Laboratorul 5

October 29, 2021

1 Laboratorul 5

Visualize words in a 2- or 3-dimensional space using the embeddings given by GloVe (https://nlp.stanford.edu/projects/glove/) and word2vec (https://code.google.com/archive/p/word2vec/)

a) by choosing randomly the dimensions

b) by using PCA to extract 2- or 3-dimensional embedding

[1]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

from sklearn.decomposition import PCA

from gensim.models import Word2Vec, KeyedVectors

import random
import shutil
import gzip

sns.set(rc={'figure.figsize':(11.7,8.27)})
```

[2]:
```
embeddings_dict = {}
```

1.1 Glove embeddings

[3]:
```
with open("glove.6B.50d/glove.6B.50d.txt", 'r', encoding="utf-8") as f:
    for line in f:
        values = line.split()
        word = values[0]
        vector = np.asarray(values[1:], "float32")
        embeddings_dict[word] = vector
```

[4]:
```
words = 'cryogenics fluid magnifier computer sigmoid planet graphics hydrogen␣
 ↪star'.split(' ')
words
```

[4]:
```
['cryogenics',
 'fluid',
 'magnifier',
 'computer',
 'sigmoid',
 'planet',
 'graphics',
 'hydrogen',
 'star']
```

[5]:
```
embedded_words = []
for word in words:
    embedding = embeddings_dict[word]
    embedded_words.append(embedding)
embedded_words = np.array(embedded_words)
embedded_words[:2]
```

[5]:
```
array([[ 0.32958  , -0.73718  , -0.36456  ,  0.041095 , -0.54833  ,
        -0.49963  ,  0.37722  ,  0.093935 ,  0.6809   ,  0.47867  ,
         0.31596  ,  0.043084 ,  0.75995  ,  0.33483  , -0.39797  ,
         0.20393  , -0.30002  ,  0.85011  ,  0.049964 ,  0.20503  ,
         0.54068  ,  0.20317  ,  0.2346   ,  0.1452   , -0.066562 ,
         0.29062  ,  0.067025 ,  0.26032  , -0.075532 ,  0.10474  ,
        -0.88235  , -0.69557  , -0.16237  , -0.47868  ,  0.27801  ,
         0.30508  , -0.30258  ,  0.77719  ,  0.86199  , -0.17563  ,
         0.34946  , -0.4917   ,  0.14196  ,  0.81657  ,  0.34925  ,
         0.13989  ,  0.93586  ,  0.46772  , -0.044696 ,  0.18155  ],
       [ 0.91461  , -0.19968  , -0.063328 , -0.21995  , -0.42288  ,
         1.3524   ,  1.2826   ,  0.10437  ,  0.63458  ,  0.39955  ,
         1.154    ,  0.16028  ,  0.089479 ,  0.57338  , -0.47122  ,
         0.36571  , -0.1557   ,  0.65002  , -0.0071772, -1.164    ,
        -0.15359  , -0.1474   ,  1.092    , -0.37995  ,  0.10608  ,
        -0.20822  , -0.20269  ,  1.2033   ,  0.72448  ,  1.1884   ,
         2.2879   , -0.60778  ,  0.097401 , -0.64997  , -0.37436  ,
         0.7148   , -0.32962  ,  1.1009   ,  1.415    ,  0.43381  ,
         0.87576  ,  0.32935  , -0.67945  ,  1.0241   , -0.65547  ,
        -0.55301  ,  1.234    ,  0.11762  , -0.37046  , -0.24595  ]],
      dtype=float32)
```

[6]:
```
embedded_words_df = pd.DataFrame(data=embedded_words)
embedded_words_df['word'] = words
embedded_words_df
```

[6]:
```
          0         1         2         3        4         5        6  \
0  0.329580 -0.737180 -0.364560  0.041095 -0.54833 -0.499630  0.37722
1  0.914610 -0.199680 -0.063328 -0.219950 -0.42288  1.352400  1.28260
2  0.133400 -0.623840  0.214450 -0.295720  0.45839  0.159230  0.72642
3  0.079084 -0.815040  1.790100  0.916530  0.10797 -0.556280 -0.84427
4  2.241900 -0.649430 -0.915760 -0.026611 -0.14470 -0.146270  1.48320
5  1.512800  0.842560  1.023900  0.745030  0.76455 -0.341180 -0.24314
6 -0.039968 -0.048762  1.224600  1.466000 -0.17796 -1.167600 -0.10839
7  1.279300  0.908300  1.770000 -0.341240 -0.13670  1.474800  0.74260
8 -0.210250  1.608100  0.037375  1.041100  0.61061  0.064748 -0.93674

