November 4, 2021

GTU Department of Computer Engineering
CSE 484/654 Natural Language Processing
Fall 2021 - Homework 1 Report

Akif Kartal
171044098

1 Problem Definition

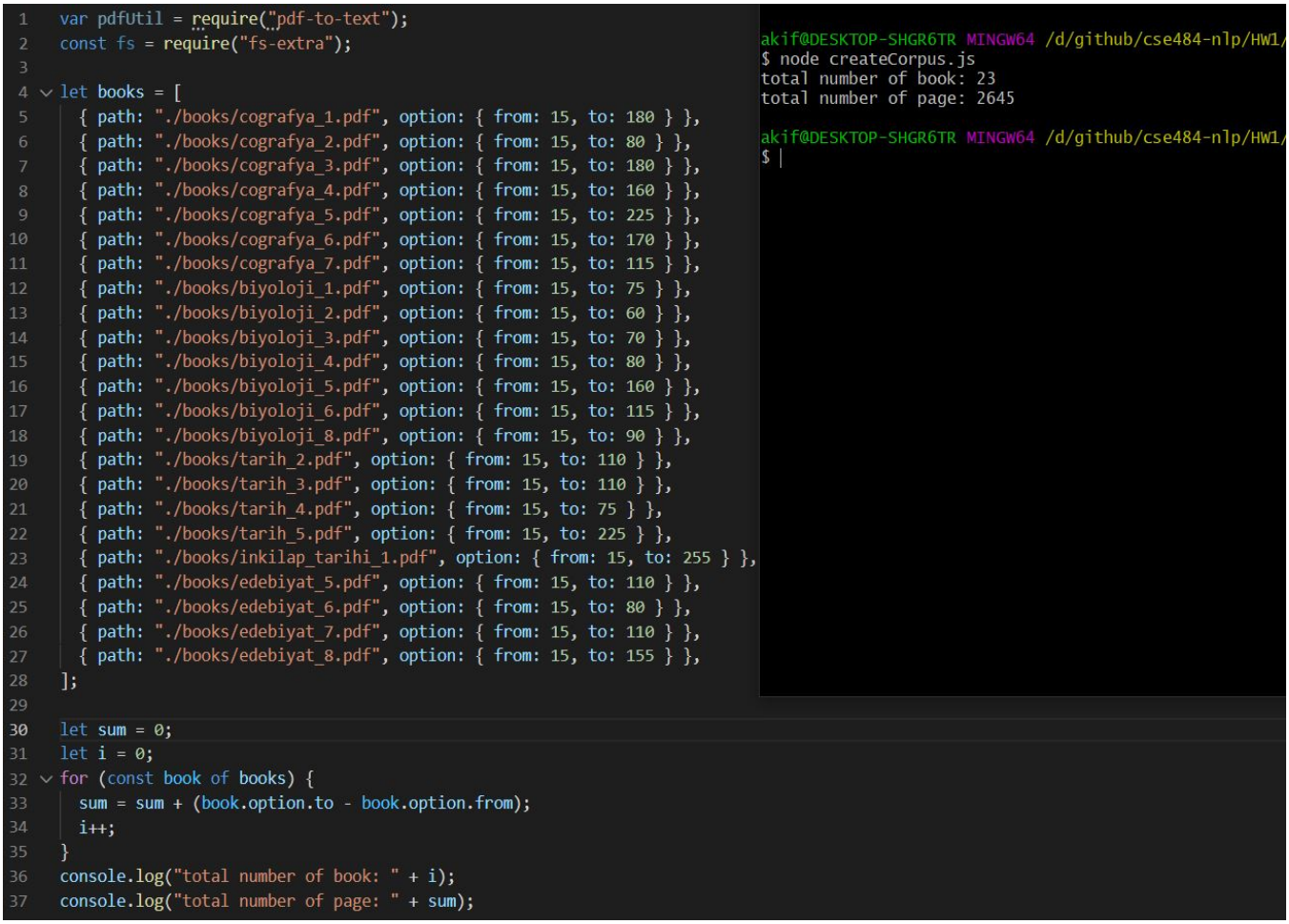
The problem is to make experiments with vector representation of words for turkish language.

