CSE654

Introduction to Natural Language Processing

Fall 2022

Homework - III

Çağla Şahin

Instructor: Yusuf Sinan Akgül

30 December 2022

In this homework, I am supposed to assign vectors for each N-gram of Turkish syllables and measure how well it captures the Turkish morphological derivations.

First three step were same with Homework-2. I passed that steps easily, but with a difference, I saved the list of syllables to file on previous homework, read it here into list variable to gain time on execution.

After that step, I imported packages needed in this project.

1. Package installation

In [152]:
```
import json
import sys
from gensim.models import Word2Vec
from sklearn.manifold import TSNE
import seaborn as sns
sns.set_style("darkgrid")

from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Json package is needed for read-write file features.

Gensim library is used for training model to get word2vec vector values for each syllables.

Sklearn library, specifically TSNE, seaborn and matplotlib is needed for visualization.

Pandas and numpy libraries used as helper when the DataFrames and Numpy arrays are needed.

2. Reading from file into list

```
In [153]: # Open the file for reading
          with open('syllables.txt', 'r') as f:
              # Read the contents of the file into a string
              s = f.read()
              # Convert the string to a list
              syllables = json.loads(s)
```

```
In [154]: syllables[:10]
Out[154]: ['cen', 'giz', ' ', 'han', '\n', 'cen', 'giz', ' ', 'han', ' ']
```

`syllables.txt` file includes the list of syllables as seen below.

3. Vectorization of the syllables in unigram

```
In [159]: tokenized = [syllables]
```

```
In [160]: model = Word2Vec(tokenized, vector_size=100, window=5, min_count=1, workers=4, sg=0)
```

```
In [161]: model.build_vocab(tokenized, progress_per=1000)
```

```
In [162]: model.corpus_count, model.epochs
Out[162]: (1, 5)
```

```
In [165]: model.train(tokenized, total_examples=model.corpus_count, epochs=model.epochs)
Out[165]: (50000, 7637920)
```

```
In [166]: model.save("./word2vec_nlp_hw3.model")
```

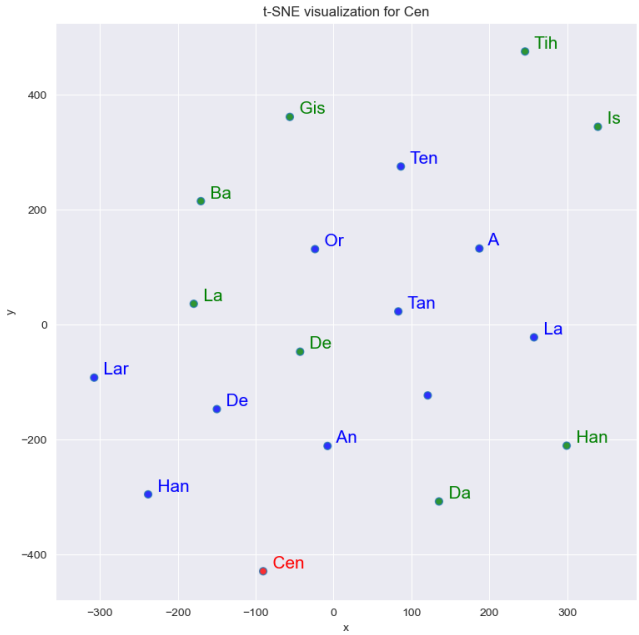
Word2Vec function belongs to Gensim library is used while creating model. Trained with at vector size 100. sg=0 means, training algorithm is CBOW.

To see the predictions of model, let's see some experiment results.