          7        8        9  …        41        42        43       44  \
0  0.093935  0.68090  0.47867  … -0.491700  0.141960  0.816570  0.34925
1  0.104370  0.63458  0.39955  …  0.329350 -0.679450  1.024100 -0.65547
2 -1.210600  0.70613  1.27430  …  0.452150 -0.325860  0.531170 -0.54348
3 -1.495100  0.13418  0.63627  …  0.055129  0.037891  1.327500  0.30991
4  0.490120 -0.72032  0.55829  …  0.126990 -0.825200  0.411900 -0.43986
5 -0.388380  0.55591  0.24943  … -0.182250  0.291150 -0.048251 -0.75363
6 -1.996700 -0.92577  0.23371  …  0.369500 -0.119940 -0.309090  0.20326
7 -0.135240 -0.46832  2.14660  … -0.698790 -0.190780 -0.647580  0.47832
8 -0.030028 -0.18348  0.73875  …  1.341200 -0.340810 -0.501830 -0.25140

        45       46        47        48       49        word
0  0.13989  0.93586  0.467720 -0.044696  0.18155  cryogenics
1 -0.55301  1.23400  0.117620 -0.370460 -0.24595       fluid
2  0.54810  0.93206 -0.050126 -0.092933  0.25020   magnifier
3  0.50697  1.23570  0.127400 -0.114340  0.20709    computer
4 -0.16950  1.18910 -0.757350  1.207900  0.48866     sigmoid
5  0.10815  0.32424 -1.109100  0.093539 -0.12124      planet
6  0.93519  1.46120  0.260330  0.172000  1.00760    graphics
7  0.14998  0.27132  0.357060  0.351510 -0.52624    hydrogen
8 -0.10199  0.19292 -0.489340 -0.417930  0.18085        star

[9 rows x 51 columns]
```

[7]:
```
def choose_dimensions(number_dimensions = 2):
    dimensions = []
    possible_dimensions = list(range(0,50))
    for dimension in range(0,number_dimensions):
        chosen_dim = random.choice(possible_dimensions)
        dimensions.append(chosen_dim)
        possible_dimensions.remove(chosen_dim)
    return dimensions
```

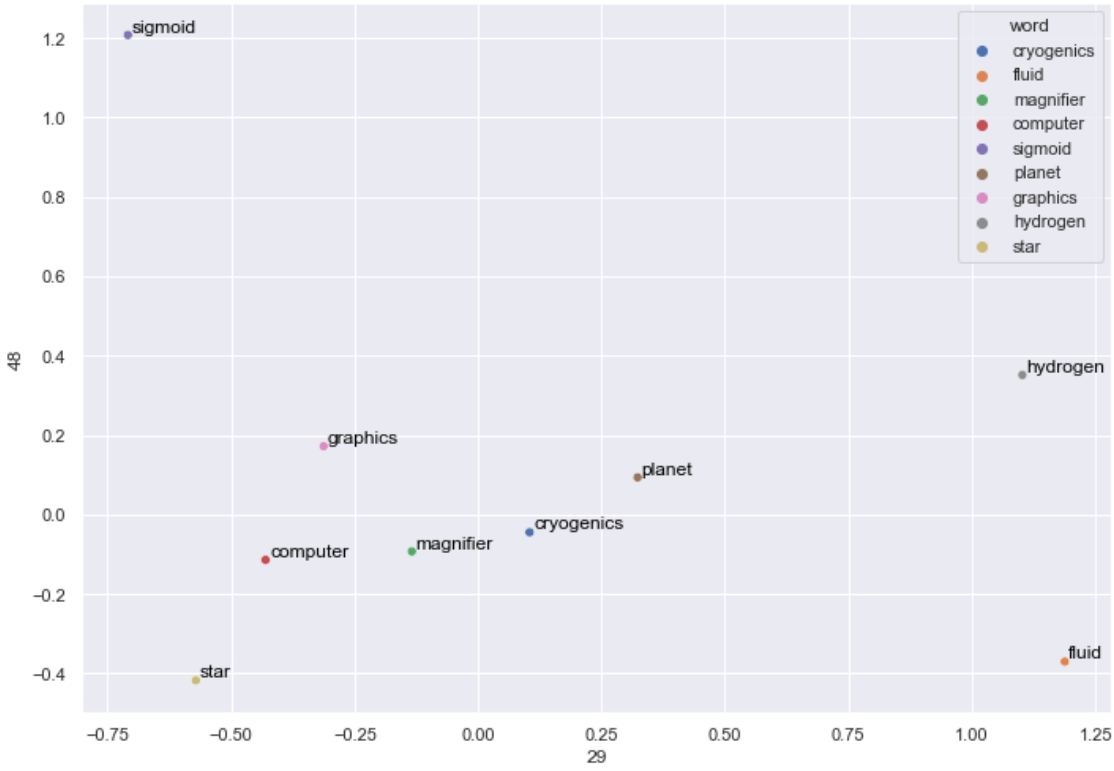
1.1.1 2 dimensions

Randomly chosen

[8]:
```
dimensions = choose_dimensions(2)
dimensions
```

[8]:
```
[29, 48]
```

[9]:
```
sns.scatterplot(data=embedded_words_df, x=dimensions[0], y=dimensions[1],␣
↪hue="word")
for i in range(embedded_words_df.shape[0]):
    plt.text(x=embedded_words[i][dimensions[0]]+0.
↪008,y=embedded_words[i][dimensions[1]]+0.008,s=words[i],
         fontdict=dict(color='black',size=12))
plt.show()
```



PCA components

[10]:
```
pca = PCA(n_components=2)
new_components = pca.fit_transform(embedded_words)
new_components
```

[10]:
```
array([[ 1.7381048 , -0.7712375 ],
       [ 1.0864409 ,  2.1533725 ],
       [ 1.488218  , -1.935567  ],
       [-3.4715166 , -0.89831424],
```