2 Solution

The homework was finished as expected in homework pdf file.

2.1 Creating corpus

To create big turkish text, I have used a NodeJs library with javascript. Check the following picture.



```javascript
var pdfUtil = require("pdf-to-text");
const fs = require("fs-extra");

let books = [
  { path: "./books/cografya_1.pdf", option: { from: 15, to: 180 } },
  { path: "./books/cografya_2.pdf", option: { from: 15, to: 80 } },
  { path: "./books/cografya_3.pdf", option: { from: 15, to: 180 } },
  { path: "./books/cografya_4.pdf", option: { from: 15, to: 160 } },
  { path: "./books/cografya_5.pdf", option: { from: 15, to: 225 } },
  { path: "./books/cografya_6.pdf", option: { from: 15, to: 170 } },
  { path: "./books/cografya_7.pdf", option: { from: 15, to: 115 } },
  { path: "./books/biyoloji_1.pdf", option: { from: 15, to: 75 } },
  { path: "./books/biyoloji_2.pdf", option: { from: 15, to: 60 } },
  { path: "./books/biyoloji_3.pdf", option: { from: 15, to: 70 } },
  { path: "./books/biyoloji_4.pdf", option: { from: 15, to: 80 } },
  { path: "./books/biyoloji_5.pdf", option: { from: 15, to: 160 } },
  { path: "./books/biyoloji_6.pdf", option: { from: 15, to: 115 } },
  { path: "./books/biyoloji_8.pdf", option: { from: 15, to: 90 } },
  { path: "./books/tarih_2.pdf", option: { from: 15, to: 110 } },
  { path: "./books/tarih_3.pdf", option: { from: 15, to: 110 } },
  { path: "./books/tarih_4.pdf", option: { from: 15, to: 75 } },
  { path: "./books/tarih_5.pdf", option: { from: 15, to: 225 } },
  { path: "./books/inkilap_tarihi_1.pdf", option: { from: 15, to: 255 } },
  { path: "./books/edebiyat_5.pdf", option: { from: 15, to: 110 } },
  { path: "./books/edebiyat_6.pdf", option: { from: 15, to: 80 } },
  { path: "./books/edebiyat_7.pdf", option: { from: 15, to: 110 } },
  { path: "./books/edebiyat_8.pdf", option: { from: 15, to: 155 } },
];

let sum = 0;
let i = 0;
for (const book of books) {
  sum = sum + (book.option.to - book.option.from);
  i++;
}
console.log("total number of book: " + i);
console.log("total number of page: " + sum);
```

```
akif@DESKTOP-SHGR6TR MINGW64 /d/github/cse484-nlp/HW1/
$ node createCorpus.js
total number of book: 23
total number of page: 2645

akif@DESKTOP-SHGR6TR MINGW64 /d/github/cse484-nlp/HW1/
$ |
```

Figure 1: Program to create text file(corpus)

You can see the books and their used page numbers from picture. I have 2645 page text with 23 book.

Following code piece is showing how I convert pdf to text. Note that I converted pdfs to text by **removing all new line feeds** such as carriage return, line feed, tabs etc. Also I converted all words to **lower case** in order to make proper compare with syllable based text.

2.1.1 Convert pdfs to one line text file

```
//convert pdf to text
let i = 0;
for (const book of books) {
  pdfUtil.pdfToText(book.path, book.option, function (err, data) {
    //console.log(book.path);
    if (err) {
      console.log(book.path);
      throw err;
    }
    try {
      const found = data.replace(/\s+/g, " ").replaceAll("..", "");
      const foundtr = found.toLocaleLowerCase("tr");
      //const foundtr2 = foundtr.replace(re,"");
      fs.appendFile("sample36.txt", foundtr, (err) => {
        if (err) throw err;
        //console.log(book.path + "\n");
        console.log(i);
        i++;
      });
    } catch (error) {
      console.error(error);
    }
  });
}
```

Figure 2: Convert pdfs to text file

```
1. ünite canlılar dünyası 9 insanlar, tüm insanlık tarihi boyunca, doğada yaşayan canlıları merak etmiş,
```

