bwftTarih": "30.06.2017", "aciklama": "Ortaklarımızdan Akkardan Sanayi ve Ticaret A.Ş.'nin, şirketimiz tarafından devralınması suretiyle birleşilmesi hususunda SPK'ya yapılan başvuru kapsamındaki belgeler ekte ortaklarımızın ve diğer ilgililerin bilgisine sunulmuştur."}

News articles stored in JsonList files too. These only have 3 different parameters. These are relation, source of the news and the articles itself. Here is an example.

{"iliski": "devir", "kaynak": "Dunya", "haber": "Sermaye Piyasası Kurulu (SPK), Finans Finansal Kiralama'nın 2 milyar TL … "}

For CRF++ train and test data CoNLL format was used on text files. In that format every word and punctuation have to be on different row. However, in our CoNLL format some punctuation formats were not divided. These are dates (ie. 12.06.2017) and “A.Ş.” formats. The reason for not dividing these words is, easier tagging and training for CRF++. In CRF++’s CoNLL format every word (row) has its own feature column(s), also the last column is the class label for that row. Project's goal was finding the company names, so train data’s class label has only 2 different tags. These are “I-ORG” which means a company name and “O” which means others. For feature columns 4 different features were used. These 4 features are going to be explained in section 3.2. A sample data without any features is shown below.

```
işleterek	O
Bursa	I-ORG
Çimento	I-ORG
Fabrikası	I-ORG
A.Ş.	I-ORG
'	O
ne	O
devir	O
```

If this data used for training, test results will be in the same format as this (without features). In this format CRF++ estimates the labels of the words' and writes them to the second column. But if the train data has feature columns in it, test data should have the same feature columns as train data for correct calculations. Therefore, when new data arrived at the program, it first converts it to the CoNLL format and then adds the same features in it in order to progression.

LibSVM was taking a data in a digitally represented form. The JsonList (.jl) format which used to store labeled news data was easy to convert into the required form. So, a python code was used to transform news data into required format. Transforming KAP data was a bit trickier because there was more than one operation defined for the same type of relation, so another code was written to transform KAP data into the labeled news data format, which was already transformed into LibSVM format. The bag of words approach was used to define the feature list.

For the first approach, the distribution of unique words was found in all the data and selected words presented more than 8 times in the dataset. Some possible relational words that may occur in news were also added but did not show up in the data. After checking the list and cleaning from undesired words, like special name occurrences more than 8 by chance, 1k word list which will be the features were selected. Data file simply consists of occurrence count of the unique feature words in given sample and zero occurrences do not needed to be mentioned. Below is an example of formatted training or test data by using 1k features list.

1 1:2 96:1 148:1 300:1 329:1 458:1 536:1 590:1 639:1 741:1 836:1 841:1 924:2 929:1 994:1
2 1:1 143:1 265:1 300:1 329:1 590:1 763:1 841:1 924:1 963:1 994:1
3 111:1 130:1 308:1 346:1 456:1 510:1 708:1 735:1 738:1 741:1 763:1 796:1 836:1 929:2

For the secondary approach, news data were increased up to 155 including non-relation data samples and also used another feature list consists of words defining relations only. For that purpose, relation keywords list was enchanted and ended up with 124-word keywords list. In the second approach the whole words were not used instead given length of first letters are used and also shorter words within given scope are included. For example, using 6 letters cut with scope of 2 will unite words like "devrine, devrinden, devrindeki" and "devrinin" to a single feature "devrin" but will also include words with 4 and 5 letters like "devir" and "devri". Below is an example of formatted data by using 124 features list with 6 letters cut and scope of 2 which reduces the feature count to 15.

1 4:4 5:1 9:1
2 3:1 5:2 6:3
3 6:1 9:5 10:1 11:1 12:4
4 4:1

# 3.2 Structural Model

## 3.2.1 CRF++ Features

Training a CRF++ model is easy however creating a good model is hard. For that reason, different features and templates for models were tried. As discussed at section 3.1 4 different features were used for train and test data. The definition of those features:

- Feature 1: This feature mainly focuses on the words' letters.

  Tags: L, UL, UU, UPU, UUP, UPUP
  - L: First letter of the word is lower OR it is not word (punctuation). (exp: ", bir)
  - UL: Word starts with capitalized letter and the other letters are lowercase. (exp: Türk)
  - UU: First and the second letters are uppercase so the whole word. (exp: YATIRIM)
  - UPU: First letter is uppercase then punctuation then uppercase. (exp: A.Ş)
  - UUP: Two uppercase letters then punctuation. (exp: AŞ.)
  - UPUP: Uppercase, punctuation, uppercase and punctuation (exp: A.Ş.)

3 different tags were used for "A.Ş." words because of the news. KAP language is formal and KAP's data only contains "A.Ş." format but in the news other 2 formats were used too. If word starts with lowercase CRF understands that it is not a company name because company names are proper nouns, and they have to start with uppercase. The difference between "UL" and "UU" is checking words' whole letters are uppercase or not.

- Feature 2: If the word is city or person name.

  Tags: CITY, NAME, O
  - CITY: If the word is a city name in Turkey. (exp: İstanbul)
  - NAME: If the word is a person name in Turkish. (exp: Ahmet)
  - O: If the word is neither a city nor a person name. (exp: bütün)

In the script that adds these features, it only checks the words if they start with uppercase letters. Also, script checks those names from the lists. 2 different text files used for city names (81) and the person names (12k). Therefore, checking all the names takes a little bit time.

- Feature 3: Checking punctuations and special ones.

  Tags: A, P, PUNC
  - A: If the row's element is apostrophe. (exp: ')
  - P: If the row's element is parenthesis. (exp: "(", ")" )
  - PUNC: If the row's element is punctuation but not apostrophe or parenthesis. (exp: ",", ".")

Checking apostrophe character is important because if Turkish proper nouns have any suffix, suffix part is divided by apostrophe (ie. A.Ş.’nin). So, if the current row’s element is apostrophe that specifies that previous element is proper noun. Company names are proper nouns too so, checking apostrophe character is noteworthy. Parenthesis was added as a special punctuation because in news some companies have their abbreviations after their names (ie. “Borsa İstanbul A.Ş. (BİST)”). Lastly, adding punctuation as a feature is good for eliminating non-company names.

- Feature 4: Checking for digits.

  Tags: DGT, CDGT, NDGT

  - DGT: If the row’s element is completely digit. (exp : 2012)
  - CDGT: If the element contains some digit in it. (exp: 20.05.2015)
  - NDGT: If the element has not any digit in it. (exp: bünyesinde)

This feature is good for finding “O” tags because company names does not contain any digit in it.

Small example of featured CRF++ data is below.

| | | | | | |
|---|---|---|---|---|---|
| SPK | UU | O | NPNC | NDGT | O |
| ayrıca | L | O | NPNC | NDGT | O |
| Nuh | UL | NAME | NPNC | NDGT | I-ORG |
| Çimento | UL | O | NPNC | NDGT | I-ORG |
| Sanayi | UL | O | NPNC | NDGT | I-ORG |
| A.Ş. | UPUP | O | NPNC | NDGT | I-ORG |
| ' | L | O | A | NDGT | O |
| nin | L | O | NPNC | NDGT | O |
| sermayesinin | L | O | NPNC | NDGT | O |
| ve | L | O | NPNC | NDGT | O |

In order to train a model with or without features CRF++ needs a template file. It uses template file to determine how it is going to be use the features or words itself. While training different models for better results several template files were tried. Some of them used words itself without any features, some of them used words and features together. Example template is at below.

```
U02:%x[-1,1]
U03:%x[0,1]
U04:%x[1,1]
U05:%x[2,1]
```

[x, y] In this format x represents the rows by current word and y represents the columns for the word or features. So [-1,0] represents the previous word of the current word and [0,2]

represents the current words' second feature. There are 2 types of template features in CRF++. One of them is unigram the other one is bigram. Bigram features were not used in templates therefore whole template file consists of all unigram feature templates. Coordinates previous part is the name of each feature templates. They have to be different from each other in order to separate them. So, they were numbered sequentially. It has to start with "U" because they are unigram feature templates. Other than these feature templates, there are comparison templates. An example is below.

U33:%x[-1,2]/%x[0,2]/%x[1,2]

In this template, CRF++ counts all possible scenarios within this template and stores in its model file.

### 3.2.2 LibSVM Features

As mentioned in section 3.1, 1k feature list was used as in the first approach. But there were 320 from KAP and 57 from Google news, in total 377 labeled samples. This caused too much sparsity on the features which means there were too many features with zero values for each sample. Data also had three different labels which are "birleşme", "bölünme" and "devir" that forces any test data to be classified as one of these. Actually, the test data was hand chosen and each sample is in one of these classes was known but in real life this would be a problem. Sparsity of features and fewness of samples made it difficult to separate classes. Below is the first few 15-word samples of relation type "devir" and "bölünme":

2 1:5 16:1 24:1 40:1 48:1 49:1 58:2 123:1 132:1 134:1 165:1 174:1 180:1 182:1 191:1 …
2 1:2 2:1 12:1 23:1 24:1 41:1 57:3 58:1 61:1 63:1 77:3 79:6 84:1 87:1 90:1 111:1 116:5 ...
3 40:1 49:1 58:1 70:2 116:1 127:1 180:1 183:2 252:1 258:1 286:2 307:1 345:1 380:1 ...
3 9:1 47:1 49:1 58:1 61:1 70:5 127:3 132:1 158:1 163:1 170:1 256:2 273:1 278:5 287:1 ...

As you can see most of the features occur only once and similarity in different classes is more than their differences and this causes to classify all of them as same. The accuracy of these models was around 25%. This might be because of using KAP data for training and testing on Google data. So, a mixed training and test sets were created and this raised our accuracy up to 42%.

After these low scores data had been rechecked and too much similar declarations observed in KAP data. After cleaning there were 240 of them left. Also, Google data have been cleared a little and ended up with 52 news. As the next step the keyword list used to select features. on this tests KAP and news models ended up 29% average and mixed model raised up to 44% accuracy.

To overcome previous sparsity issue getting closer to the root forms of words was tried and to achieve this, a new script written that cuts letters from desired length (l) and include shorter words with in scope (s) to create a reduced set of features. With this approach a final 84% accuracy have been achieved.

Although the result was satisfying, different language of KAP data and Google data was making the thing difficult also there was no samples of no-relation in data set. So, a radical decision was made at this point and we go back to labeling more data from crawled Google news and KAP data left.

After all these proses there was a new data set consisting of total of 155 labeled news data but including non-relational samples this time. However, the new data set was somehow unbalanced - 27 “birleşme”, 4 “bölünme”, 102 “devir”, 22 “ilişkisiz” - but this was all, and there was no time for crawling more sources.

With letter cut technique 70% cross validation score and 57% trained model accuracy was reached over 1k feature set, this method was much more successful.

Dataset was splitted as 40 test and 115 training samples. But against 423 features the data would still be sparse. So, as a next step, instead of 1k set, unique keywords list was used, and augmented the previous list to end up with 124 unique keywords defining our relations.

As defined in section 3.1, the latest method uses letter cut and its scope approach. In the first trial 8 letter cut was used with a scope of 4 that reduces the feature count to 47. Here is a few samples from each class type.

1 11:1 13:3 20:1 36:1
1 9:1 10:1 11:3
2 4:3
2 4:1 17:2 25:3
3 24:1 36:1 43:1 47:3
3 25:1 27:1 36:4 44:1 47:3
4
4 11:1
4

It was easier to classify this data because there were fewer features and more samples accordingly. Although, class 4 “ilişkisiz” may be confused with class 1 “birleşme” most of the time type 1 relations include some more extra features. As a result, the accuracy rose to 65% which was promising.

After trying out different combinations, it was found out that a smaller letter cuts with a small or no scope was producing the most effective feature sets. One of the best trials was a 5-letter cut with a scope of 0 that reduces the feature count to 7. Here is the feature list and a data sample.

