COMPARATIVE SENTENCE ANALYSIS IN TURKISH DOCUMENTS

by

Bartu İnce

Submitted to the Department of Computer
Engineering in partial fulfillment of
the requirements for the degree of
Bachelor of Science

Undergraduate Program in Computer Engineering
Boğaziçi University

Spring 2019

COMPARATIVE SENTENCE ANALYSIS IN TURKISH DOCUMENTS

APPROVED BY:

Prof. Tunga Güngör .....
(Project Supervisor)

DATE OF APPROVAL: 30.01.2019

ACKNOWLEDGEMENTS

I would like to thank Prof. Dr. Gülşen Cebiroğlu Eryiğit for allowing me to access the ITU NLP Pipeline and its API for my project.

ABSTRACT

COMPARATIVE SENTENCE ANALYSIS IN TURKISH DOCUMENTS

This project revolves around the concept of analyzing comparative sentences in Turkish text documents. This analysis process involves identification of whether a sentence possesses comparative qualities, and if so, extracting the part that is actually involved in the comparison. The idea of classifying these types of sentiments is immensely helpful in decision making based on feedback. Quite a lot of business decisions are made depending on the opinions about their products that they are presented with, which allows them to take actions accordingly. Therefore, finding comparisons, and detecting the two objects or concepts that are being compared, alongside the factors they are being compared upon is the primary goal of the work.

ÖZET

TÜRKÇE BELGELERDE KIYASLAMALI CÜMLE ANALİZİ

Bu proje, Türkçe belgelerde kıyaslamalı cümlelerin analizi fikri üzerinde uğraşmaktadır. Bu analiz süreci bir cümlenin kıyaslayıcı özelliklere sahip olup olmadığını belirlemek, ve eğer varsa, kıyaslama ile alakakı olan kısmı ortaya çıkarmaktan oluşmaktadır. Bu çeşit duygusal yorumları sınıflandırma fikri, geri bildirimlere dayanarak karar verme konusunda oldukça kullanışlıdır. Bir çok iş kararı sundukları ürünler hakkında karşılarına sunulan düşüncelere dayanılarak verilir. Bu karar verme onların en doğru şekilde harekete geçmelerini sağlar. Bu yüzden, kıyaslamaları bulmak, kıyaslama yapılan iki objeyi veya fikri tespit edebilmek, ve hangi yönlerden karşılaştırıldıklarını anlayabilmek bu projenin temel amacıdır.