Figure 3: Converted text file

2.1.2 Testing created text vectors

Following pictures contains some examples of distance and anology experiment on created text corpus with word2vec C code.

```
Enter word or sentence (EXIT to break): canlılar

Word: canlılar  Position in vocabulary: 314

                                          Word       Cosine distance
----------------------------------------------------------------------
                                    ekosistemde              0.783176
                                      ekosistem              0.766303
                                      canlılar,              0.751963
                                ayrıştırıcı           0.745233
                                         ototrof              0.739848
                                     canlıların               0.734736
                            ayrıştırıcılar           0.724327
                                     zincirinin               0.722676
                                    canlılardan               0.719397
                                  ekosistemlerde              0.717956
                                     bakteriler,              0.709912
                                        arkeler,              0.702818
                                         parazit              0.702738
                                      mantarlar,              0.694671
                                            1.4.              0.692695
                                   ekosistemdeki              0.692065
                                       ekolojisi              0.691333
                                      bakteriler              0.690876
                                     soylarını                0.690689
                                     üreticiler               0.688150
```

Figure 4: Distance test with canlılar word

```
Enter three words (EXIT to break): ilkbahar mevsim sonbahar

Word: ilkbahar  Position in vocabulary: 3155

Word: mevsim  Position in vocabulary: 2388

Word: sonbahar  Position in vocabulary: 7452

                                          Word              Distance
----------------------------------------------------------------------
                                         zebra,              0.749609
                                      ağaçlar                0.735614
                                       soğuğa                0.733082
                                      mevsimsel              0.730056
                                       yazları               0.724803
                                     sonbaharda              0.721980
                                     kışları         0.717123
                                       zürafa,               0.716582
                                       antilop,              0.712416
                                     ilkbaharda              0.707555
                                          fazla,             0.701030
                                       kışlar                0.699683
                                   yağışlı,          0.696767
                                          geyik,             0.695553
                                    yağışlı          0.695373
                                    mevsimlerde              0.693546
                                       ufalanma              0.687226
                                          yazlar             0.684551
```

Figure 5: Anology test with ilkbahar mevsim sonbahar words

2.2 Dividing Turkish words into syllables

In order to divide turkish words into syllables, I have used following open source python library.

`https://github.com/MeteHanC/turkishnlp`

Code to divide syllables is following;

```python
from turkishnlp import detector

obj = detector.TurkishNLP()

f = open('sample36.txt','r',encoding = 'utf-8')
file = open('heceler.txt','w',encoding = 'utf-8')

line = f.read()
arr = obj.syllabicate_sentence(line)
for element in arr:
    for innerelement in element:
        file.write(innerelement)
        file.write(" ")
file.close()
f.close()
```

Figure 6: Simple code to divide words into syllables

```
ü ni te can lı lar dün ya sı in san lar, tüm in san lık ta ri hi bo yun ca, do
```

Figure 7: Syllables of words after dividing

2.3 Word similarity accuracy

In order to find similarity between two vector we will use cosine similarity formula.

$$\text{cosine}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \, \|\mathbf{b}\|}$$

Figure 8: cosine similarity formula

```python
def __cosine_similarity(self, vect1, vect2):
    xx, xy, yy = 0, 0, 0
    for i in range(len(vect1)):
        x = vect1[i]
        y = vect2[i]
        xx += x * x
        yy += y * y
        xy += x * y
    return xy / math.sqrt(xx * yy)
```

Figure 9: Implementation of cosine similarity formula

2.4 Measuring the accuracy

In order to measure accuracy of algorithm, we will use following formula.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

Figure 10: Mean Absolute Percentage Error formula

As seen above,we initially calculate the absolute difference between the Actual Value(Human Score)(A) and the Experiment(cosine similarity of two word)(F) value.

```python
acc = float(guess)
self.__sum2 += (abs(result-acc) / acc) * 100
accuracy = 100 - (self.__sum2 / len(self.__wordList1))
```