[“BÖLÜN”, “BİRLE”, “DEVRA”, “DEVRE”, “DEVRO”, “DEVRİ”, “DEVİR”]

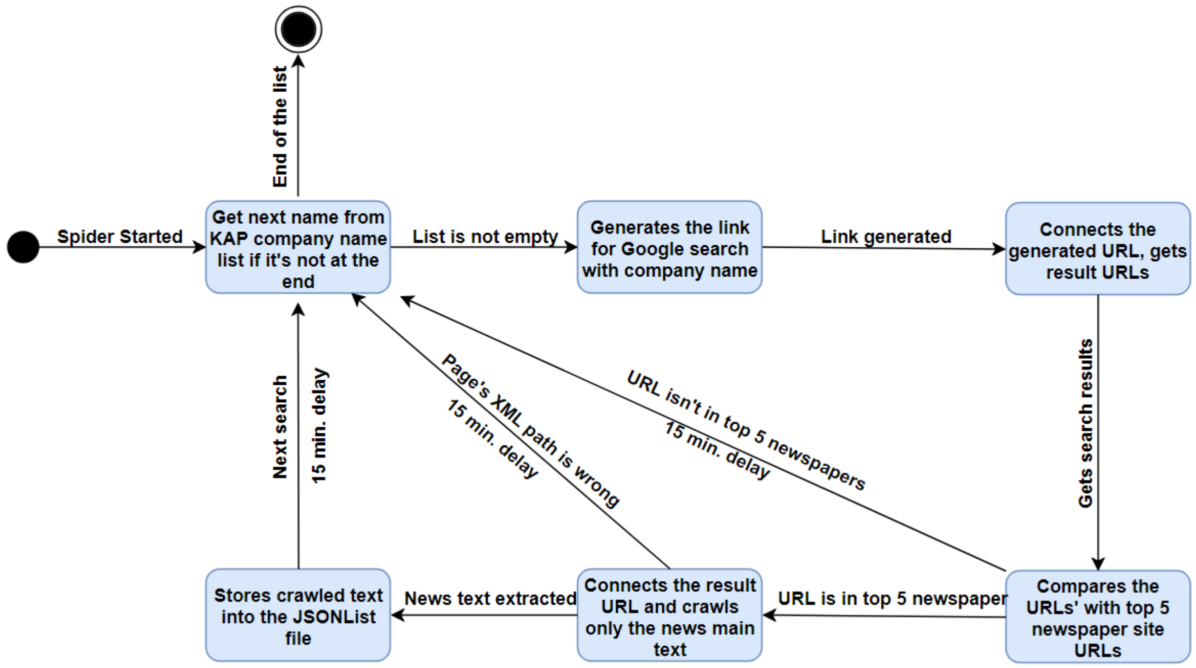
1 2:4 3:1 6:1
1 2:3
1 2:10 4:1
2 1:1 4:1
2 1:3
3 6:3 7:1
3 4:8 6:1 7:2
3 3:1 7:1
4
4 2:1
4

With this final model 82,5% accuracy achieved with the projects’ limited data size.

## 3.3 Dynamic Model

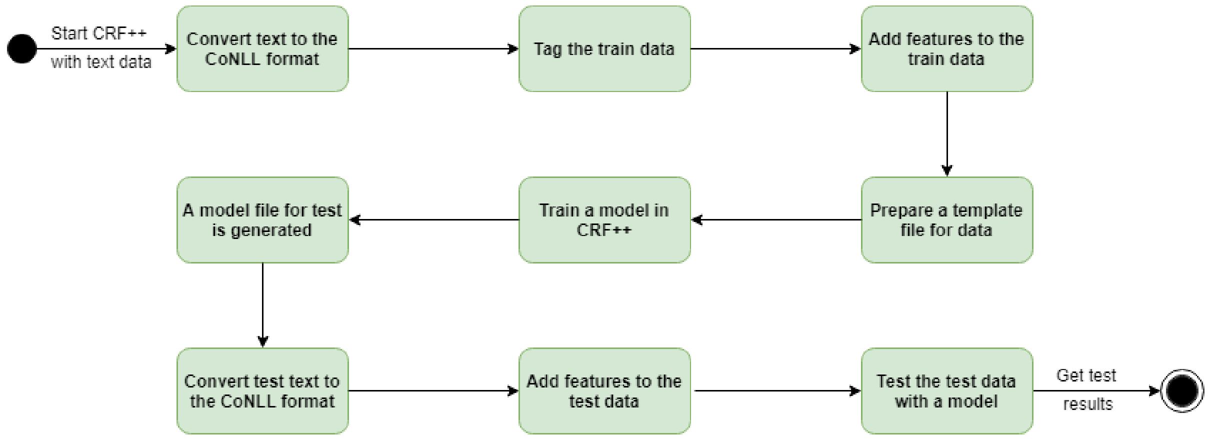
There are 4 different state charts below. These show how project parts work individually and project works as a whole.

### 3.3.1 Google Spider State Chart



## 3.3.2 CRF++ State Chart

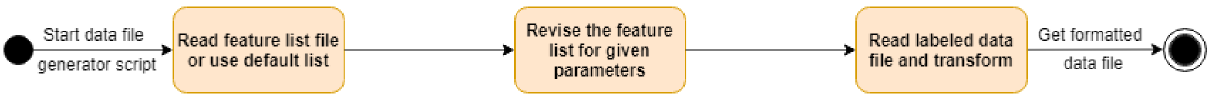
### Training & Testing



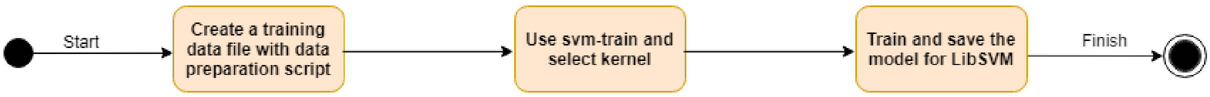

### Testing


## 3.3.3 LibSVM State Chart

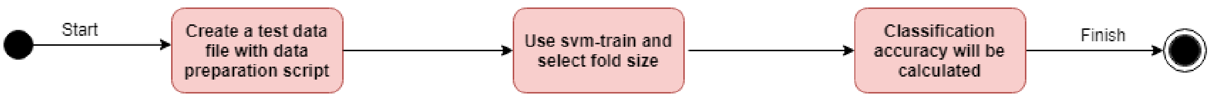
### Data Preparation



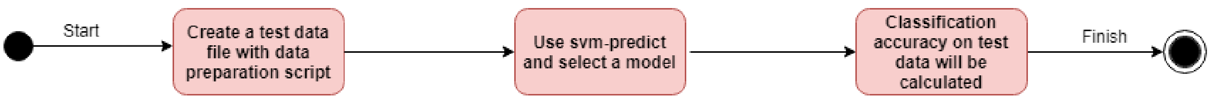

### Training



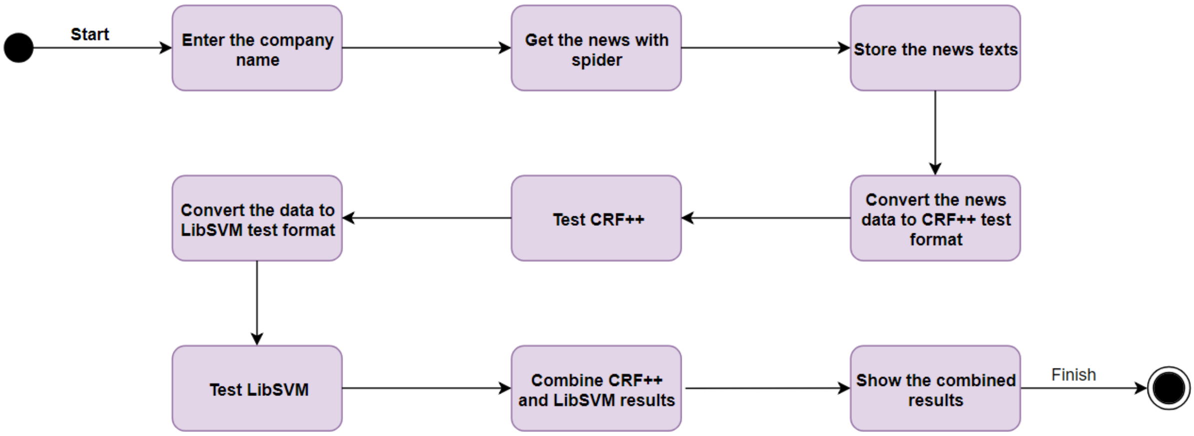

### Testing with k-fold cross validation




### Testing with a model



## 3.3.4 All Project State Chart



# 4 Experimentation Environment and Experiment Design

Python or Anaconda Minimum System Requirements
• Processors: Intel Atom® processor or Intel® CoreTM i3 processor
• Operating systems: Windows Vista or later, macOS, and Linux
• 32- or 64-bit computer.
• For Miniconda — 400 MB disk space.
• For Anaconda — Minimum 3 GB disk space to download and install. Scrapy Tool Requirements
• For Scrapy — Python 2.7 and Python 3.4 or above
• Microsoft Visual C++ Build Tools İTÜ Turkish Language Processing API Non-transferable License Agreement Requirements
• For CRF++ — C++ compiler (gcc 3.0 or higher)
• For LibSVM — Unix or Windows OS

## 4.1 CRF++ Tests

CRF++ was tested with different template files and features. Project's aim is finding company names in newspapers but only 57 news have been extracted. Therefore, additional KAP data were used. In some models and templates, a model was trained with KAP data and was tested on news. Because of the non-formal language in news, recall rate of these models are not very good. Therefore, in some cases news and KAP data were shuffled and then that mixed data model was tested on the rest of the data. Rates were better with that method. At some point only, news data was tested. News data was divided into 2 parts. CRF++ was trained with bigger part and then tested with the smaller part. Also, CRF++ has 3 different algorithms in it. For better results, data was tested with 2 different algorithms and these algorithms are called CRF-L2 and MIRA. MIRA is a different method than the CRF. Also, the difference between these algorithms in CRF++ is MIRA uses more iterations than the CRF-L2, and MIRA cannot work with multi threats so it takes more time than the other. At graphs if the name of the column does not write "/MIRA" that means it is the default algorithm of the CRF++ CRF-L2. There are 4 different graphs below and each of them represent different concepts of the data or template/features. All graphs show 3 accuracy measurements for CRF++ and machine learning techniques. These percentages are accuracy, percentage and recall.

- Confusion matrix:

| | ***Predicted* O** | ***Predicted* I-ORG** |
|---|---|---|
| ***Actual* O** | O_O **(TN)** | O_I-ORG **(FP)** |
| ***Actual* I-ORG** | I-ORG_O **(FN)** | I-ORG_I-ORG **(TP)** |

- Accuracy: (I-ORG_I-ORG + O-O) / (I-ORG_I-ORG + O-O + O_I-ORG + I-ORG_O)
- Precision: I-ORG_I-ORG / (I-ORG_I-ORG + O_I-ORG)
- Recall: I-ORG_I-ORG / (I-ORG_I-ORG + I-ORG_O)

● **Graph 1:** Different Train and Test Data

- Aim of this graph is how train and test data change the rates of the result.
- First 2 columns' (green/yellow/orange) train data is KAP declarations and the test data of these are news. Green ones use CRF-L2 algorithm while yellow/orange ones use MIRA algorithm.
- Last 2 columns' (blue/purple) train and test data are mixed. They consist of 70% of KAP and 30% news on both train and test data. Same with the first 2 columns, blue columns' algorithm is CRF-L2 and purple ones' algorithm is MIRA.

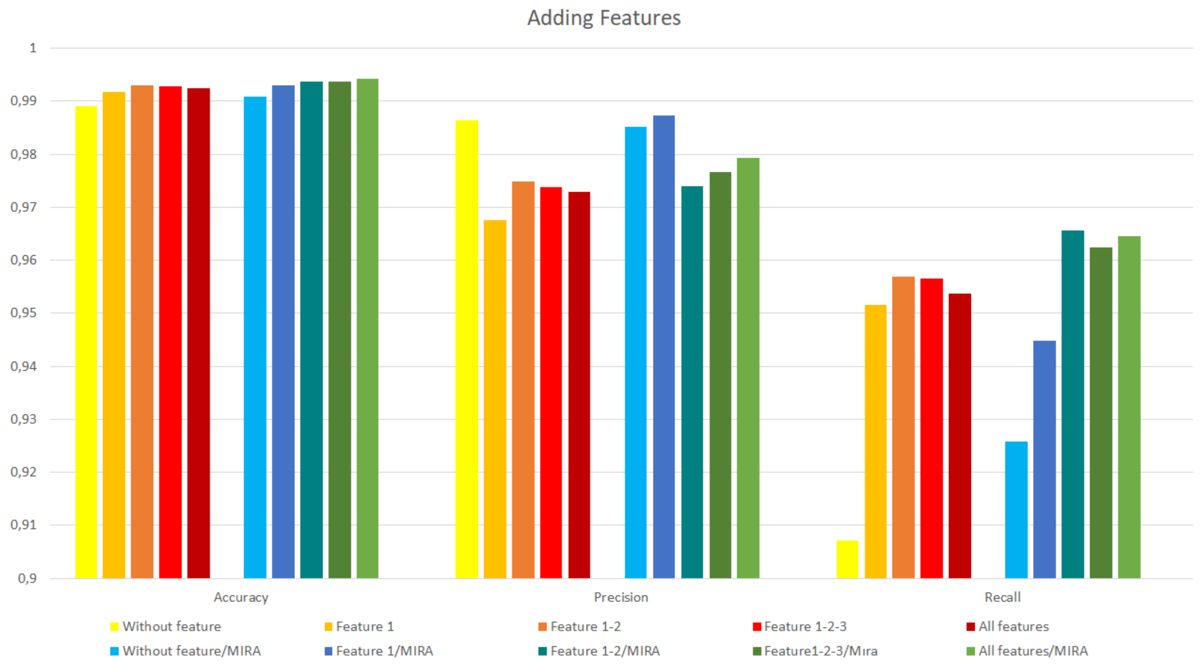

**Graph 1:** Different Train & Test Data

● **Results of the Graph 1 test:**

- This test shows that mixing KAP and news for train and test data increases the all rates. However, recall rate is increased by nearly 200%.
- Adding features to both of the data increases accuracy and recall but decreases the precision rate.
- Mixed data models outperform the KAP data models in every percentage.
- MIRA algorithm slightly better on accuracy and precision, however it is a bit better on recall rates. MIRA rates increase with features.