TABLE OF CONTENTS

LIST OF FIGURES

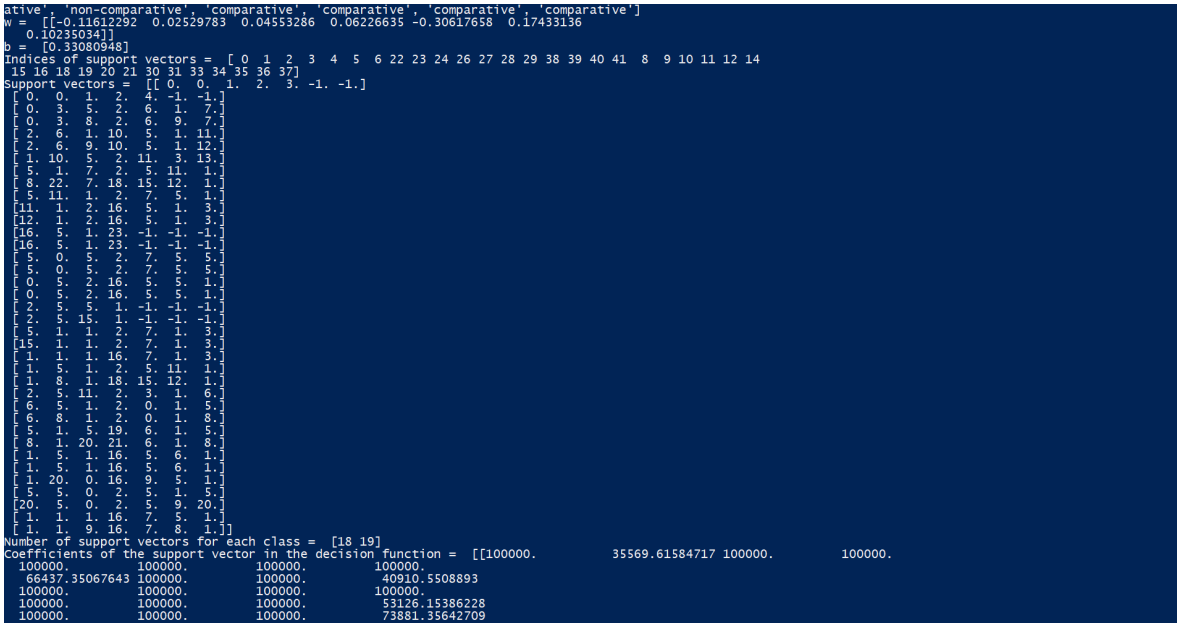


Figure 1. SVM results

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

SVM	Support Vector Machine
NB	Naive Bayesian
NLP	Natural Language Processing
POS	Part of Speech
API	Application Programming Interface
ID	Identifier

1. INTRODUCTION AND MOTIVATION

- There are many cases in our daily life where something needs to be judged by its qualities, causing it to be graded based on how well these qualities perform compared to its competitors. Of course, not all of these judgments are extensively explained. In fact, most of them are little more than personal observations. This makes them difficult to detect and analyze. This projects aims to bring an improvement to the field by detecting all sorts of comparative sentences, and deeply analyzing them. There is no doubt that this kind of information will be valuable for businesses.

2. STATE OF THE ART

- As far as our research goes, there has not been any research done on this subject in Turkish. Even the English counterpart can be considered somewhat lacking, as there are perhaps 1 or 2 papers published on the matter, made by the same group[1]. There exist a few more papers about this on other languages as well (Chinese, Korean etc.), but they are written in their native languages, making them hard to access.
- However, as we go through more layers of abstraction, more data naturally becomes available to us. Experiments have been made about detecting subjectivity and extracting knowledge from a sentence[2], and a book has been published about coordination and comparatives as a whole[3]. These researches, while not being directly related to our project, help us come up with various strategies and understand the subject matter better.
- Thankfully, one important factor of our experiment is also part of this available data. ITU's publicly available NLP Pipeline system[4] was the result of a tremendous amount of effort, and it allows us to easily examine the properties of every single word in a given sentence. This is crucial for the methods we have followed, as the project revolves around what the words represent, rather than what they say.

3. METHODS

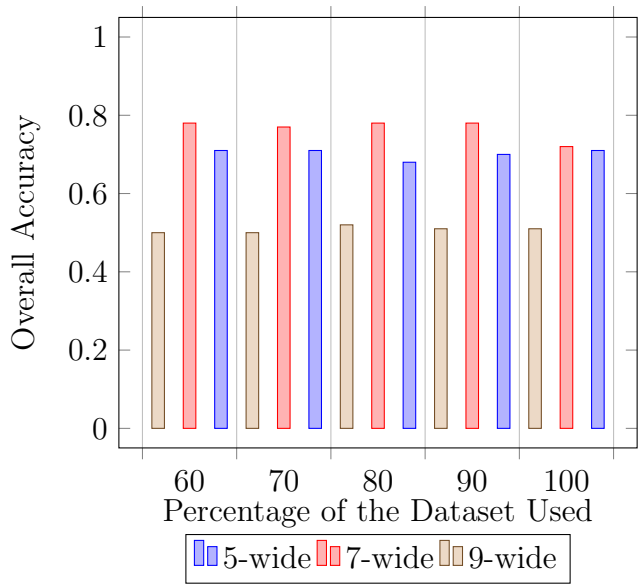
- After arduous experimentation, I believe that our methods have led us to a satisfying conclusion.
- As mentioned previously, our starting point was provided by ITU Turkish Natural Language Processing Pipeline. Once they gave me permission to use their pipeline as an API, I was able to process any given sentence, and give the properties of every word inside the sentence, along with its relation to other words located in it. Most importantly, I was able to see the POS tag of each word, which is crucial for finding and generating word sequences. Python was the language that I used, as that seemed to have the shortest setup time.
- Afterwards, I took the time to gather example sentences, both comparative and non-comparative. The gathered dataset is somewhat uniform, as it does not make use of too many different sources, but it should not be too hard to expand it in the future if necessary. One thing of note is that the amount of non-comparative sentences is significantly higher than the amount of comparative sentences, which seems to reflect how they are used in daily speech.
- Once the API processed a sentence, I acquired 8 fields of data for each word contained within it. Almost all of these fields were useful, but the parts I was interested in were the ones that provided the POS tags. So I took these fields of data and turned them into a dictionary, which made processing the data much easier, as they originally came as a string. After that, I stored each of the dictionaries into a list, and wrote the list into a file, to reduce the need for future API usage.
- When that was done, I decided on the kinds of keywords or suffixes that could be used for comparison. The obvious choices were the most commonly used words that are used when comparing things ('daha', 'kadar', 'gibi', 'göre', 'hem', '-dan').
- I searched for our keywords/suffixes in the sentence dictionaries. Once it found a match, I took the words, or more accurately, their POS tags, in a certain radius of the matched word, and created a sequence from them.
- After applying this search to all sentences, I saved all the found sequences along