Figure 11: Simple implementation of MAPE formula

2.5 Syntactic distance accuracy

Here we have 20 turkish Syntactically similar words and their human scores.

```
büyüklüğünü büyüdüğü 0.50
değişiminin değişmeye 0.55
gelişme gelişmelerden 0.60
tarih tarihsel 0.90
coğrafya coğrafi 0.90
yazılmış yazılan 0.70
olmaktan olmaksızın 0.60
göz gözetmeden 0.40
aile ailesinin 0.60
canlı canlıların 0.85
oluşur oluşturulması 0.60
bölünmeler bölünmüş 0.60
gitmek gidildikçe 0.30
hava havasının 0.85
olur oluşmuş 0.60
savaş savaşı'nda 0.70
göç göçlerinin 0.50
söz sözleşme 0.55
yaşam yaşamsal 0.90
bölge bölgesel 0.90
```

Figure 12: Syntactically similar words and their human scores(0-1)

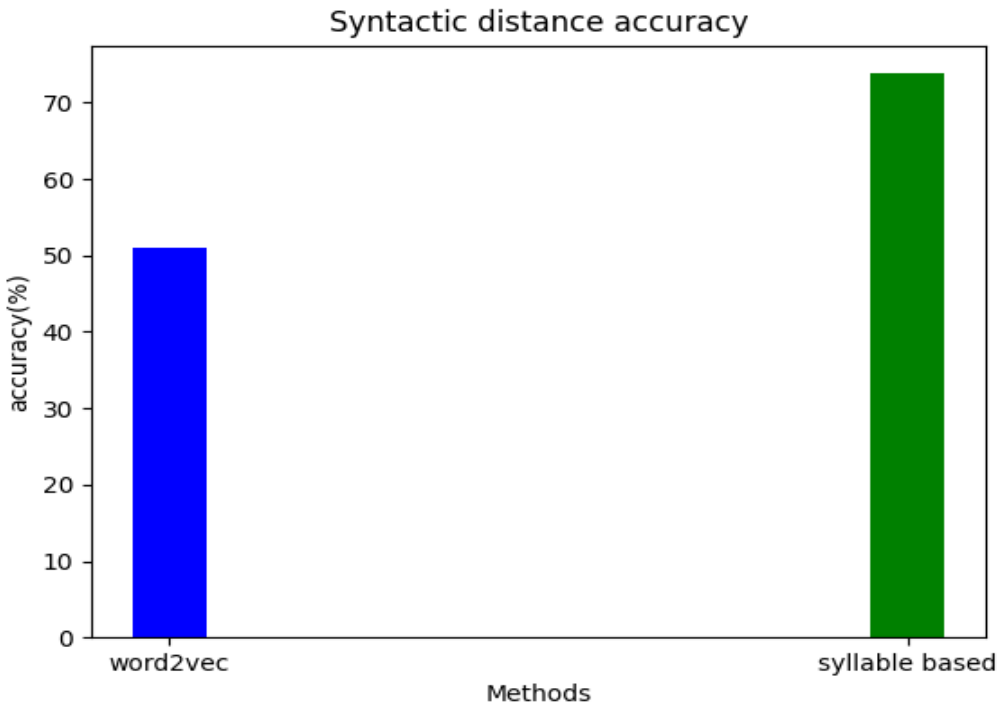


Figure 13: Syntactic distance accuracy of algorithms with above words

As you can see from graph **syllable based** algorithm is much more successful with in syntactic distance accuracy.

Semantic distance accuracy

Here we have 15 turkish semantically similar word and their human scores.

```
türkiye cumhuriyeti 0.70
yazı metin 0.50
tuz mineral 0.50
göl nehir 0.60
fetih osmanlı 0.60
bölge sınır 0.50
gece gündüz 0.70
ağaç soyağacı 0.10
atatürk inkılap 0.70
üreme canlı 0.70
kuzey yön 0.50
tarih savaş 0.50
şiir edebiyat 0.50
kitap yazar 0.50
hava oksijen 0.60
```

Figure 14: Semantically similar word and their human scores(0-1)