```
       [ 4.478141  , -1.9911498 ],
       [-0.8270078 ,  1.5515256 ],
       [-2.8494458 , -2.368132  ],
       [ 0.60904264,  4.126927  ],
       [-2.2519772 ,  0.13257527]], dtype=float32)
```

[11]:
```
pca_embedded_df = pd.DataFrame(data=new_components)
pca_embedded_df['word'] = words
```

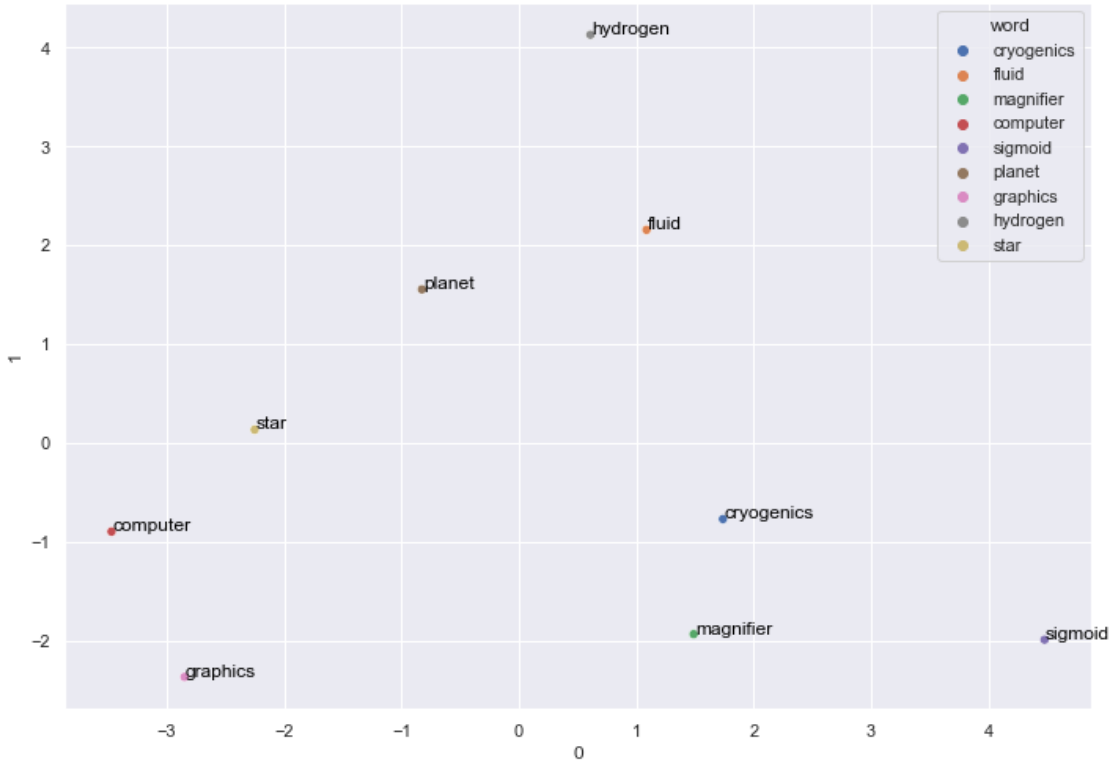
[12]:
```
pca_embedded_df
```

[12]:

	0	1	word
0	1.738105	-0.771237	cryogenics
1	1.086441	2.153373	fluid
2	1.488218	-1.935567	magnifier
3	-3.471517	-0.898314	computer
4	4.478141	-1.991150	sigmoid
5	-0.827008	1.551526	planet
6	-2.849446	-2.368132	graphics
7	0.609043	4.126927	hydrogen
8	-2.251977	0.132575	star

[13]:
```
sns.scatterplot(data=pca_embedded_df, x=0, y=1, hue="word")