with their class (comparative/non-comparative), and tried to create a detection model using an SVM library. I had to convert each POS tag into a number for this phase. One problem that arose here was that not all sequences have the same length, as sentences may end or start sooner than expected. As a result, I had to pad the lacking sequences with an unused number (-1), which may have an impact on accuracy and feature weights.

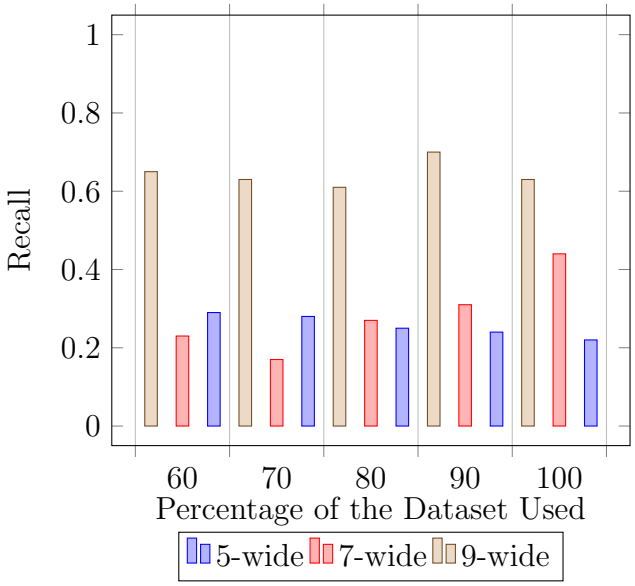
- Originally, the "POS to number" process simply consisted of assigning an ID to a POS tag when it appears for the first time, and using that ID for any future encounters. However, that seemed to have severely skewed the results, as it was not an accurate representation of its weight in the dataset.
- As a solution to this, I decided that measuring the frequency values of the POS tags was a much healthier approach, as the algorithm could decide how important each tag is this way. By calculating the raw frequency, and then occurrence of the tags, I was able to obtain values that defined each tag's importance. In this approach, the "padding" tags simply had an importance of 0.
- This alone was not enough to obtain absolute results, however. This is where I noticed that the dataset was somewhat skewed, as the SVM seemed to decide all test data was either completely non-comparative or completely comparative. It was an expected result, as I have mentioned earlier that the amount of comparative sentences is lacking compared to the non-comparative ones. To get around this issue, I needed to carefully adjust the ratio of sentences taken from both classes.
- Finally, I changed the control settings for a larger result set. The radius of the words taken from the sentence (e.g. 3 words before and after the keyword), and the percentage of the data set that was used for training and testing (e.g. 60 percent of the 1000 sentences) were edited for differing results.