```
In [167]: # Experiments

In [168]: model.wv.similarity("cen", "giz")
Out[168]: 0.9989365

In [169]: model.wv.similarity("cen", "da")
Out[169]: 0.99791956

In [170]: model.wv.similarity("cen", "la")
Out[170]: 0.9973073

In [171]: model.wv.similarity("is", "tih")
Out[171]: 0.994846

In [172]: model.wv.similarity("cen", "is")
Out[172]: 0.99625134

In [173]: model.wv.doesnt_match(['te', 'mu', 'cen', 'de'])
Out[173]: 'mu'

In [174]: model.wv.similar_by_word('mu')
Out[174]: [('cin', 0.9751501679420471),
           ('in', 0.9512126445770264),
           ('bor', 0.9482771754264832),
           ('is', 0.9475966691970825),
           ('te', 0.9464403390884399),
           ('', 0.9421329498291016),
           ('mek', 0.941533625125885),
           ('mel', 0.941185712814331),
           ('best', 0.9401475191116333),
           ('et', 0.9393740892410278)]
```

4. Visualization for syllable "cen"

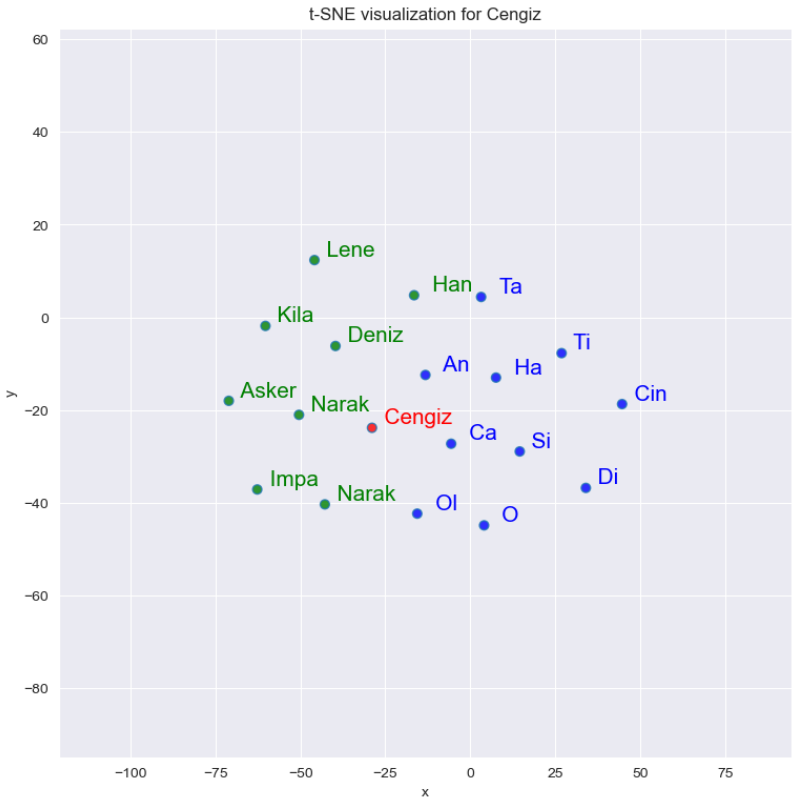
>>tsnescatterplot(model, 'cen', ['gis', 'han', 'da', 'la', 'de', 'is', 'tih', 'ba'])



Vectorization, model training and visualization processed in similar ways for bigram and threegram, so I am not going to every code piece I wrote during process here. You may reach them through jupyter notebook on same directory.

5. Bigram Visualization

>>tsnescatterplot(model2, 'cengiz', ['asker', 'deniz', 'narak', ' han', 'kila', 'lene', 'narak', 'impa'])