- **Graph 2:** Adding Features One by One

- This test shows the percentage increase or decrease with the combination of the features.
- Train and test data: Mixed. (70% KAP, 30% news)
- Yellow-red columns shows the different features with CRF-L2 algorithm.
- Blue-green columns demonstrates the MIRA algorithm test results.




**Graph 2:** Adding Features

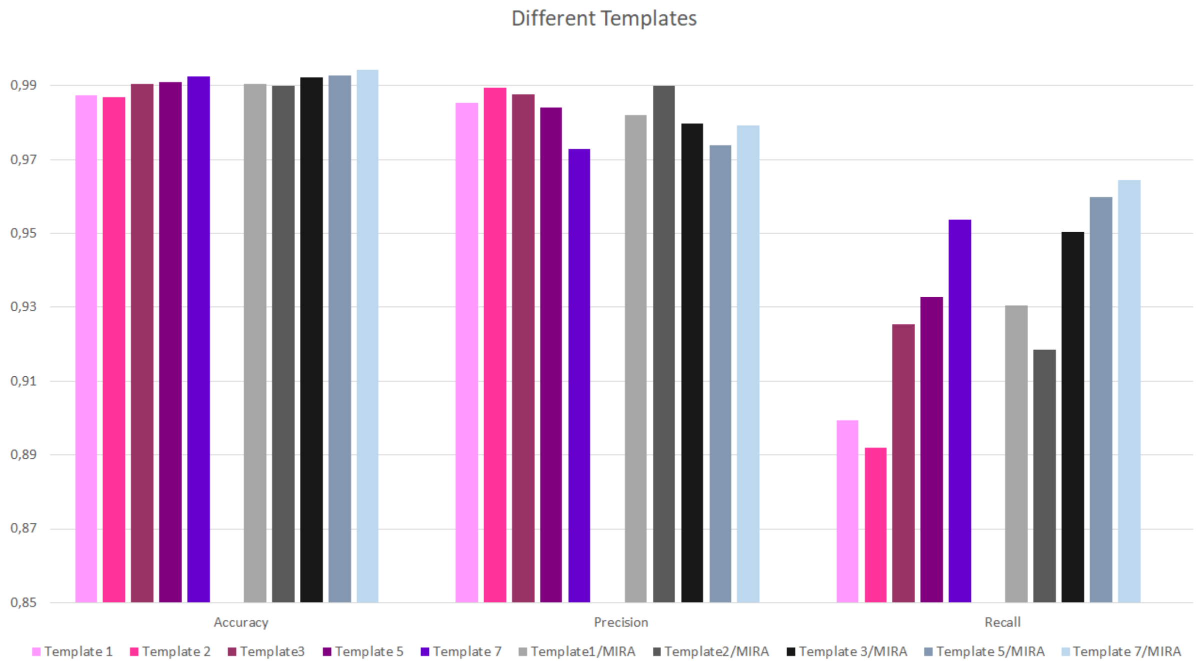
- **Results of the Graph 2 test:**

- This test shows that adding features does not increase the rates all the time.
- Adding features slightly increases the accuracy. However, in some cases it decreases a little on CRF-L2 algorithm.
- Precision does not change regularly with different combination of the features. But it increases with MIRA algorithm, however it decreases with feature 2. CRF-L2 algorithm's precision generally decreases with each feature. Each feature has different effect on number of false positive (FP).
- Recall rates increases with feature 1 and feature 2 but feature 3 and 4 decreases the rate a little. That means feature 1 and 2 decreases the number of false negative (FN) however feature 3 and 4 does not affect the FP number that much.
- MIRA algorithm works better with features by CRF-L2. In most cases MIRA percentages are higher than the CRF-L2 ones.

- **Graph 3:** Different Templates

- This test demonstrates the CRF++ template files affect.
- Train and test data: Mixed. (70% KAP, 30% news)

- Pink/purple columns shows the CRF-L2 algorithm's results.
- Greyish tones indicate MIRA algorithm.
- There are some sample templates in the graph. There is no need to show all templates because some of them very similar to each other (ie template 3 - template 4). However, names of the templates in the graph did not change. So, graph have template 1-2-3-5-7.
  - *Template 1:* Without checking the features, just looks the 3 previous words, current word and 3 next words.
- *Template 2:* Same as template 1 but this template also has comparison in it (ie. U08:%x[-1,0]/%x[0,0]/%x[1,0] ). It compares the previous and next words with current one.
- *Template 3:* Similar to the template 3. The only difference is it also checks the feature 1 and compares these features between them as in template 2.
- *Template 5:* It controls all 4 features and their 3 previous and 3 next ones. Template 5 also compares these features by their property. For example, it only checks punctuations' previous feature not the 3 previous ones.
- *Template 7:* It controls all 4 features and their 3 previous and 3 next ones. However, it does not have any comparison in it.




**Graph 3:** Different Template Files

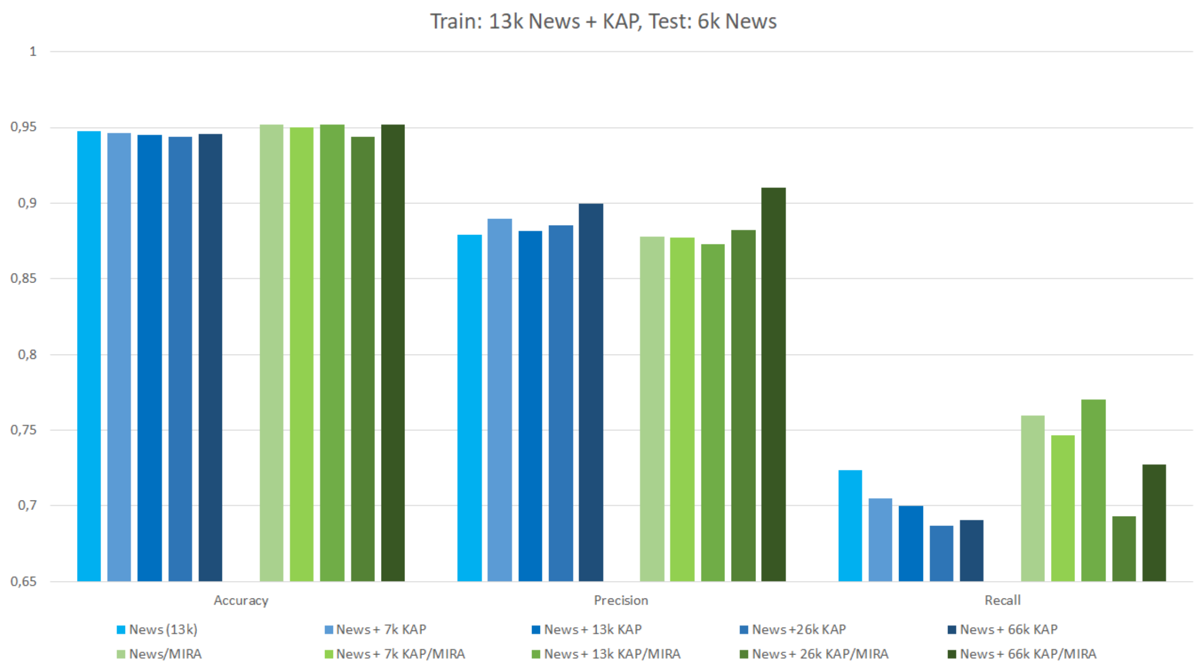
- **Results of the Graph 3 test:**

- This test shows that as the project goes on better template files have been created for CRF++.
- Template 1 → template 2: Precision increases with words' comparison however recall and accuracy decreases with that method.

- Template 2 → template 3: Adding one feature increases the accuracy and recall percentages but decreases the precision one. That means feature 1 in that template increased the number of false positives (FP).
- Template 3 → template 5: Adding other features acts like the previous template change. It increases the accuracy and recall and increases the precision.
- Template 5 → template 7: Getting rid of the feature and word comparison increases the all rates expect the CRF-L2 precision. That shows that comparison is not necessary for better results.
- As in the other 2 graphs, MIRA algorithm has barely better percentages than the CRF-L2 algorithm.

● **Graph 4:** Adding KAP data to news data for training

- There were 57 news and nearly 300 KAP declarations. Same data in ConLL format had 20.000 lines of news and 66.000 lines of KAP declarations. In this graph 7.000 lines of the news were selected in CoNLL format, and they were seperated it for only testing. The remaining part (nearly 13.000) was the first train data. And then some KAP data was added on top of that for seeing if it increases the rates or not. First half of the news amount of KAP was added in train data, then the number of KAP lines was increased to exact number of news lines. Finally, twice the news data from KAP were added and then the whole KAP was added for final test.
- Blue columns show the CRF-L2 algorithms with changed data.
- Green ones show the MIRA algorithm results similar to blue ones.




**Graph 4:** Adding KAP to the news data for training

- **Results of the Graph 4 test:**

- This test aims to find if KAP data is beneficial on news. However, train data without KAP is probably insufficient for test. So, adding KAP to the train data also increases the size of the train data and that is an advantage too.
- Adding KAP data does not change the accuracy for both CRF-L2 and MIRA algorithms. With each iteration it just shifted a bit.
- Precision acts like accuracy, does not change too much except adding whole KAP data. With all KAP data precision increases higher. That probably means, if you have more data to add, it will increase more and more.
- CRF-L2 recall rates drops with each KAP data. However, MIRA rates change differently with each addition. That means in CRF-L2 algorithm, KAP data increases the number of false negative (FN).

## 4.2 LibSVM Tests

Various tests have been made to get a high prediction accuracy for finding the relation type from the texts of interest. The data in JsonList format have been reformatted to make training and testing data which is used with LibSVM to train models and making predictions.

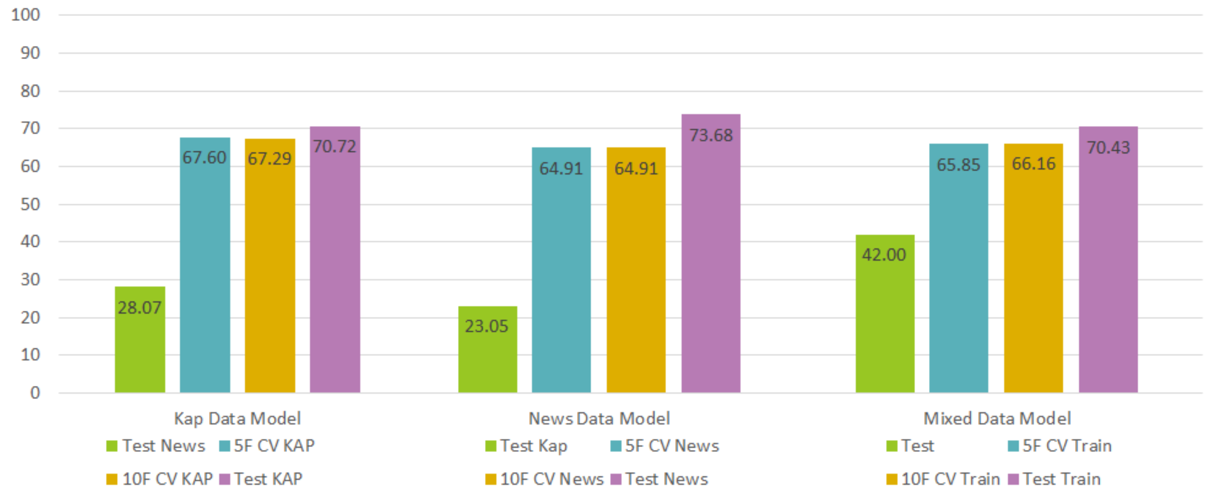
### 4.2.1 First tests

Three different models trained; “KAP data model”, “News data model” and “Mixed data model”. For KAP models test set is news data and for news models tests set was KAP data. For mixed models, KAP and news data combined and ⅓ ratio tried to be preserved between test/training sets.  Following test have been made for each model.