4. RESULTS

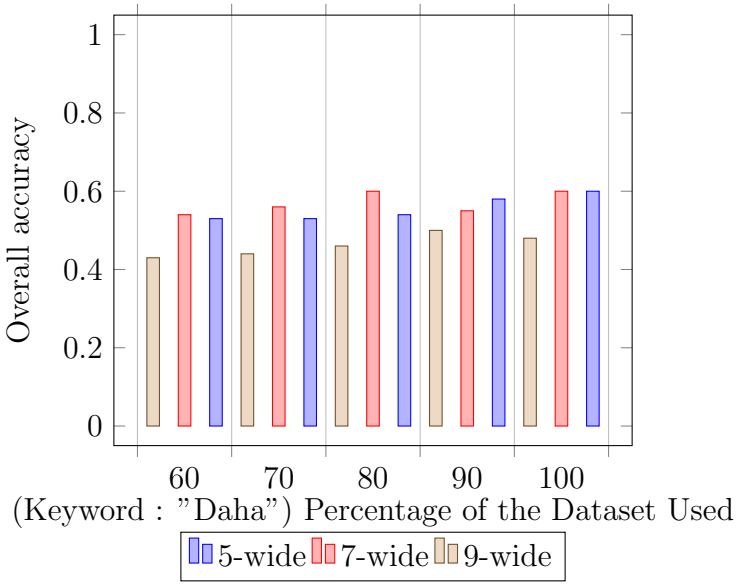
- I have managed to obtain a variety of results, based on the amount of variables changed.

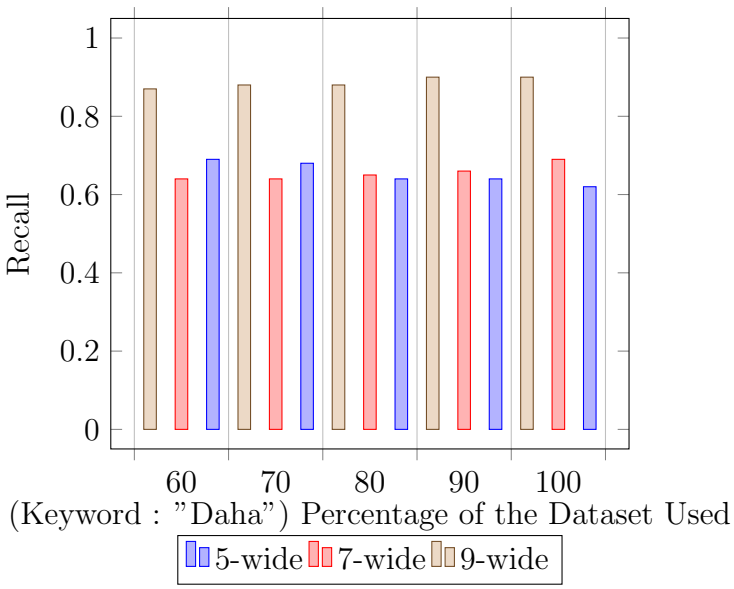


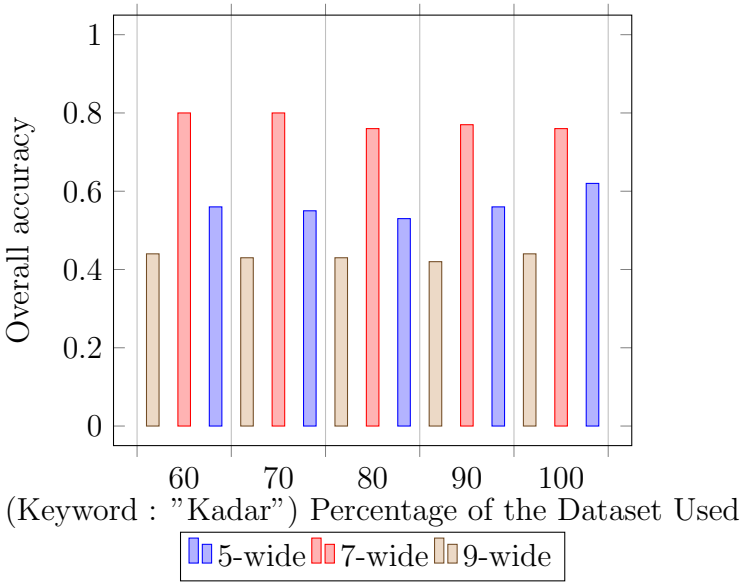
- Interestingly, from these values, it would seem that the 9-wide option (4 words before and after the keyword) is the least accurate of the three options by far. However, when checked in detail, it can be seen that it has a much greater recall value.

