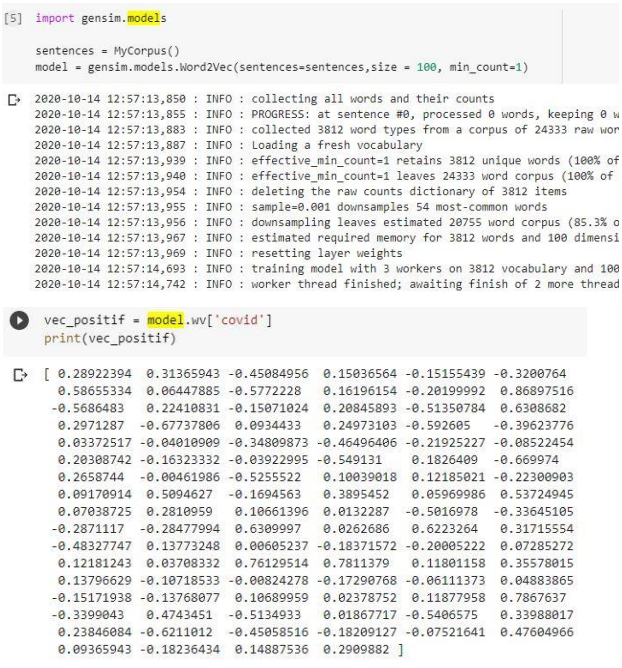
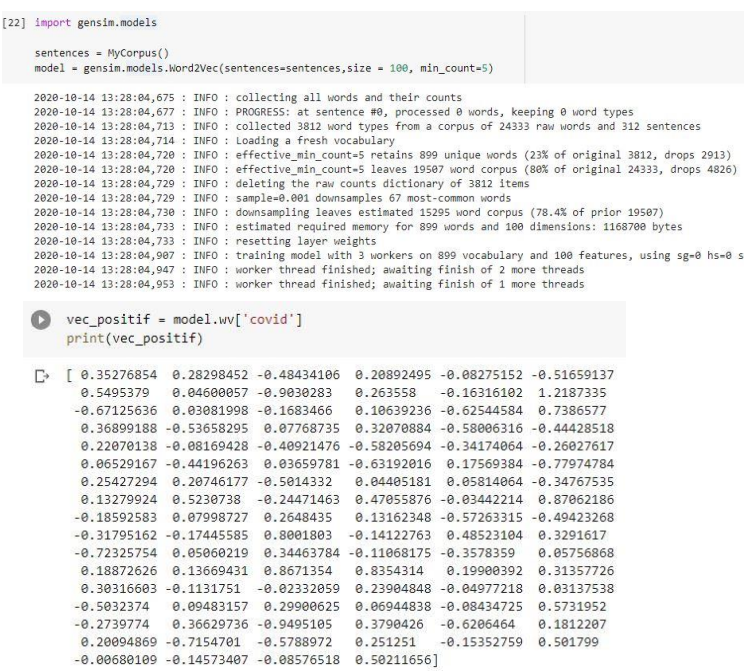
Laporan Analisis Word2vec Skip-Gram menggunakan library Gensim

1. min_count digunakan untuk mengabaikan semua kata dengan frekuensi total lebih rendah dari apa yang kita setting. Setting yang digunakan pada experiment kali ini adalah min_count = 1 dan min_count=5. Jadi apabila kata kata yang jarang muncul maka akan dihapus sebelum pelatihan dilakukan.