for i in range(embedded_words_df.shape[0]):
     plt.text(x=new_components[i][0]+0.008,y=new_components[i][1]+0.
 ↪008,s=words[i],
          fontdict=dict(color='black',size=12))
plt.show()
```



1.1.2 3 dimensions

Randomly chosen

[14]:
```
dimensions = choose_dimensions(3)
fig = px.scatter_3d(embedded_words_df, x=dimensions[0], y=dimensions[1],␣
↪z=dimensions[2], color="word", text="word")
fig.show()
```

PCA components

[15]:
```
pca = PCA(n_components=3)
new_components = pca.fit_transform(embedded_words)
```

```
new_components
```

[15]:
```
array([[ 1.7381048 , -0.7712375 , -0.948682  ],
       [ 1.0864409 ,  2.1533725 , -1.3643448 ],
       [ 1.488218  , -1.935567  ,  0.08669309],
       [-3.4715166 , -0.89831424, -1.3450954 ],
       [ 4.478141  , -1.9911498 ,  0.86181384],
       [-0.8270078 ,  1.5515256 ,  2.2685962 ],
       [-2.8494458 , -2.368132  , -1.8082263 ],
       [ 0.60904264,  4.126927  , -1.3064274 ],
       [-2.2519772 ,  0.13257527,  3.5556743 ]], dtype=float32)
```

[16]:
```
pca_embedded_df = pd.DataFrame(data=new_components)
pca_embedded_df['word'] = words
```

[17]:
```
pca_embedded_df
```

[17]:

	0	1	2	word
0	1.738105	-0.771237	-0.948682	cryogenics
1	1.086441	2.153373	-1.364345	fluid
2	1.488218	-1.935567	0.086693	magnifier
3	-3.471517	-0.898314	-1.345095	computer
4	4.478141	-1.991150	0.861814	sigmoid
5	-0.827008	1.551526	2.268596	planet
6	-2.849446	-2.368132	-1.808226	graphics
7	0.609043	4.126927	-1.306427	hydrogen
8	-2.251977	0.132575	3.555674	star

[18]:
```
fig = px.scatter_3d(pca_embedded_df, x=0, y=1, z=2, color="word", text="word")
fig.show()
```

1.2 Google word2vec embeddings

[19]:
```
with gzip.open('GoogleNews-vectors-negative300.bin.gz', 'rb') as f_in:
    with open('GoogleNews-vectors-negative300.txt', 'wb') as f_out:
        shutil.copyfileobj(f_in, f_out)
```

[20]:
```
w2v_model = KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.
↪txt', binary=True)
```

[21]:
```
print("Number of words in vocablulary: ",len(list(w2v_model.index_to_key)))
```