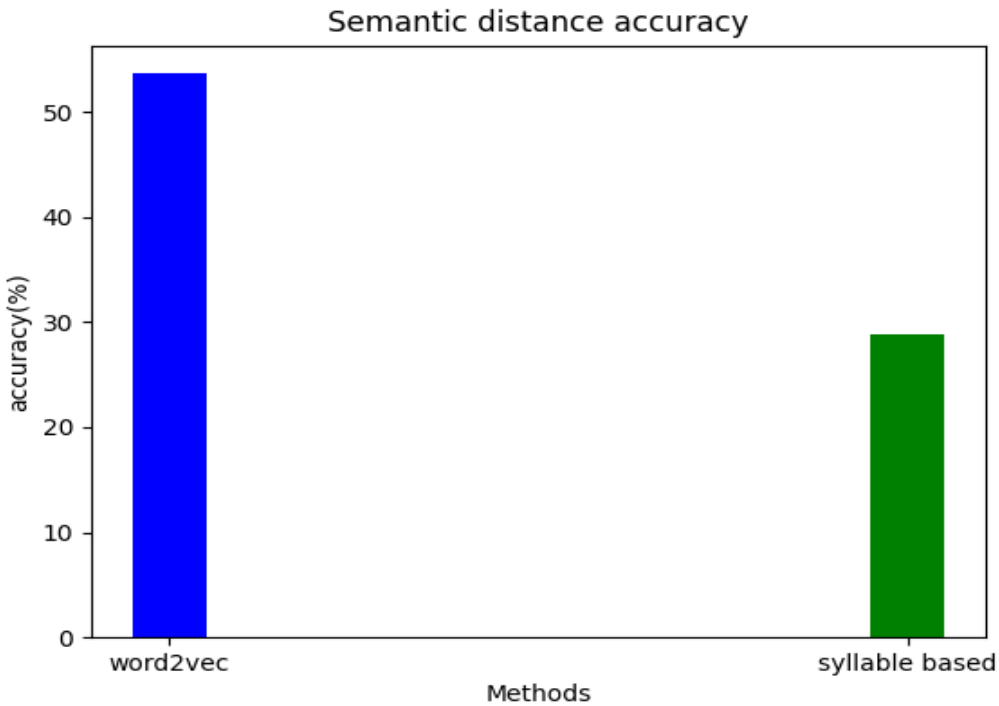


Figure 15: Semantic distance accuracy of algorithms with above words

As you can see from graph **word2vect** algorithm is much more successful with in semantic distance accuracy.

Full results

```
Syntactic distance accuracy
-----------------------------
word2vec: 50.97699953894986
syllable based: 73.79902054273141
------------------------------
Semantic distance accuracy
-----------------------------
word2vec: 53.630150917623155
syllable based: 28.775421378233972
------------------------------
```

Figure 16: Distance accuracy results(%)

2.6 Word analogy accuracy

In order to measure analogy, we will use following formulas.

$$d = \underset{i}{\operatorname{argmax}} \frac{(x_b - x_a + x_c)^T x_i}{\|x_b - x_a + x_c\|}$$

Figure 17: Full formula

$$x_b - x_a + x_c = x_d.$$

Figure 18: With the vectors

2.6.1 Syntactic analogy accuracy

Here we have 5 turkish syntactically similar analogy words and their human scores.

```
canlı canlılar bitki bitkiler 0.90
görmek görülmüş yapmak yapılmış 0.90
yaşamak yaşar almak alır 0.90
olur olmaz eder etmez. 0.80
etmek etmiş çalışmak çalışmış 0.85
```

Figure 19: Syntactically similar analogy words and their human scores(0-1)

Here, in order to calculate analogy we will use cosine similarity.

For example;

similarity(vector[bitkiler], vector[canlılar] + vector[bitki] - vector[canlı])

After getting similarity ratio then we will measure accuracy with human score. You can see resuts from graph.