```
In [175]: model2.wv.most_similar("cengiz")
Out[175]: [(' ta', 0.9957867860794067),
           ('si ', 0.9956386685371399),
           (' ha', 0.9956256151199341),
           ('di ', 0.9956230521202087),
           ('ti ', 0.9954684972763062),
           ('cin ', 0.9954051375389099),
           (' o', 0.9953967332839966),
           (' ol', 0.9953184127807617),
           (' ca', 0.9952512979507446),
           (' an', 0.9952484369277954)]
```

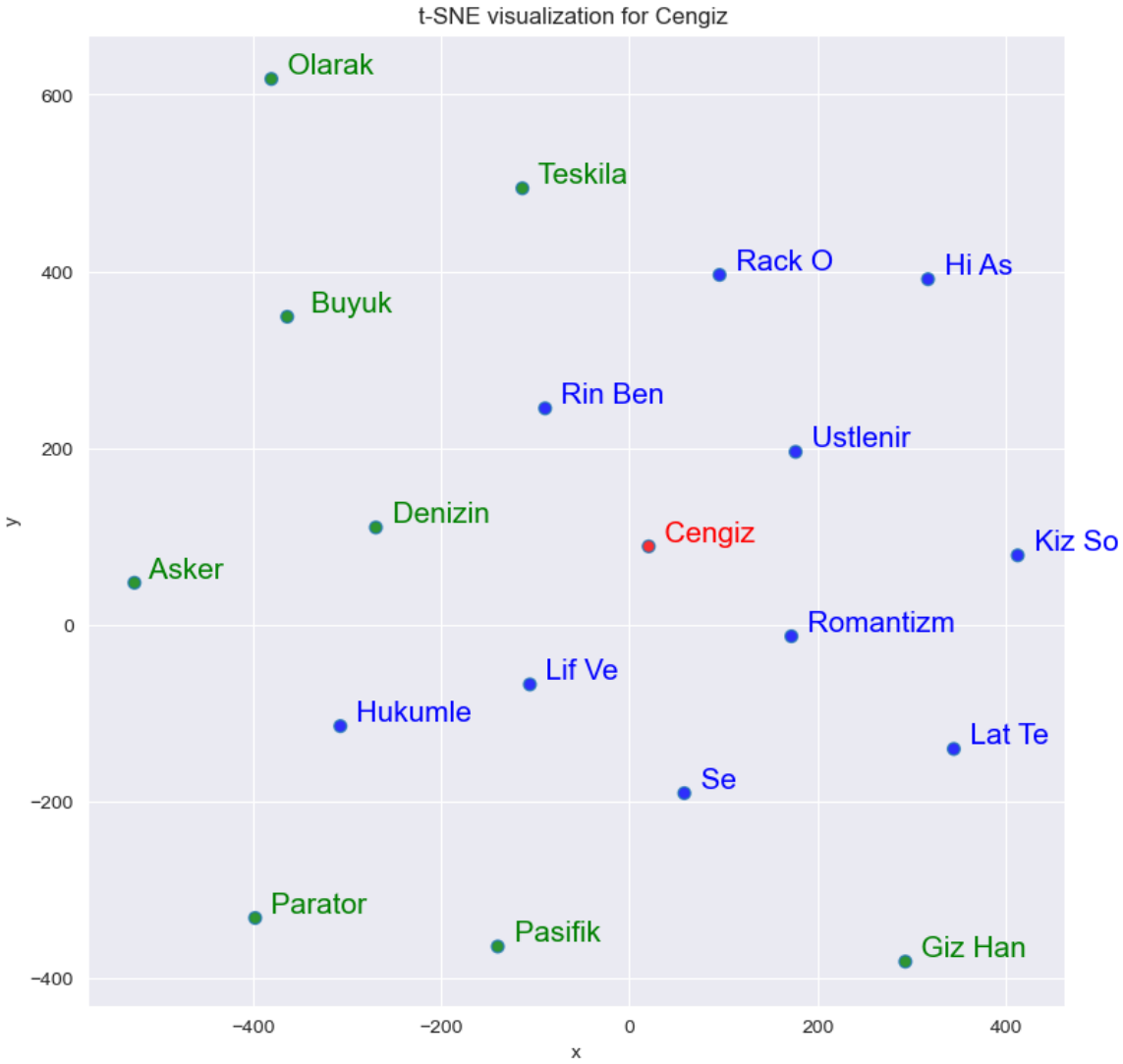
```
In [176]: model2.wv.most_similar("deniz")
Out[176]: [(' de', 0.9228073954582214),
           ('mogol', 0.9209035634994507),
           (' san', 0.9200536608695984),
           (' han', 0.9200060367584229),
           (' o', 0.9198488593101501),
           ('da ', 0.9194271564483643),
           ('cin ', 0.9192154407501221),
           (' te', 0.9191370606422424),
           ('mucin', 0.9189898967742920),
           ('bir ', 0.9182526469230652)]
```