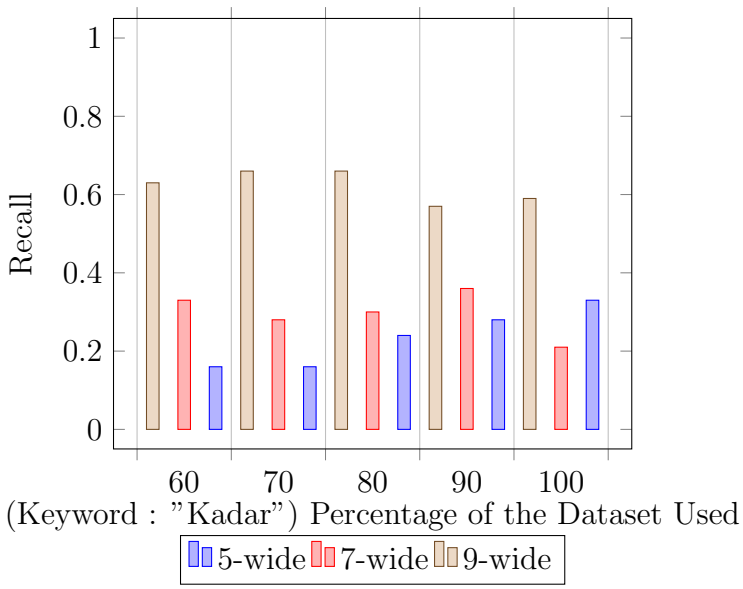


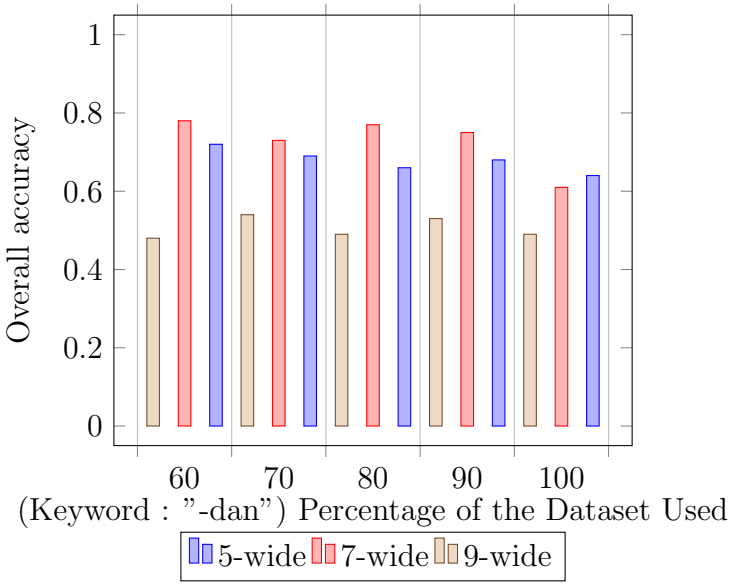
- The ratio of the data set that was taken was completely unchanged, leading me to believe that using 9 words is an important factor on correctly deducing comparative sentences.
- Here are a few more charts depicting the accuracy and recall values for various keywords in particular.