- Testing of test data with trained model
- 5-fold cross validation of training set
- 10-fold cross validation of training set
- Testing of training data with trained model

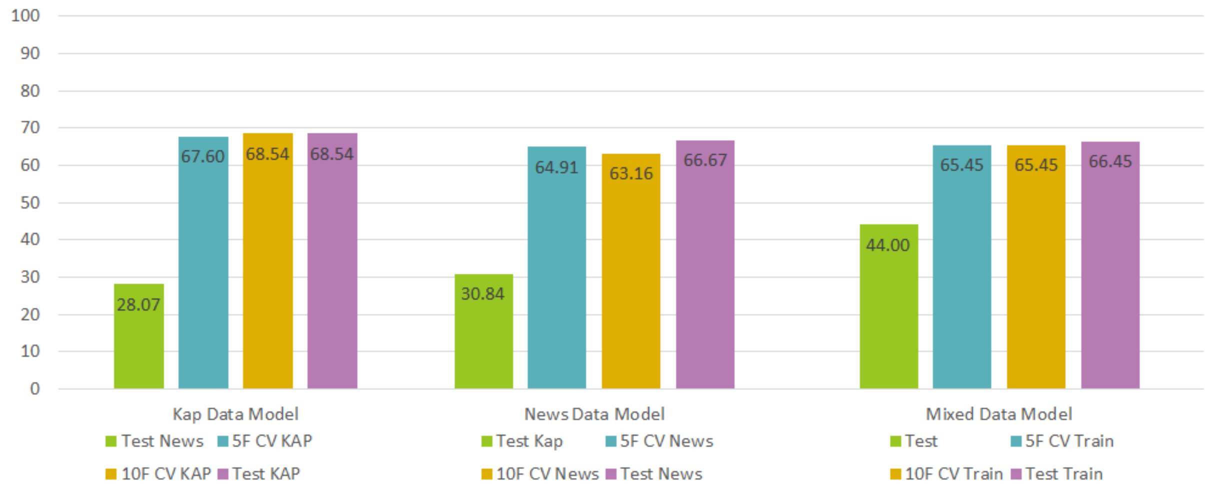
Cross validation and self-testing of trained data with created model have been made to see how much valuable information actually exist on training set and to get a clue on the performance of model.

First tests were on KAP and news data prepared by using “1k feature set” (Graph-5). Trained model tests against test data were around 25% and all predictions were falling into same class (ie. all predicted as “merger”) for KAP and news models. But mixed model did a better job by actually splitting data into different classes.




**Graph 5:** 1k features set results on KAP and news data

Second tests were on KAP and news data prepared by using "Keywords feature set" (Graph 6). Trained model tests against test data were around 29% and again all predictions were falling into same class for KAP model. News model was somehow predicted a few other classes. On the other hand, mixed model did a good job again.

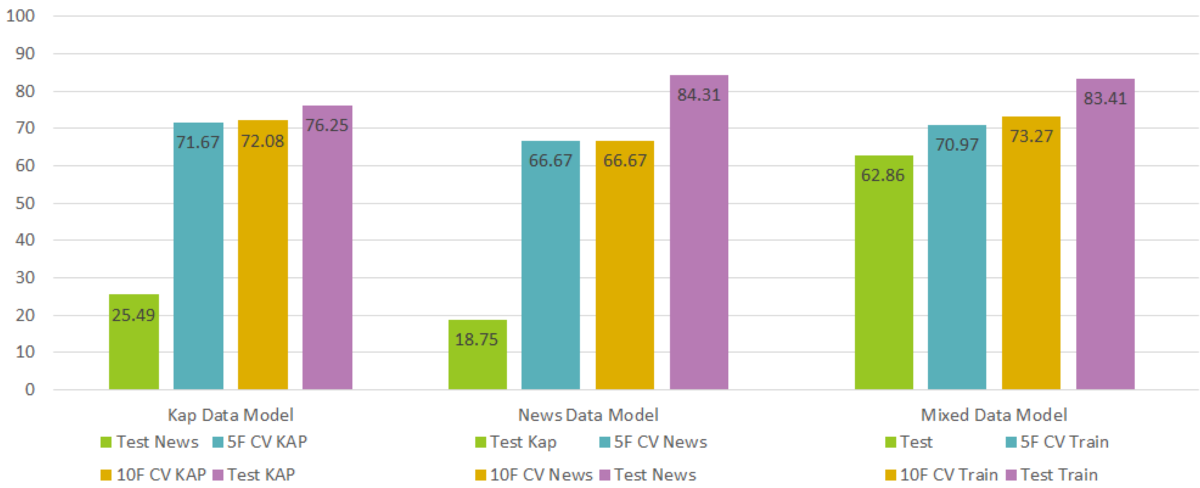



**Graph 6:** Keyword features set results on KAP and news data

So, mixing the data was a good idea. Actually, this was not a surprise, languages of KAP data and news data were different and mixing them made some balance.

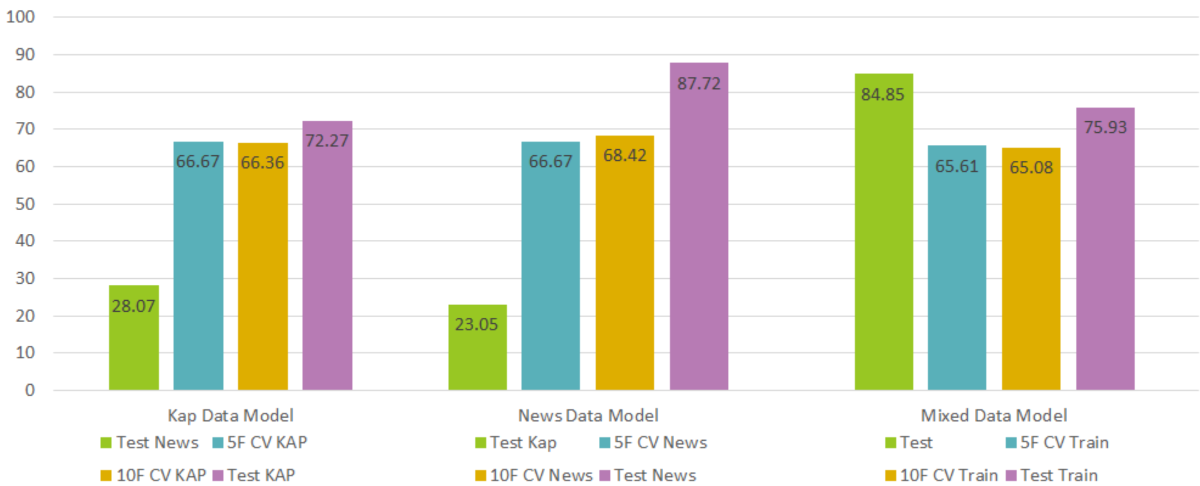
On the other hand, using large feature set might be causing sparsity on the low size data. To eliminate this, letter cut approach have been applied to shrink feature set size.

Models made with 4-letter cut and scope 1 (means 3 letter long words are also included) feature set did not make any good for KAP and news models. But mixed data model was boosted over 60% accuracy, which was promising (Graph 7).




**Graph 7:** 4-Letter cut with 1-Scope features set results on KAP and news data

As next step only a 4-letter cut without a scope tried. As expected, there was no gain on KAP and news models. But this slight difference took the accuracy up to 85% on mixed data even higher than the cross-validation scores (Graph 8).

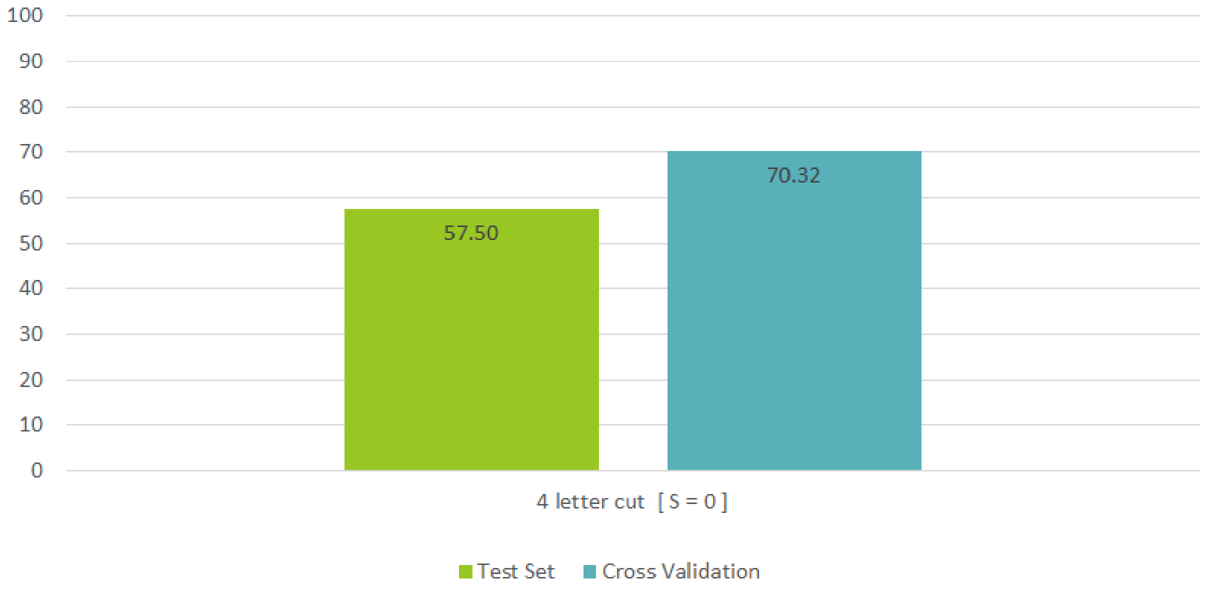



**Graph 8:** 4-Letter cut features set results on KAP and news data

### 4.2.1 Final tests

There were two main problems spotted while making first tests. Mixing data was a good solution for the data in hand, but our goal was to classify news data, so both the training and test data should be from similar sources for a good model. Also, training models with three classes would force a non-relational news to fall into the one of three classes. To overcome these issues more news data have been labeled including non-relational news.

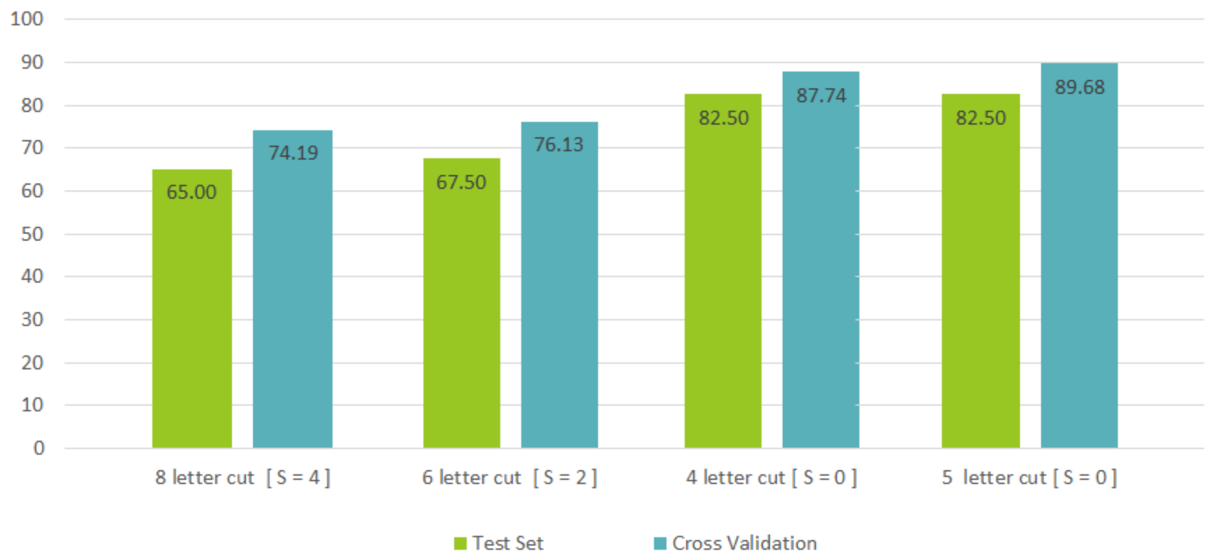
First test with new data made with 1k features set. 57% accuracy was not bad but 70% cross validation score was not promising too much for new data set (Graph 9). Actually, the new data set was less than half of the mixed models used previously, so the sparsity problem should be affecting new data much more now. So, to minimize feature set size an enhanced version of keyword list with 124 words prepared. All test after this point are made by using this keyword set.