```
Number of words in vocablulary:  3000000
```

[22]:
```
embedded_words = []
for word in words:
    embedding = w2v_model[word]
    embedded_words.append(embedding)
embedded_words = np.array(embedded_words)
```

[23]:
```
embedded_words_df = pd.DataFrame(data=embedded_words)
embedded_words_df['word'] = words
embedded_words_df
```

[23]:

	0	1	2	3	4	5	6	7	8	9	…	291	292	293	294	295	296	297	298	299	word
0	-0.316406	0.199219	0.328125	-0.092773	-0.255859	0.523438	0.271484	-0.241211	-0.119629	-0.263672	…	0.058350	-0.408203	0.208008	0.073730	-0.230469	-0.021118	0.047363	-0.201172	0.104492	cryogenics
1	0.073242	0.190430	-0.047363	-0.255859	-0.500000	0.000957	0.056885	-0.164062	-0.079102	-0.199219	…	0.002502	-0.034180	-0.267578	-0.192383	-0.006348	0.092285	0.194336	-0.083008	-0.135742	fluid
2	0.120117	-0.117188	-0.363281	-0.068848	-0.247070	0.406250	0.198242	0.269531	0.025391	-0.082031	…	0.122070	0.039551	-0.083008	-0.014832	-0.200195	0.180664	-0.093750	0.147461	-0.035400	magnifier
3	0.107422	-0.201172	0.123047	0.211914	-0.091309	0.216797	-0.131836	0.083008	0.202148	0.047852	…	0.170898	0.056641	-0.104492	0.138672	-0.157227	0.003235	-0.048096	-0.248047	-0.062012	computer
4	-0.189453	0.051758	0.078613	0.093262	-0.281250	0.030396	0.071289	-0.146484	0.145508	-0.070801	…	0.081055	-0.057861	0.171875	0.021362	0.069336	-0.096191	0.066406	0.162109	-0.011597	sigmoid
5	-0.015503	0.207031	0.185547	0.115234	-0.257812	0.196289	0.300781	-0.208984	0.053955	-0.013550	…	0.237305	-0.345703	0.143555	0.016724	-0.041992	-0.075684	0.016235	0.275391	0.005280	planet
6	0.236328	-0.139648	0.079590	0.160156	-0.084961	0.118652	0.207031	-0.064453	-0.026489	0.056885	…	0.007202	-0.035156	0.050049	0.149414	-0.039795	0.038574	-0.361328	-0.006866	-0.166992	graphics
7	-0.127930	0.000904	0.065918	-0.292969	-0.330078	-0.373047	0.123535	-0.394531	0.014648	0.077148	…	-0.027710	-0.375000	-0.013855	-0.151367	-0.030518	-0.376953	-0.121582	0.036865	-0.137695	hydrogen
8	0.164062	0.188477	0.141602	-0.029419	0.020874	0.137695	-0.016846	-0.326172	0.075195	-0.052002	…	0.072754	0.008240	-0.126953	-0.003113						