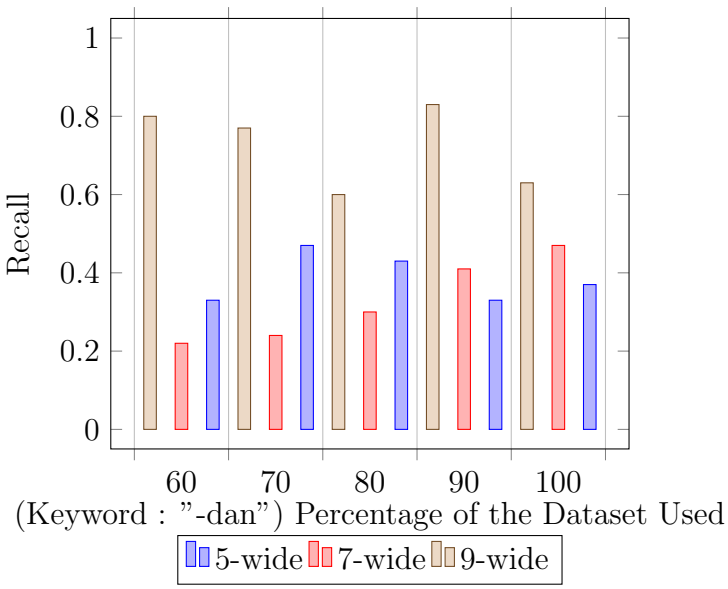

Overall accuracy
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : "Daha") Percentage of the Dataset Used
5-wide
7-wide
9-wide


Recall
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : "Daha") Percentage of the Dataset Used
5-wide
7-wide
9-wide


Overall accuracy
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : "Kadar") Percentage of the Dataset Used
5-wide
7-wide
9-wide


Recall
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : "Kadar") Percentage of the Dataset Used
5-wide
7-wide
9-wide


Overall accuracy
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : ”-dan”) Percentage of the Dataset Used
5-wide
7-wide
9-wide


Recall
1
0.8
0.6
0.4
0.2
0
60
70
80
90
100
(Keyword : ”-dan”) Percentage of the Dataset Used
5-wide
7-wide
9-wide

5. CONCLUSION AND DISCUSSION

- From the data I have gathered, we can reach the conclusion that the SVM method is certainly an interesting strategy. It was highly sensitive to the number of sentences that was provided for training, with the most accurate results being the ones where the comparative and non-comparative sentences were provided roughly equally. Making additions to the data set by getting more example sentences, especially comparative ones, can be very helpful. Additionally, obtaining the sentences from varied sources might help as well, as people's way of phrasing sentences tends to be different.
- Looking at the results, it seems that choosing 7-word wide features brings out the highest accuracy. However, since a majority of the test data (and the training data) consists of non-comparative sentences, it is questionable as to how useful the accuracy metric is. On the other hand, the 9-word wide features have the highest recall value, with a minor loss in accuracy. While I do not know if choosing an even longer sequence of words would bring the recall further up, I can deduce that choosing 9-word wide features are optimal for this particular experiment.
- One thing of note is that, the experiments in [1] also made use of Naive Bayesian classification. Perhaps this setup could also be tried with that system in the future.

6. FUTURE WORK

- As there aren't many other papers published on this particular subject, the future of this project is somewhat open ended. If the methods that were discussed in the Conclusion section are taken into account, or if someone comes up with a new approach, I'm certain progress can be made.
- The primary objective of any future work should be to create a much larger corpus, with more attention given to the equality of class ratios if possible. Additionally, improving the list of keywords, alongside improving the complexity of Turkish POS tags in general can be a massive step in progress.

REFERENCES

[1]https://www.cs.uic.edu/~liub/publications/sigir06-comp.pdf

[2]https://www.cs.ubc.ca/labs/lci/papers/docs2005/careninikcap05.pdf

[3]http://www.ai.mit.edu/projects/dm/theses/moltmann92.pdf

[4]http://tools.nlp.itu.edu.tr/index.jsp

APPENDIX A: DATA AVAILABILITY STATEMENT

- If anyone wishes to try out this system for themselves, `http://tools.nlp.itu.edu.tr/index.jsp` has the service available for public use. Keep in mind that it is required to get permissions for an account if you wish to make use of the API.

  A demonstrative version of the project is available at `https://github.com/bartuinceQR/Comparative-Sentence-Finder` if anyone wishes to try it out. Keep in mind that you will need to have access to the ITU NLP Pipeline, as the API needs a verification token to function.

APPENDIX B: STANDARDS, LAWS, REGULATIONS AND DIRECTIVES

•