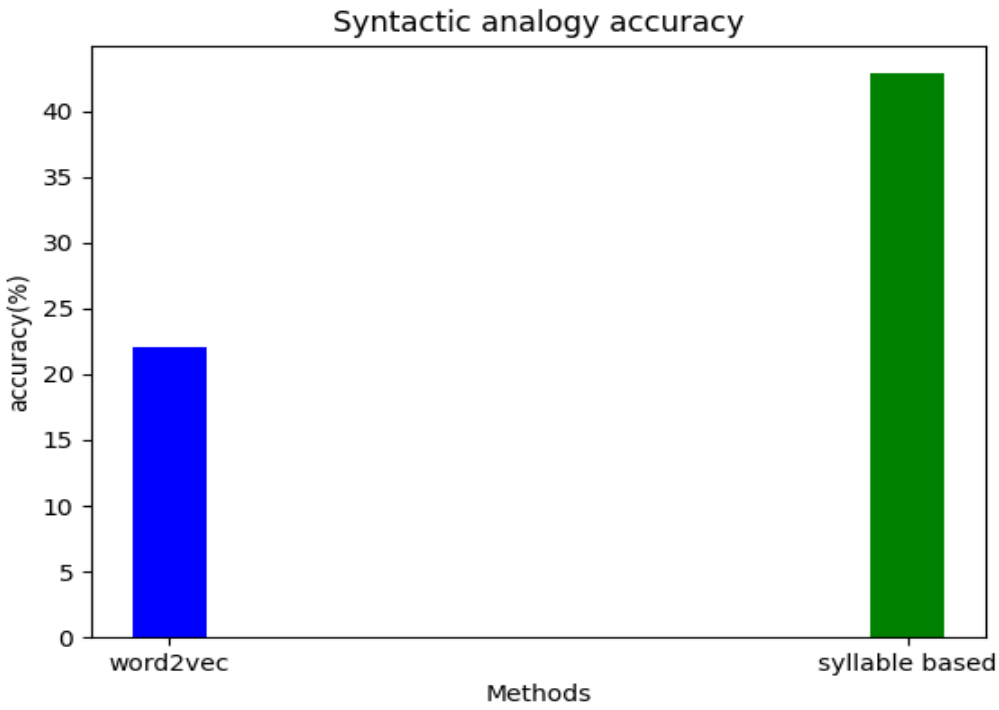


Figure 20: Syntactic analogy accuracy of algorithms with above words

As you can see from graph **syllable based** algorithm is much more successful with in syntactic analogy accuracy.

2.6.2 Semantic analogy accuracy

Here we have 5 turkish semantically similar analogy words and their human scores.

```
gece karanlık gündüz aydınlık 0.80
türkiye cumhuriyeti osmanlı devleti 0.80
kış soğuk yaz sıcak 0.60
erkek dişi baba anne 0.80
mantar bitki kuş hayvan 0.70
```

Figure 21: Semantically similar analogy words and their human scores(0-1)