**Graph 9:** 4-Letter cut features set results on new data

For finding optimal model various combinations of letter cuts and scope been tried. Graph 10 shows most distinctive ones of them. On the leftmost test with 8-letter cut and scope 4, feature set reduces to 47 words which is almost ⅓ of new data set. Using a 6-letter cut and scope 2, feature set reduces to 15 words but there is no significant gain.

The best result observed at 5 and 4 letters cut sets without using any scope which is 82.5% accuracy for both.




**Graph 10:** Various test results on new data

# 5 Comparative Evaluation and Discussion

In this project different methods and techniques were tried for every step. Some of them worked well and improved the process but some of them did not improve so much. In interim report it was mentioned that ITU Turkish NLP API would be used for language processing parts. But this tool has not been used for that. Instead, CRF++ and LibSVM were used for different purposes. These 2 tools worked fine for the project but maybe using ITU-NLP tool can change the efficiency of the project. In similar projects [6, 7] researchers use dynamic conditional random fields (DCRF) instead of CRF. This method may be tried later in future works for accuracy and performance comparison.

The other topics for future are spiders and taggers. There could be a better spider with dynamic IP for Google searches. Also, for tagging part, it could be a better search algorithm or even mixed of algorithms. On the other hand, due to the time congestion spiders and taggers could not be improved. Finally, our goals for that project in internal report are listed below.

*Internal report goals:*

● Backend services data extraction time should be measurable with hours but not in days (Ex: 2 hours or 20 hours but not much as 2 days. Criterion is intuitive.)
● NLP processing time should be measurable with hours (Ex: 1 hours or 3 hours but not much as 1 day. Criterion is intuitive.)
● Model creation time should be measurable with hours (Ex: 3 hours or 30 hours but not much as 3 days. Criterion is intuitive.)
● Frontend service should create visual graphs within minutes (Ex: 1 minute or 10 minutes but not much as 1 hours. Criterion is intuitive.)
● 10-fold cross validation on training dataset should provide 0.9 precision at least
● Pipeline should work at least 80% accuracy over dataset.

*At the end of the project:*

● Data extraction spiders are not perfect because of the IP blocks yet it will not take 2 days or higher time.
● CRF++ model creation is very fast. It took maximum 10 seconds so it holds the goal. Also, training a LibSVM model is fast but predicting the classes of given data depends on feature size and news size. For our data set it usually took under a second.
● Researching machine learning techniques for finding names and relations took long time. Later on, improving the accuracies of these methods come to issue, therefore no frontend with visual graphics has been developed.
● CRF++ accuracy and others rates are higher than 90% in general but LibSVM accuracy is slightly above 80% for a few chosen model.
● Total average accuracy for best tagging and labeling is 89%.

# 6 Conclusion and Future Work

Completing this project was hard for us. Finding clean and high-quality data sources took long investigation times. Several financial websites and daily newspaper web pages were researched. After selecting KAP as a main data source, reaching the relevant declarations was investigated. KAP's search function wasn't useful for the project because of the website's dynamic structure. Solution for that problem was checking every declaration and then crawling the relevant ones. After completing KAP process, the extracting news articles was started. Company relations news had been searched hence Google was used for that purpose. Every company that are in KAP were searched on Google with relation keywords. However, Google search needed to be used 15 times for every company. 15 searches were needed because of the character limitation of Google (x3) also for top 5 news sites (x5). Due to Google's IP blocks for consecutive search, 15 minutes delays between each search were added. So extracting news took long times with 15 searches and 15 minutes delays. After extracting the news data, nearly 4.000 articles were stored. However, some of them were the exact same ones, some of them were very similar and some of them were irrelevant. Data was needed to cleared and it had to done by hand. Clearing the data took long time but relevant 57 news were selected. Then both KAP and news data were converted to CoNLL format. In order to train a CRF++ model company names had to be tagged. First different search algorithms like Rabin Karp and bitap were tried for that job. After couple of trials a python bitap code was implemented for tagging company names. But it was not perfect and it missed some company names. Hence, the rest of the companies had to be tagged by hand. This process took longer times because of nearly 90.000 lines of data. This process was finished and the training a model process was started. In this step different templates and features were used for increasing the accuracy of model. In the final model 4 features and a template file without any comparison were used. After increasing the accuracy of the CRF relation extraction part was started. LibSVM was used for classifying news data and bag of words approach was used for feature set and the most frequent top 1000 unique words from all over the data were selected. With this 1k set models were created by training with KAP data but test results on Google data was very low. This was mainly caused by sparsity effect of the large feature set and fewness of our data samples. Also, the language of KAP was much formal than Google. So KAP data was left and news data size was increased by labeling out more Google news data and increased the class size to 4 by adding non-relational news. Results were promising, 57,5% accuracy was reached even on first try. But 1k set was still large for our data, so decided to go into the root forms of words by taking first few letters and cutting out the rest but including the whole words shorter than this cut within a scope. This method took us to almost 70% percents. As a next step keyword list was used and it consists of words only determining a relation and 4-5 letter cut features from this set gave us the best result of 82.5% accuracy.

For future works, Scrapy spider can improve and dynamic IP spider will work better with IP blocks. Also, other newspaper websites can be implemented for more data source.

Only CRF++ was tested for company names tagging, different CRF models and methods can be test for better results. CRF++ works fast but for bigger data other methods may work faster.

For relation extraction, even a regex search might be work by using a good feature set for a low size data like ours. For a moderate data one may want to use word-vectors approach to extract relation words. But for large data sizes, which you have enough data for training a good model, SVM classification will work better even with a large feature set.

# 7 References

[1] J. Lafferty, A. McCallum, and F.C.N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", . June 2001.

[2] T. Kudo, CRF++: Yet Another CRF toolkit. [Online] Available at: https://taku910.github.io/crfpp/

[3] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their applications*, *13*(4), 18-28.

[4] C.-C. Chang and C.-J. Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011

[5] Y. Jin, Y. Matsuo, and M. Ishizuka, "Extracting Social Networks Among Various Entities on the Web, The Semantic Web", In Proc. ESWC 2007, pp. 251-266, 2007.

[6] T. Lappas, G. Valkanas, and D. Gunopulos, "Efficient and Domain Invariant Competitor Mining". In Proc. KDD 2012, pp.408-416, 2012.

[7] R. Y. K. Lau and W. Zhang, "Semi-supervised Statistical Inference for Business Entities Extraction and Business Relations Discovery", In Proc. EOS 2011, 2011.

[8] A. Yamamoto, Y. Miyamura, K. Nakata and M. Okamoto, "Company Relation Extraction from Web News Articles for Analyzing Industry Structure", 2017 IEEE 11th International Conference on Semantic Computing (ICSC), 2017 [Online]. Available: http://ieeexplore.ieee.org/document/7889512/. [Accessed: 15-July- 2018]

[9] F. Naumann, "Business Relationship Extraction", Hpi.de, 2018. [Online]. Available: https://hpi.de/naumann/projects/completed-projects/business-relationship- extraction.html. [Accessed: 01- Feb- 2018]

[10] Z. Hao, R. Cai, Y. Yang, W. Wen and L. Liang, "A Dynamic Conditional Random Field Based Framework for Sentence-level Sentiment Analysis of Chinese Microblog", In Proc. IEEE 11th International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), 21-24 July 2017, Guangzhou, Guangdong, China

[11] R. Ehsani, "Turkish Morphological Disambiguation Using Multiple Conditional Random Fields", M.S. Thesis, Computer and Informatics Fac., Istanbul Technical Uni., Istanbul, Jan. 2013. [Accessed: 13- July- 2018]

[12] M. S. Salleh, S. A. A. Asmai, H. Basiron and S. Ahmad, "A Malay Named Entity Recognition Using Conditional Random Fields", In Proc. Fifth International Conference on Information and Communication Technology (ICoICT), 17-19 May 2017, Melaka, Malaysia

[13] S. Kazkılınç, “Türkçe Metinlerin Etiketlenmesi”, M.S. Thesis, Computer and Informatics Fac., Istanbul Technical Uni., Istanbul, Oct. 2012. [Accessed: 13- July- 2018]

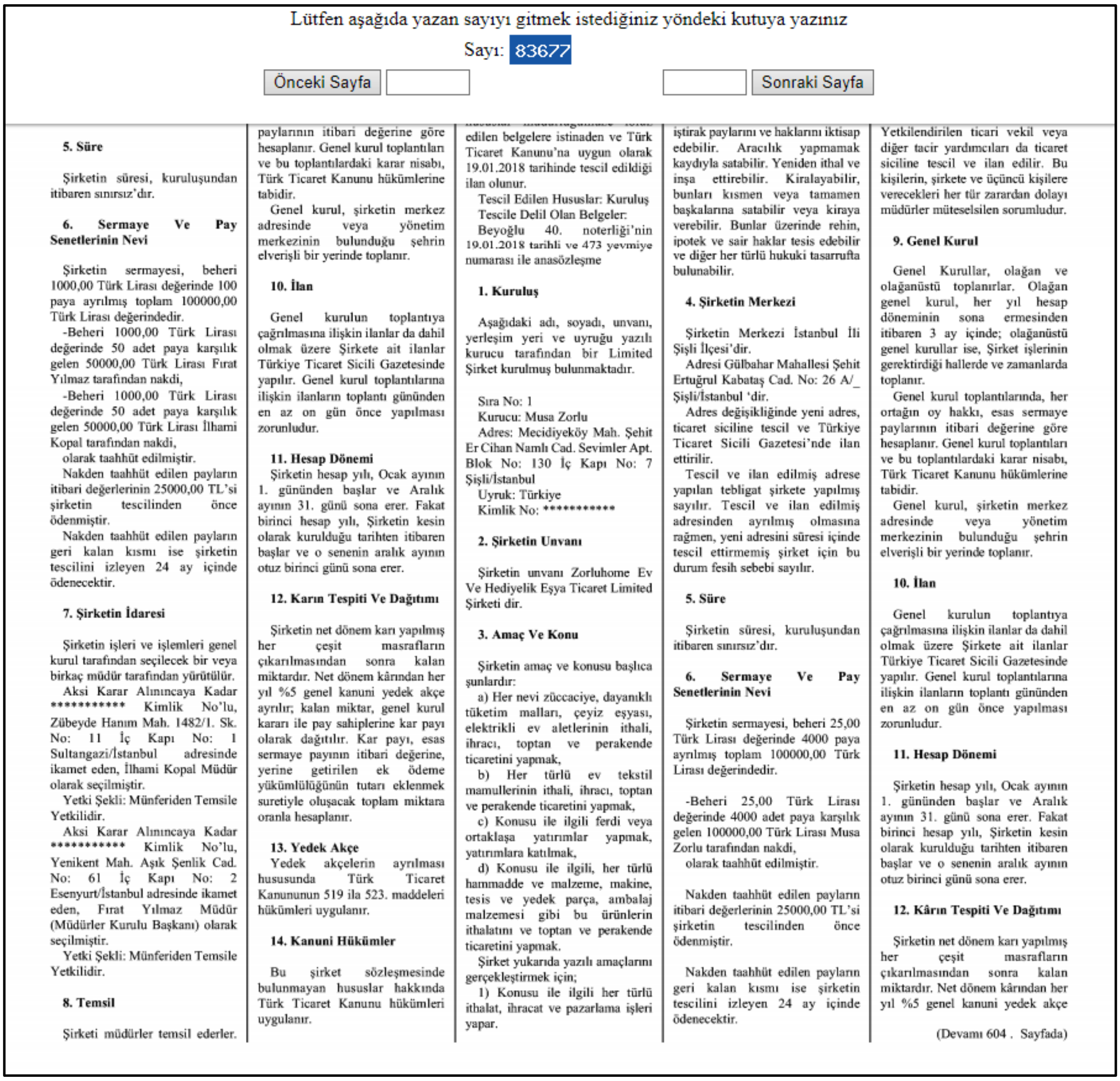
[14] R. C. Bunescu, R. J. Mooney, “A Shortest Path Dependency Kernel for Relation Extraction”, [Online]. Available: https://dl.acm.org/citation.cfm?id=1220666.