```
8 -0.231445 -0.167969 -0.071289 -0.102051  0.014465        star

[9 rows x 301 columns]
```

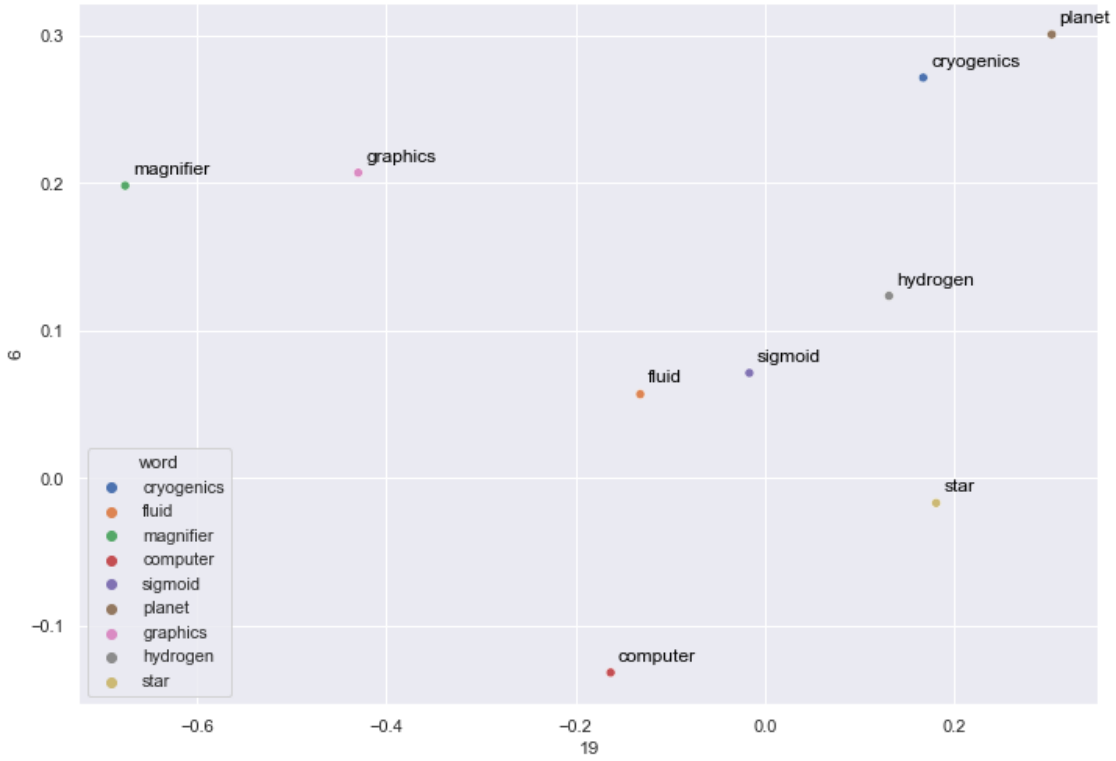
1.2.1 2 dimensions

Randomly chosen

[24]:
```
dimensions = choose_dimensions(2)
dimensions
```

[24]:
```
[19, 6]
```

[25]:
```
sns.scatterplot(data=embedded_words_df, x=dimensions[0], y=dimensions[1],␣
↪hue="word")
for i in range(embedded_words_df.shape[0]):
    plt.text(x=embedded_words[i][dimensions[0]]+0.
↪008,y=embedded_words[i][dimensions[1]]+0.008,s=words[i],
         fontdict=dict(color='black',size=12))
plt.show()
```



PCA components

[26]:
```
pca = PCA(n_components=2)
new_components = pca.fit_transform(embedded_words)
new_components
```

[26]:
```
array([[ 1.2745526 ,  1.184909  ],
       [ 0.35304463, -0.67453027],
       [-2.1682465 ,  2.0517476 ],
       [-0.99068975, -0.14618105],
       [-0.0029891 , -0.46935183],
       [ 0.48165616, -1.347399  ],
       [-1.4469341 , -0.45695087],
       [ 2.6041012 ,  1.1088123 ],
       [-0.10449445, -1.251055  ]], dtype=float32)
```

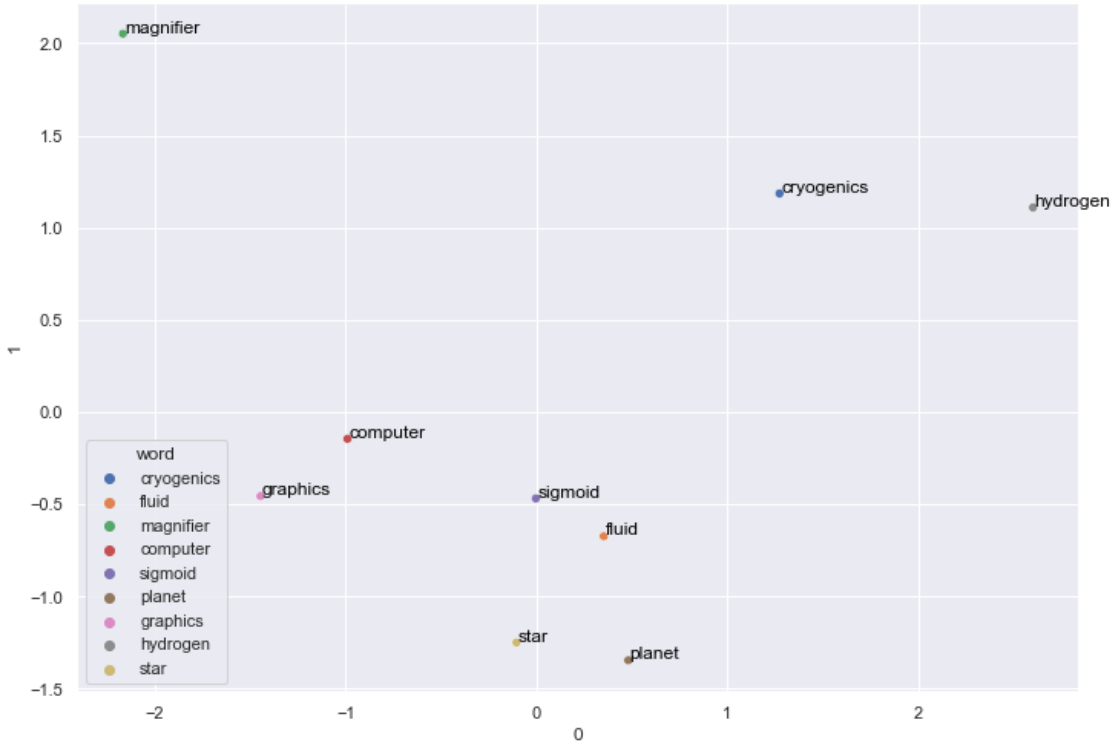
[27]:
```
pca_embedded_df = pd.DataFrame(data=new_components)
pca_embedded_df['word'] = words
```

[28]:
```
pca_embedded_df
```

[28]:

	0	1	word
0	1.274553	1.184909	cryogenics
1	0.353045	-0.674530	fluid
2	-2.168247	2.051748	magnifier
3	-0.990690	-0.146181	computer
4	-0.002989	-0.469352	sigmoid
5	0.481656	-1.347399	planet
6	-1.446934	-0.456951	graphics
7	2.604101	1.108812	hydrogen
8	-0.104494	-1.251055	star

[29]:
```
sns.scatterplot(data=pca_embedded_df, x=0, y=1, hue="word")
for i in range(embedded_words_df.shape[0]):