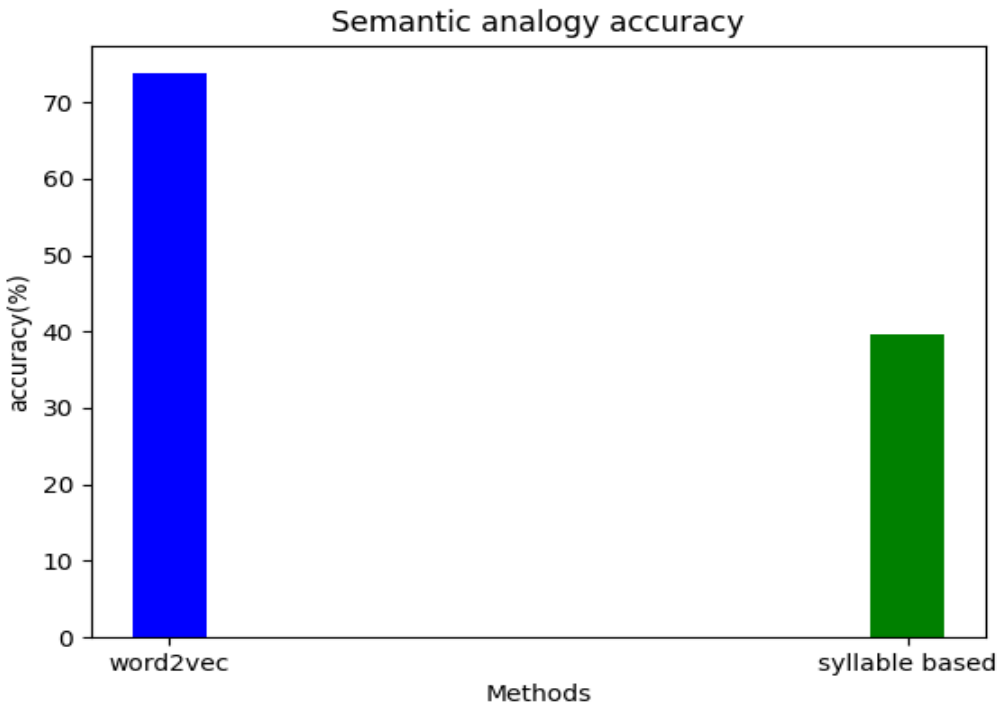


Figure 22: Semantic analogy accuracy of algorithms with above words

As you can see from graph **word2vect** algorithm is more successful with in semantic analogy accuracy.

Full results

```
Syntactic analogy accuracy
------------------------------
word2vec: 22.109520552826638
syllable based: 42.85211752771954
--------------------------------
Semantic analogy accuracy
------------------------------
word2vec: 73.73361117241532
syllable based: 39.614806756326104
--------------------------------
```

Figure 23: Analogy accuracy results(%)

How to ensure my algorithm is working

My distance and analogy algorithms works as perfect as word2vec C code. You can see results from image with same input files.

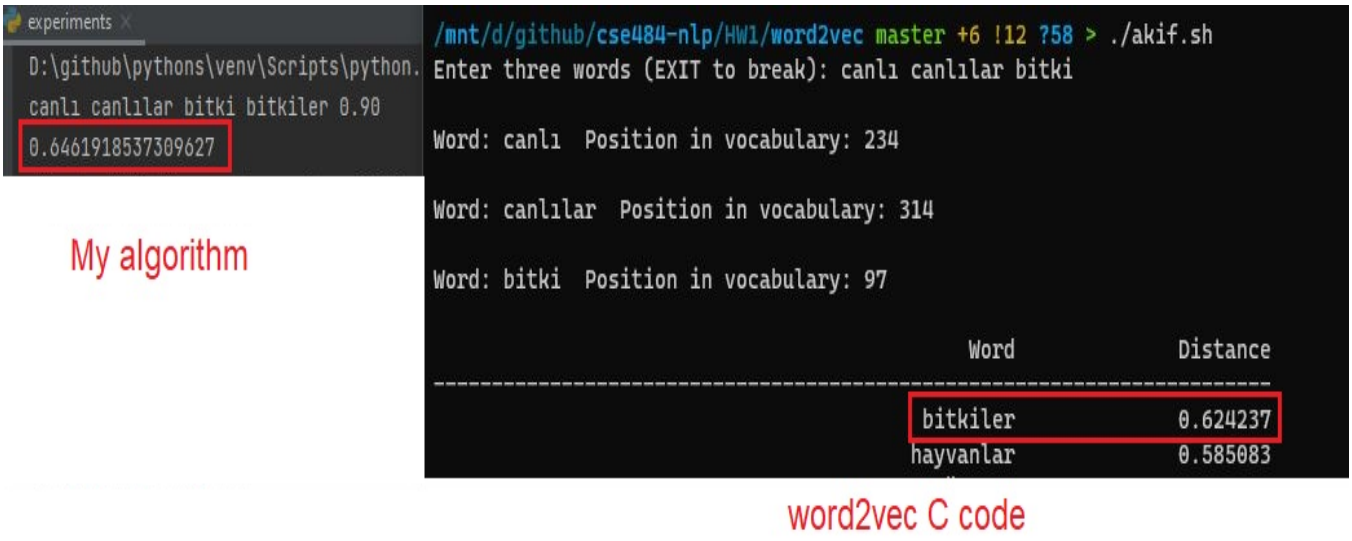


Figure 24: Compare with word2vec C code with same input files

3 References

- cs224n-2020-lecture02-wordvecs2.pdf file from lecture notes
- https://cs224d.stanford.edu/lecture_notes/notes2.pdf
- https://github.com/MeteHanC/turkishnlp
- https://www.npmjs.com/package/pdf-to-text
- http://aok.meb.gov.tr/kitap/
- https://github.com/tmikolov/word2vec
- https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/

4 How to run

Just install needed python libraries and run 171044098.py file. If you want, you can change test input words, but be sure word is present in vector file.