[15] R. C. Bunescu, R. J. Mooney, “Learning to Extract Relations from the Web using Minimal Supervision”, In Proc. ACL 2007, June 2007, Prague, Czech Republic, pp. 576-583.

[16] D. Zelenko, C. Aone, A. Richardella, “Kernel Methods for Relation Extraction”, Journal of Machine Learning Research, no 3, pp. 1083-1106, 2003, [Online]. Available: http://www.jmlr.org/papers/v3/zelenko03a.html.

# 8 Appendix

## A - Screenshots of data source candidate web pages



Lütfen aşağıda yazan sayıyı gitmek istediğiniz yöndeki kutuya yazınız

Sayı: 83677

Önceki Sayfa | Sonraki Sayfa

**5. Süre**

Şirketin süresi, kuruluşundan itibaren sınırsız'dır.

**6. Sermaye Ve Pay Senetlerinin Nevi**

Şirketin sermayesi, beheri 1000,00 Türk Lirası değerinde 100 paya ayrılmış toplam 100000,00 Türk Lirası değerindedir.

-Beheri 1000,00 Türk Lirası değerinde 50 adet paya karşılık gelen 50000,00 Türk Lirası Fırat Yılmaz tarafından nakdi,

-Beheri 1000,00 Türk Lirası değerinde 50 adet paya karşılık gelen 50000,00 Türk Lirası İlhami Kopal tarafından nakdi,

olarak taahhüt edilmiştir.

Nakden taahhüt edilen payların itibari değerlerinin 25000,00 TL'si şirketin tescilinden önce ödenmiştir.

Nakden taahhüt edilen payların geri kalan kısmı ise şirketin tescilini izleyen 24 ay içinde ödenecektir.

**7. Şirketin İdaresi**

Şirketin işleri ve işlemleri genel kurul tarafından seçilecek bir veya birkaç müdür tarafından yürütülür.

Aksi Karar Alınıncaya Kadar *********** Kimlik No'lu, Zübeyde Hanım Mah. 1482/1. Sk. No: 11 İç Kapı No: 1 Sultangazi/İstanbul adresinde ikamet eden, İlhami Kopal Müdür olarak seçilmiştir.

Yetki Şekli: Münferiden Temsile Yetkilidir.

Aksi Karar Alınıncaya Kadar *********** Kimlik No'lu, Yenikent Mah. Aşık Şenlik Cad. No: 61 İç Kapı No: 2 Esenyurt/İstanbul adresinde ikamet eden, Fırat Yılmaz Müdür (Müdürler Kurulu Başkanı) olarak seçilmiştir.

Yetki Şekli: Münferiden Temsile Yetkilidir.

**8. Temsil**

Şirketi müdürler temsil ederler.

paylarının itibari değerine göre hesaplanır. Genel kurul toplantıları ve bu toplantılardaki karar nisabı, Türk Ticaret Kanunu hükümlerine tabidir.

Genel kurul, şirketin merkez adresinde veya yönetim merkezinin bulunduğu şehrin elverişli bir yerinde toplanır.

**10. İlan**

Genel kurulun toplantıya çağrılmasına ilişkin ilanlar da dahil olmak üzere Şirkete ait ilanlar Türkiye Ticaret Sicili Gazetesinde yapılır. Genel kurul toplantılarına ilişkin ilanların toplantı gününden en az on gün önce yapılması zorunludur.

**11. Hesap Dönemi**

Şirketin hesap yılı, Ocak ayının 1. gününden başlar ve Aralık ayının 31. günü sona erer. Fakat birinci hesap yılı, Şirketin kesin olarak kurulduğu tarihten itibaren başlar ve o senenin aralık ayının otuz birinci günü sona erer.

**12. Karın Tespiti Ve Dağıtımı**

Şirketin net dönem karı yapılmış her çeşit masrafların çıkarılmasından sonra kalan miktardır. Net dönem kârından her yıl %5 genel kanuni yedek akçe ayrılır; kalan miktar, genel kurul kararı ile pay sahiplerine kar payı olarak dağıtılır. Kar payı, esas sermaye payının itibari değerine, yerine getirilen ek ödeme yükümlülüğünün tutarı eklenmek suretiyle oluşacak toplam miktara oranla hesaplanır.

**13. Yedek Akçe**

Yedek akçelerin ayrılması hususunda Türk Ticaret Kanununun 519 ila 523. maddeleri hükümleri uygulanır.

**14. Kanuni Hükümler**

Bu şirket sözleşmesinde bulunmayan hususlar hakkında Türk Ticaret Kanunu hükümleri uygulanır.

edilen belgelere istinaden ve Türk Ticaret Kanunu'na uygun olarak 19.01.2018 tarihinde tescil edildiği ilan olunur.

Tescil Edilen Hususlar: Kuruluş

Tescile Delil Olan Belgeler:

Beyoğlu 40. noterliği'nin 19.01.2018 tarihli ve 473 yevmiye numarası ile anasözleşme

**1. Kuruluş**

Aşağıdaki adı, soyadı, unvanı, yerleşim yeri ve uyruğu yazılı kurucu tarafından bir Limited Şirket kurulmuş bulunmaktadır.

Sıra No: 1

Kurucu: Musa Zorlu

Adres: Mecidiyeköy Mah. Şehit Er Cihan Namlı Cad. Sevimler Apt. Blok No: 130 İç Kapı No: 7 Şişli/İstanbul

Uyruk: Türkiye

Kimlik No: ***********

**2. Şirketin Unvanı**

Şirketin unvanı Zorluhome Ev Ve Hediyelik Eşya Ticaret Limited Şirketi dir.

**3. Amaç Ve Konu**

Şirketin amaç ve konusu başlıca şunlardır:

a) Her nevi züccaciye, dayanıklı tüketim malları, çeyiz eşyası, elektrikli ev aletlerinin ithali, ihracı, toptan ve perakende ticaretini yapmak,

b) Her türlü ev tekstil mamullerinin ithali, ihracı, toptan ve perakende ticaretini yapmak,

c) Konusu ile ilgili ferdi veya ortaklaşa yatırımlar yapmak, yatırımlara katılmak,

d) Konusu ile ilgili, her türlü hammadde ve malzeme, makine, tesis ve yedek parça, ambalaj malzemesi gibi bu ürünlerin ithalatını ve toptan ve perakende ticaretini yapmak.

Şirket yukarıda yazılı amaçlarını gerçekleştirmek için;

1) Konusu ile ilgili her türlü ithalat, ihracat ve pazarlama işleri yapar.

iştirak paylarını ve haklarını iktisap edebilir. Aracılık yapmamak kaydıyla satabilir. Yeniden ithal ve inşa ettirebilir. Kiralayabilir, bunları kısmen veya tamamen başkalarına satabilir veya kiraya verebilir. Bunlar üzerinde rehin, ipotek ve sair haklar tesis edebilir ve diğer her türlü hukuki tasarrufta bulunabilir.

**4. Şirketin Merkezi**

Şirketin Merkezi İstanbul İli Şişli İlçesi'dir.

Adresi Gülbahar Mahallesi Şehit Ertuğrul Kabataş Cad. No: 26 A/_ Şişli/İstanbul 'dir.

Adres değişikliğinde yeni adres, ticaret siciline tescil ve Türkiye Ticaret Sicili Gazetesi'nde ilan ettirilir.

Tescil ve ilan edilmiş adrese yapılan tebligat şirkete yapılmış sayılır. Tescil ve ilan edilmiş adresinden ayrılmış olmasına rağmen, yeni adresini süresi içinde tescil ettirmemiş şirket için bu durum fesih sebebi sayılır.

**5. Süre**

Şirketin süresi, kuruluşundan itibaren sınırsız'dır.

**6. Sermaye Ve Pay Senetlerinin Nevi**

Şirketin sermayesi, beheri 25,00 Türk Lirası değerinde 4000 paya ayrılmış toplam 100000,00 Türk Lirası değerindedir.

-Beheri 25,00 Türk Lirası değerinde 4000 adet paya karşılık gelen 100000,00 Türk Lirası Musa Zorlu tarafından nakdi,

olarak taahhüt edilmiştir.

Nakden taahhüt edilen payların itibari değerlerinin 25000,00 TL'si şirketin tescilinden önce ödenmiştir.

Nakden taahhüt edilen payların geri kalan kısmı ise şirketin tescilini izleyen 24 ay içinde ödenecektir.

Yetkilendirilen ticari vekil veya diğer tacir yardımcıları da ticaret siciline tescil ve ilan edilir. Bu kişilerin, şirkete ve üçüncü kişilere verecekleri her tür zarardan dolayı müdürler müteselsilen sorumludur.

**9. Genel Kurul**

Genel Kurullar, olağan ve olağanüstü toplanırlar. Olağan genel kurul, her yıl hesap döneminin sona ermesinden itibaren 3 ay içinde; olağanüstü genel kurullar ise, Şirket işlerinin gerektirdiği hallerde ve zamanlarda toplanır.

Genel kurul toplantılarında, her ortağın oy hakkı, esas sermaye paylarının itibari değerine göre hesaplanır. Genel kurul toplantıları ve bu toplantılardaki karar nisabı, Türk Ticaret Kanunu hükümlerine tabidir.

Genel kurul, şirketin merkez adresinde veya yönetim merkezinin bulunduğu şehrin elverişli bir yerinde toplanır.

**10. İlan**

Genel kurulun toplantıya çağrılmasına ilişkin ilanlar da dahil olmak üzere Şirkete ait ilanlar Türkiye Ticaret Sicili Gazetesinde yapılır. Genel kurul toplantılarına ilişkin ilanların toplantı gününden en az on gün önce yapılması zorunludur.

**11. Hesap Dönemi**

Şirketin hesap yılı, Ocak ayının 1. gününden başlar ve Aralık ayının 31. günü sona erer. Fakat birinci hesap yılı, Şirketin kesin olarak kurulduğu tarihten itibaren başlar ve o senenin aralık ayının otuz birinci günü sona erer.

**12. Kârın Tespiti Ve Dağıtımı**

Şirketin net dönem karı yapılmış her çeşit masrafların çıkarılmasından sonra kalan miktardır. Net dönem kârından her yıl %5 genel kanuni yedek akçe

(Devamı 604 . Sayfada)