    plt.text(x=new_components[i][0]+0.008,y=new_components[i][1]+0.
 ↪008,s=words[i],
             fontdict=dict(color='black',size=12))
plt.show()
```



1.2.2 3 dimensions

Randomly chosen

[30]:
```
dimensions = choose_dimensions(3)
fig = px.scatter_3d(embedded_words_df, x=dimensions[0], y=dimensions[1],␣
↪z=dimensions[2], color="word", text="word")
fig.show()
```

PCA components

[31]:
```
pca = PCA(n_components=3)
new_components = pca.fit_transform(embedded_words)
new_components
```

[31]:
```
array([[ 1.2745526 ,  1.184909  ,  2.2507687 ],
       [ 0.35304463, -0.67453027, -0.46162206],
       [-2.1682465 ,  2.0517476 , -1.1953447 ],
       [-0.99068975, -0.14618105,  0.8968929 ],
       [-0.0029891 , -0.46935183, -0.59046   ],
       [ 0.48165616, -1.347399  , -0.29771113],
       [-1.4469341 , -0.45695087,  0.9910242 ],
       [ 2.6041012 ,  1.1088123 , -1.1075666 ],
       [-0.10449445, -1.251055  , -0.4859815 ]], dtype=float32)
```

[32]:
```
pca_embedded_df = pd.DataFrame(data=new_components)
pca_embedded_df['word'] = words
```

[33]:
```
pca_embedded_df
```

[33]:

	0	1	2	word
0	1.274553	1.184909	2.250769	cryogenics
1	0.353045	-0.674530	-0.461622	fluid
2	-2.168247	2.051748	-1.195345	magnifier
3	-0.990690	-0.146181	0.896893	computer
4	-0.002989	-0.469352	-0.590460	sigmoid
5	0.481656	-1.347399	-0.297711	planet
6	-1.446934	-0.456951	0.991024	graphics
7	2.604101	1.108812	-1.107567	hydrogen
8	-0.104494	-1.251055	-0.485981	star

[34]:
```
fig = px.scatter_3d(pca_embedded_df, x=0, y=1, z=2, color="word", text="word")
fig.show()
```

[ ]:
```
```