```
In [177]: model2.wv.most_similar("kolu")
Out[177]: [(' le', 0.647417426109314),
           ('adiy', 0.6349949240684509),
           ('fakat', 0.6343328356742859),
           ('sefer', 0.6250205039978027),
           ('mensup', 0.6248916983604431),
           ('sure', 0.624230146408081),
           ('linde', 0.6229454278945923),
           (' yer', 0.6228820085525513),
           ('kazan', 0.6198706030845642),
           (' ti', 0.6182495951652527)]
```

```
In [178]: model2.wv.most_similar("larda")
Out[178]: [(' mo', 0.9184617400169373),
           ('mak ', 0.9161664247512817),
           (' ko', 0.9150940775871277),
           ('i ', 0.9141795635223389),
           (' ken', 0.914069652557373),
           ('ler ', 0.9140181541442871),
           ('ca ', 0.9138591289520264),
           (' di', 0.913798451423645),
           ('bir ', 0.9135225415229797),
           ('na ', 0.9135101437568665)]
```

6. Threegram Visualization

>>tsnescatterplot(model3, 'cengiz ', ['giz han', 'asker ','olarak', ' buyuk', 'parator', 'teskila', 'denizin', 'pasifik'])




```
In [136]: model3.wv.most_similar("cengiz ")
Out[136]: [('rin ben', 0.406869113445282),
           ('kiz so', 0.3948172330856323),
           ('lat te', 0.3865315616130829),
           ('lif ve', 0.3849363923072815),
           ('hukumle', 0.3805915117263794),
           ('ustlenir', 0.377753883600235),
           ('romantizm', 0.37199556827545166),
           ('hi as', 0.3710998296737671),
           ('se  ', 0.3697696328163147),
           ('rack o', 0.3690072000026703)]
```

```
In [148]: model3.wv.most_similar("lari")
Out[148]: [('verekli', 0.4123723804950714),
           ('ve ah', 0.4005851745605469),
           ('port to', 0.39064010977745056),
           ('ti\njus', 0.37841248512268066),
           ('cun is', 0.3764881193637848),
           ('deaux', 0.37581926584243774),
           ('testinde', 0.37369439005851746),
           ('laslanan', 0.3722316324710846),
           ('roun ', 0.36243927478790283),
           ('ki rol', 0.36221837997436523)]
```

```
In [138]: model3.wv.most_similar("asker ")
Out[138]: [('fik ko', 0.44116273522377014),
           ('derati', 0.4363101124763489),
           ('nasli ', 0.43031901121139526),
           ('fes es', 0.40192973613739014),
           (' misse', 0.3909081816673279),
           ('dirgerler', 0.39034366607666016),
           ('anlayip', 0.38480859994888306),
           ('sertab ', 0.3810317814350128),
           ('l ve', 0.37797054648399353),
           ('eslesme', 0.37779340147972107)]
```

```
In [143]: model3.wv.most_similar("giz han")
Out[143]: [('yordum ', 0.42144888639450073),
           (' niyet', 0.4183834493160248),
           (' kemik', 0.40929079055786133),
           ('bir bas', 0.399245947599411),
           ('zer a', 0.3916325867176056),
           ('ruh fut', 0.38856154680252075),
           ('ger ilk', 0.3852227032184601),
           ('restlikle', 0.38172122836112976),
           (' asik', 0.3814218044281006),
           ('ke el', 0.3750901222229004)]
```