**Figure 1:** Ticari Sicil Gazetesi (TSG) declarations

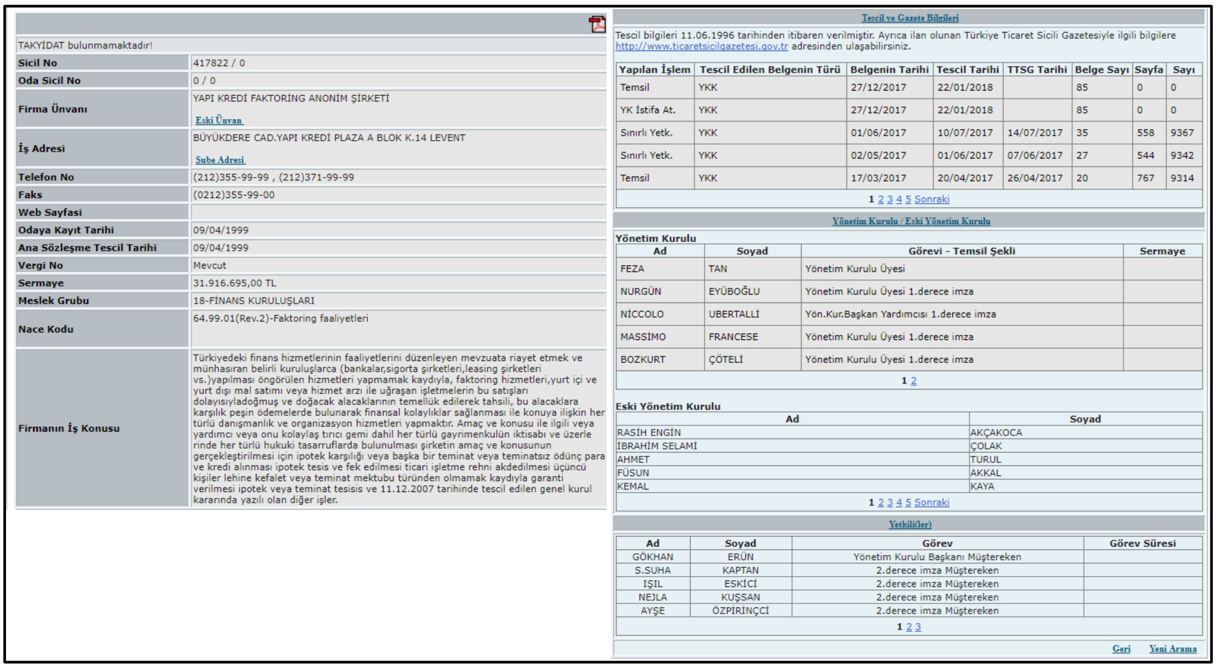



**Figure 2:** Ticaret Odası Kayıtları (TOK) example company info

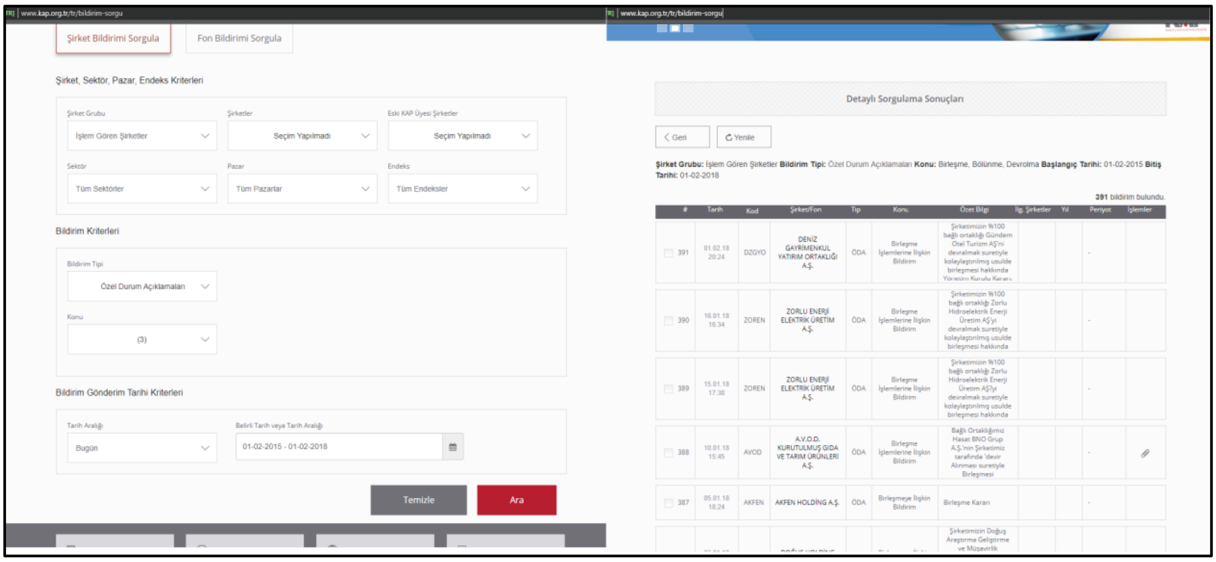
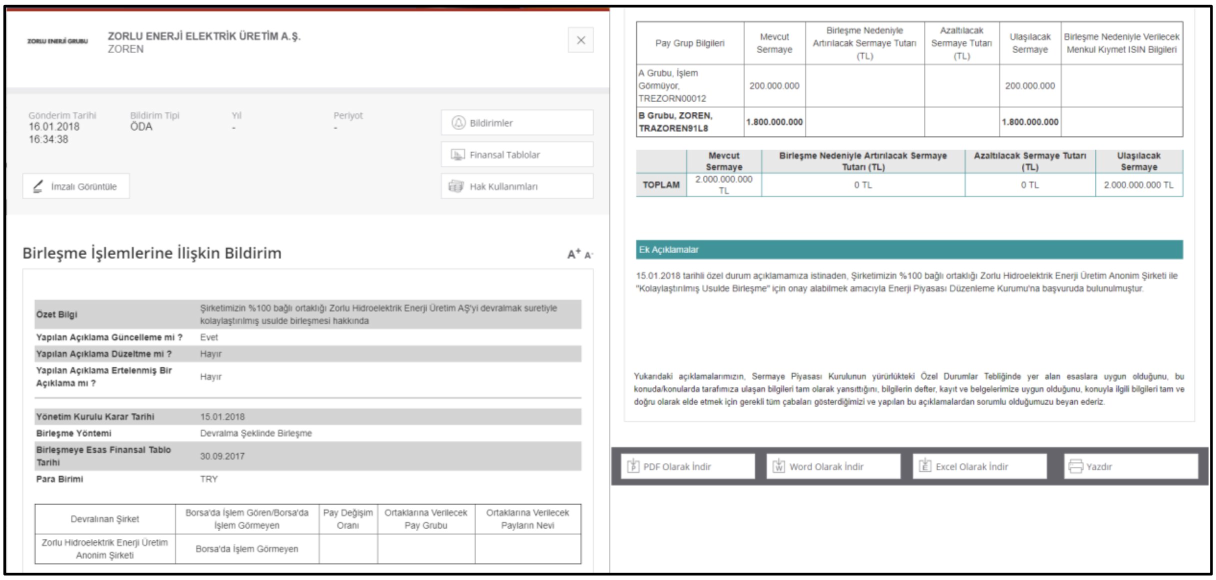



**Figure 3:** Kamu Aydınlatma Platformu (KAP) before/after search



ZORLU ENERJİ ELEKTRİK ÜRETİM A.Ş.
ZOREN

Gönderim Tarihi 16.01.2018 16:34:38 | Bildirim Tipi ÖDA | Yıl - | Periyot -

Bildirimler | Finansal Tablolar | Hak Kullanımları | İmzalı Görüntüle

Birleşme İşlemlerine İlişkin Bildirim

| | |
|---|---|
| Özet Bilgi | Şirketimizin %100 bağlı ortaklığı Zorlu Hidroelektrik Enerji Üretim AŞ'yi devralmak suretiyle kolaylaştırılmış usulde birleşmesi hakkında |
| Yapılan Açıklama Güncelleme mi ? | Evet |
| Yapılan Açıklama Düzeltme mi ? | Hayır |
| Yapılan Açıklama Ertelenmiş Bir Açıklama mı ? | Hayır |
| Yönetim Kurulu Karar Tarihi | 15.01.2018 |
| Birleşme Yöntemi | Devralma Şeklinde Birleşme |
| Birleşmeye Esas Finansal Tablo Tarihi | 30.09.2017 |
| Para Birimi | TRY |

| Devralınan Şirket | Borsa'da İşlem Gören/Borsa'da İşlem Görmeyen | Pay Değişim Oranı | Ortaklarına Verilecek Pay Grubu | Ortaklarına Verilecek Payların Nevi |
|---|---|---|---|---|
| Zorlu Hidroelektrik Enerji Üretim Anonim Şirketi | Borsa'da İşlem Görmeyen | | | |

| Pay Grup Bilgileri | Mevcut Sermaye | Birleşme Nedeniyle Artırılacak Sermaye Tutarı (TL) | Azaltılacak Sermaye Tutarı (TL) | Ulaşılacak Sermaye | Birleşme Nedeniyle Verilecek Menkul Kıymet ISIN Bilgileri |
|---|---|---|---|---|---|
| A Grubu, İşlem Görmüyor, TREZORN00012 | 200.000.000 | | | 200.000.000 | |
| **B Grubu, ZOREN, TRAZOREN91L8** | **1.800.000.000** | | | **1.800.000.000** | |

| | Mevcut Sermaye | Birleşme Nedeniyle Artırılacak Sermaye Tutarı (TL) | Azaltılacak Sermaye Tutarı (TL) | Ulaşılacak Sermaye |
|---|---|---|---|---|
| TOPLAM | 2.000.000.000 TL | 0 TL | 0 TL | 2.000.000.000 TL |

Ek Açıklamalar

15.01.2018 tarihli özel durum açıklamamıza istinaden, Şirketimizin %100 bağlı ortaklığı Zorlu Hidroelektrik Enerji Üretim Anonim Şirketi ile "Kolaylaştırılmış Usulde Birleşme" için onay alabilmek amacıyla Enerji Piyasası Düzenleme Kurumu'na başvuruda bulunulmuştur.

Yukarıdaki açıklamalarımızın, Sermaye Piyasası Kurulunun yürürlükteki Özel Durumlar Tebliğinde yer alan esaslara uygun olduğunu, bu konuda/konularda tarafımıza ulaşan bilgileri tam olarak yansıttığını, bilgilerin defter, kayıt ve belgelerimize uygun olduğunu, konuyla ilgili bilgileri tam ve doğru olarak elde etmek için gerekli tüm çabaları gösterdiğimizi ve yapılan bu açıklamalardan sorumlu olduğumuzu beyan ederiz.

PDF Olarak İndir | Word Olarak İndir | Excel Olarak İndir | Yazdır

**Figure 4:** Kamu Aydınlatma Platformu (KAP) example declaration

# B - Feature keywords lists

- **52 relation keywords & weights list**

| | | | | | |
|---|---|---|---|---|---|
| Birleşme | 598 | bölünmüştür | 5 | devrolma | 4 |
| birleşmesi | 72 | bölünmenin | 1 | devrine | 3 |
| birleşilmesi | 69 | bölünmesi | 1 | devrolunacak | 3 |
| birleşmesine | 46 | devir | 125 | devralmıştır | 2 |
| birleşilmesine | 41 | devralınması | 61 | devredilmiştir | 2 |
| birleşmeye | 31 | devralma | 36 | devrettiği | 2 |
| birleşmenin | 21 | devralan | 27 | devralmasına | 2 |
| birleştirilmesine | 11 | devrolacak | 27 | devredilmesinin | 1 |
| birleştirilmesi | 9 | devralması | 22 | devralınacak | 1 |
| birleşmesinin | 2 | devrolan | 18 | devralınan | 1 |
| birleştirilme | 2 | devrolunması | 16 | devredilecek | 1 |
| birleşmede | 2 | devralmak | 15 | devredilen | 1 |
| birleşmeden | 1 | devrolunan | 14 | devrolunmasına | 1 |
| birleşmiştir | 1 | devralınmasına | 12 | devralınmış | 1 |
| birleştirilmesiyle | 1 | devredilmesi | 12 | devralınarak | 1 |
| bölünme | 85 | devrolması | 9 | devrolunarak | 1 |
| bölünmeye | 19 | devri | 6 | | |
| bölünen | 7 | devredilmesine | 4 | | |

- **124 relation keywords & weights list**

| | |
|---|---|
| birleşecek | 1 |
| birleşeceğini | 1 |
| birleşerek | 1 |
| birleşilmesi | 69 |
| birleşilmesine | 41 |
| birleşiyor | 1 |
| birleşme | 598 |
| birleşmede | 2 |
| birleşmeden | 1 |
| birleşmeleri | 1 |
| birleşmelerinde | 1 |
| birleşmelerine | 1 |
| birleşmelerini | 1 |
| birleşmenin | 21 |
| birleşmesi | 72 |
| birleşmesinden | 1 |
| birleşmesine | 46 |
| birleşmesinin | 3 |
| birleşmesiyle | 1 |
| birleşmeye | 31 |
| birleşmeyi | 1 |
| birleşmeyle | 1 |
| birleşmiştir | 1 |
| birleştirilirken | 1 |
| birleştirilme | 2 |
| birleştirilmesi | 10 |
| birleştirilmesine | 12 |
| birleştirilmesini | 2 |
| birleştirilmesiyle | 1 |
| birleştirilmiştir | 2 |
| birleştirme | 2 |
| birleştirmek | 1 |
| birleştirmesinden | 1 |
| bölündü | 2 |
| bölünen | 7 |
| bölünerek | 1 |
| bölünme | 85 |
| bölünmenin | 1 |
| bölünmesi | 1 |
| bölünmesiyle | 1 |
| bölünmeye | 19 |
| bölünmüştür | 5 |
| devir | 125 |
| devirden | 1 |
| devire | 1 |
| devirle | 1 |
| devirler | 1 |
| devirleri | 1 |
| devirlerin | 1 |
| devirlerinin | 1 |
| devirlerle | 1 |
| devralacak | 1 |
| devralan | 27 |
| devralarak | 1 |
| devraldı | 1 |
| devraldığını | 1 |
| devralinan | 1 |
| devralinarak | 1 |
| devralma | 36 |
| devralmadan | 1 |
| devralmak | 15 |
| devralmalar | 1 |
| devralmalarına | 1 |
| devralması | 22 |
| devralmasına | 2 |
| devralmasıyla | 1 |
| devralmış | 1 |
| devralmıştır | 2 |
| devralınacak | 2 |
| devralındığı | 1 |
| devralınması | 61 |
| devralınmasına | 12 |
| devralınmasını | 1 |
| devralınmasıyla | 1 |
| devralınmış | 1 |
| devralınmıştır | 1 |
| devredecek | 1 |
| devredeceğini | 1 |
| devreden | 2 |
| devredenin | 1 |
| devrederek | 1 |
| devredildiği | 2 |
| devredilecek | 2 |
| devredileceği | 1 |
| devredilen | 2 |
| devredilirken | 2 |
| devredilmedi | 1 |
| devredilmek | 1 |
| devredilmesi | 12 |
| devredilmesine | 4 |
| devredilmesinin | 1 |
| devredilmesiyle | 1 |
| devredilmiş | 1 |
| devredilmiştir | 2 |
| devrediyor | 1 |
| devretme | 1 |
| devretmek | 1 |
| devretmesi | 1 |
| devretmesinin | 1 |
| devretmeye | 1 |
| devretmiş | 1 |
| devretmişlerdir | 1 |
| devretmişti | 1 |
| devretmiştir | 1 |
| devretti | 1 |
| devrettiği | 2 |
| devrettiğini | 1 |
| devri | 7 |
| devrin | 1 |
| devrinde | 1 |
| devrinden | 1 |
| devrine | 3 |
| devrini | 1 |
| devrinin | 1 |
| devriyle | 1 |
| devrolacak | 27 |
| devrolan | 18 |
| devrolma | 4 |
| devrolması | 9 |
| devrolunacak | 3 |
| devrolunan | 14 |
| devrolunarak | 1 |
| devrolunması | 16 |
| devrolunmasına | |

- **1k keywords & weights list (sample)**

| | | | | | |
|---|---|---|---|---|---|
| A.Ş. | 1235 | BAHSE | 9 | BUGÜN | 68 |
| ADET | 69 | BAHSİ | 26 | BUGÜNKÜ | 10 |
| ADETLERİNİN | 43 | BAKANLIĞI | 9 | BULUNAN | 45 |
| ADINA | 13 | BANKA | 9 | BULUNDUĞU | 9 |
| ADRESLİ | 9 | BANKACILIK | 16 | BULUNULMASINA | 19 |
| ADRESİNDE | 8 | BANKAMIZ | 13 | BULUNULMUŞTUR | 9 |
| ADİL | 30 | BANKASI | 26 | BULUNUYOR | 18 |
| AKTİF | 252 | BAZI | 8 | BUNA | 28 |
| ALAN | 89 | BAĞIMSIZ | 104 | BUNLARLA | 44 |
| ALDI | 14 | BAĞLI | 110 | BÖLÜNEN | 11 |
| ALDIĞI | 11 | BAŞINDA | 9 | BÖLÜNME | 343 |
| ALIM | 13 | BAŞKANI | 25 | BÖLÜNMENİN | 1 |
| ALINACAK | 16 | BAŞLANILMASI | 9 | BÖLÜNMESİ | 1 |
| ALINAN | 56 | BAŞLANILMASINA | 11 | BÖLÜNMEYE | 19 |
| ALINARAK | 89 | BAŞLAYAN | 9 | BÖLÜNMÜŞTÜR | 5 |
| ALINDIĞI | 11 | BAŞLIKLI | 29 | BÜNYESİNDE | 179 |
| ALINMAK | 9 | BAŞLIĞINI | 88 | BÜTÜN | 225 |
| ALINMASI | 118 | BAŞTA | 9 | BÜYÜK | 48 |
| ALINMASINA | 71 | BAŞVURU | 65 | BÜYÜME | 18 |
| ALINMIŞ | 11 | BAŞVURUDA | 37 | BİLAHARE | 8 |
| ALINMIŞTIR | 12 | BAŞVURULARIN | 11 | BİLANÇOSU | 15 |
| ALIŞVERİŞ | 17 | BAŞVURUSU | 11 | BİLDİRİMİN | 11 |
| ALMA | 28 | BDDK | 24 | BİLGİ | 49 |
| ALMAK | 18 | BEDELSİZ | 12 | BİLGİSİNE | 29 |
| ALMAKTA | 12 | BEDELİNİN | 9 | BİRLEŞME | 1240 |
| ALMAKTADIR | 36 | BELGELER | 44 | BİRLEŞMEDE | 2 |
| ALMIŞ | 10 | BELGELERDE | 10 | BİRLEŞMEDEN | 1 |
| ALMIŞLARDIR | 8 | BELGELERLE | 14 | BİRLEŞMENİN | 27 |
| ALTINDA | 30 | BELGELERİN | 41 | BİRLEŞMESİ | 76 |
| AMACIYLA | 36 | BELGESİ | 11 | BİRLEŞMESİNE | 46 |
| AMAÇLA | 10 | BELLİ | 9 | BİRLEŞMESİNİN | 2 |
| ANA | 27 | BELİRLENECEK | 18 | BİRLEŞMEYE | 58 |
| ANCAK | 15 | BELİRLENEN | 23 | BİRLEŞMİŞTİR | 1 |
| ANILAN | 40 | BELİRLENMESİNE | 26 | BİRLEŞTİRİLME | 2 |
| ANLAŞMA | 8 | BELİRLENMİŞ | 13 | BİRLEŞTİRİLMESİ | 11 |
| ANONİM | 82 | BELİRTEN | 10 | BİRLEŞİLMESİ | 69 |
| ARA | 26 | BELİRTİLDİĞİ | 18 | BİRLEŞİLMESİNE | 41 |
| ARACI | 8 | BELİRTİLEN | 42 | BİRLİKTE | 123 |
| ARACILIĞIYLA | 8 | BELİRTİLMESİNE | 12 | BİRLİĞİ | 13 |
| ARASINDA | 44 | BENDİ | 13 | BİRİNCİ | 13 |
| ARDINDAN | 25 | BORSA | 119 | CEO | 9 |
| ARTIRILAN | 14 | BU | 313 | CO | 37 |

- **1k set reduced to 423**

A.Ş. ADET ADIN ADRE ADİL AKTİ ALAN ALDI ALIM ALIN ALIŞ ALMA
ALMI ALTI AMAC AMAÇ ANCA ANIL ANLA ANON ARAC ARAS ARDI ARTI
AVRU AYDI AYNI AYRI AZIN AÇIK AÇIL AÇIS AĞIR AŞAĞ AŞIL BAHS
BAKA BANK BAZI BAĞI BAĞL BAŞI BAŞK BAŞL BAŞT BAŞV BDDK BEDE
BELG BELL BELİ BEND BORS BUGÜ BULU BUNA BUNL BÖLÜ BÜNY BÜTÜ
BÜYÜ BİLA BİLD BİLG BİRL BİRİ DAHA DAHİ DAKİ DANI DAĞI DEDİ
DEFA DEKİ DENE DENİ DEST DEVA DEVL DEVR DEVİ DEĞE DEĞİ DIŞI
DOLA DOĞR DURU DUYU DÖNE DÜNY DÜZE DİJİ DİKK DİLE DİPN DİYE
DİĞE EDEC EDEN EDİL EDİY EKLİ EKON EKTE EKİN ELEK EMLA ENDÜ
ENER ESAS ETKİ ETMİ ETTİ EŞİT FAAL FABR FAZL FIKR FİNA FİYA
GAYR GELE GELM GELİ GENE GERE GERÇ GETİ GEÇE GEÇM GEÇİ GIDA
GLOB GROU GRUB GRUP GÖRE GÖRÜ GÖST GÖZE GÜNC GÜND GÜNL GÜNÜ
GÜÇL GİBİ GİRE GİRİ HAKK HAKL HALK HALİ HARİ HAVA HAYV HAZI
HAZİ HEME HERH HESA HIZL HOLD HUSU HÜKM HÜKÜ HÜRR HİSS HİZM
HİÇB INCI KABU KADA KALA KAMU KANU KAPA KAPS KARA KARŞ KASI
KATI KATL KAYD KAYI KAYN KEND KILI KISM KOLA KONS KONT KONU
KULL KURU KİRA KİŞİ LOJİ LİRA LİST MADD MAKU MAKİ MALV MALZ
MALİ MARK MEMU MENK MERK METN MEVC MEVZ MODE MUHA MÜDÜ MÜRA
MÜZİ MÜŞA MİLY NAKİ NDAN NEDE NETİ NEZD NOMİ NUMA NİTE OLAC
OLAN OLAR OLAĞ OLDU OLMA OLMU OLUM OLUP OLUŞ ONAY OPER ORAN
ORTA OTEL OTOM OYBİ PASİ PAYA PAYI PAYL PAZA PERA PETR PLAS
PLAT PORT PROF PROJ PİYA RAPO REKA RESM REVİ SADE SAHİ SAN.
SANA SATA SATI SAYI SAİR SEAN SEBE SEKT SERM SERV SERİ SIFA
SINA SINI SMMM SONR SONU SPKN STAN STRA SUNU SURE SÖYL SÖZL
SÜRE SİCİ SİST SİTE T.A. TAAH TABL TABİ TADİ TAHM TAHS TAKD
TAKI TAKİ TALE TAMA TARA TARI TARİ TASF TASL TAVA TAŞI TEBL
TEKA TEKR TEKS TEKİ TELE TEMS TEMİ TERE TESC TESP TOPL TSKB
TURİ TUTA TÜRK TÜRL TİC. TİCA ULUS UNSU USUL UYAR UYGU UYUM
UZLA UZMA UZUN VARL VARS VAZG VERE VERG VERM VERİ VEYA YABA
YAKL YAPA YAPI YAPM YAPT YASA YATI YAYI YAZI YEMİ YENİ YERİ
YETK YEŞİ YILD YILI YILL YOLU YUKA YURT YÖNE YÖNT YÜKS YÜKÜ
YÜRÜ YÜZD YİNE ZAMA ZORL ZORU ÇALI ÇATI ÇERÇ ÇIKA ÖDEM ÖDEN
ÖNCE ÖNEM ÖNGÖ ÖNÜM ÖZEL ÖZET ÜLKE ÜNCÜ ÜRET ÜRÜN ÜZER İFAD
İFAS İHRA İKİN İLAN İLET İLGİ İLİŞ İMZA İNCE İNCİ İNŞA İSTİ
İTİB İZİN İÇEC İÇER İÇİN İŞBU İŞLE İŞTİ ŞART ŞEKL ŞEKİ ŞERH
ŞUNL ŞÖYL ŞİRK

- **124 words set reduced to 47**

| | | | | | |
|---|---|---|---|---|---|
| BÖLÜNDÜ | BÖLÜNEN | BÖLÜNERE | BÖLÜNME | BÖLÜNMEN | BÖLÜNMES |
| BÖLÜNMEY | BÖLÜNMÜŞ | BİRLEŞEC | BİRLEŞER | BİRLEŞME | BİRLEŞMİ |
| BİRLEŞTİ | BİRLEŞİL | BİRLEŞİY | DEVRALAC | DEVRALAN | DEVRALAR |
| DEVRALDI | DEVRALIN | DEVRALMA | DEVRALMI | DEVRALİN | DEVREDEC |
| DEVREDEN | DEVREDER | DEVREDİL | DEVREDİY | DEVRETME | DEVRETMİ |
| DEVRETTİ | DEVROLAC | DEVROLAN | DEVROLMA | DEVROLUN | DEVRİ |
| DEVRİN | DEVRİNDE | DEVRİNE | DEVRİNİ | DEVRİNİN | DEVRİYLE |
| DEVİR | DEVİRDEN | DEVİRE | DEVİRLE | DEVİRLER | |

- **124 words set reduced to 15**

| | | | | | |
|---|---|---|---|---|---|
| BÖLÜND | BÖLÜNE | BÖLÜNM | BİRLEŞ | DEVRAL | DEVRED |
| DEVRET | DEVROL | DEVRİ | DEVRİN | DEVRİY | DEVİR |
| DEVİRD | DEVİRE | DEVİRL | | | |

- **124 words set reduced to 5**

| | | | | | |
|---|---|---|---|---|---|
| BÖLÜN | BİRLE | DEVRA | DEVRE | DEVRO | DEVRİDEVİR |

- **124 words set reduced to 4**

| | | | |
|---|---|---|---|
| BÖLÜ | BİRL | DEVR | DEVİ |