```
[5] import gensim.models

    sentences = MyCorpus()
    model = gensim.models.Word2Vec(sentences=sentences,size = 100, min_count=1)

2020-10-14 12:57:13,850 : INFO : collecting all words and their counts
2020-10-14 12:57:13,855 : INFO : PROGRESS: at sentence #0, processed 0 words, keeping 0 w
2020-10-14 12:57:13,883 : INFO : collected 3812 word types from a corpus of 24333 raw wor
2020-10-14 12:57:13,887 : INFO : Loading a fresh vocabulary
2020-10-14 12:57:13,939 : INFO : effective_min_count=1 retains 3812 unique words (100% of
2020-10-14 12:57:13,940 : INFO : effective_min_count=1 leaves 24333 word corpus (100% of
2020-10-14 12:57:13,954 : INFO : deleting the raw counts dictionary of 3812 items
2020-10-14 12:57:13,955 : INFO : sample=0.001 downsamples 54 most-common words
2020-10-14 12:57:13,956 : INFO : downsampling leaves estimated 20755 word corpus (85.3% o
2020-10-14 12:57:13,967 : INFO : estimated required memory for 3812 words and 100 dimensi
2020-10-14 12:57:13,969 : INFO : resetting layer weights
2020-10-14 12:57:14,693 : INFO : training model with 3 workers on 3812 vocabulary and 100
2020-10-14 12:57:14,742 : INFO : worker thread finished; awaiting finish of 2 more thread

vec_positif = model.wv['covid']
print(vec_positif)

[ 0.28922394  0.31365943 -0.45084956  0.15036564 -0.15155439 -0.3200764
  0.58655334  0.06447885 -0.5772228   0.16196154 -0.20199992  0.86897516
 -0.5686483   0.22410831 -0.15071024  0.20845893 -0.51350784  0.6308682
  0.2971287  -0.67737806  0.0934433   0.24973103 -0.592605   -0.39623776
  0.03372517 -0.04010909 -0.34809873 -0.46496406 -0.21925227 -0.08522454
  0.20308742 -0.16323332 -0.03922995 -0.549131    0.1826409  -0.669974
  0.2658744  -0.00461986 -0.5255522   0.10039018  0.12185021 -0.22300903
  0.09170914  0.5094627  -0.1694563   0.3895452   0.05969986  0.53724945
  0.07038725  0.2810959   0.10661396  0.0132287  -0.5016978  -0.33645105
 -0.2871117  -0.28477994  0.6309997   0.0262686   0.6223264   0.31715554
 -0.48327747  0.13773248  0.00605237 -0.18371572 -0.20005222  0.07285272
  0.12181243  0.03708332  0.76129514  0.7811379   0.11801158  0.35578015
  0.13796629 -0.10718533 -0.00824278 -0.17290768 -0.06111373  0.04883865
 -0.15171938 -0.13768077  0.10689959  0.02378752  0.11877958  0.7867637
 -0.3399043   0.4743451  -0.5134933   0.01867717 -0.5406575   0.33988017
  0.23846084 -0.6211012  -0.45058516 -0.18209127 -0.07521641  0.47604966
  0.09365943 -0.18236434  0.14887536  0.2909882 ]
```



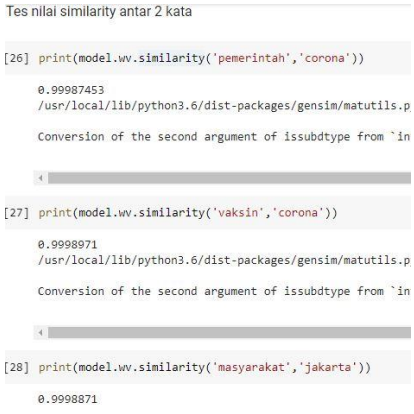
```
[22] import gensim.models

     sentences = MyCorpus()
     model = gensim.models.Word2Vec(sentences=sentences,size = 100, min_count=5)

2020-10-14 13:28:04,675 : INFO : collecting all words and their counts
2020-10-14 13:28:04,677 : INFO : PROGRESS: at sentence #0, processed 0 words, keeping 0 word types
2020-10-14 13:28:04,713 : INFO : collected 3812 word types from a corpus of 24333 raw words and 312 sentences
2020-10-14 13:28:04,714 : INFO : Loading a fresh vocabulary
2020-10-14 13:28:04,720 : INFO : effective_min_count=5 retains 899 unique words (23% of original 3812, drops 2913)
2020-10-14 13:28:04,720 : INFO : effective_min_count=5 leaves 19507 word corpus (80% of original 24333, drops 4826)
2020-10-14 13:28:04,729 : INFO : deleting the raw counts dictionary of 3812 items
2020-10-14 13:28:04,729 : INFO : sample=0.001 downsamples 67 most-common words
2020-10-14 13:28:04,730 : INFO : downsampling leaves estimated 15295 word corpus (78.4% of prior 19507)
2020-10-14 13:28:04,733 : INFO : estimated required memory for 899 words and 100 dimensions: 1168700 bytes
2020-10-14 13:28:04,733 : INFO : resetting layer weights
2020-10-14 13:28:04,907 : INFO : training model with 3 workers on 899 vocabulary and 100 features, using sg=0 hs=0 s
2020-10-14 13:28:04,947 : INFO : worker thread finished; awaiting finish of 2 more threads
2020-10-14 13:28:04,953 : INFO : worker thread finished; awaiting finish of 1 more threads

vec_positif = model.wv['covid']
print(vec_positif)

[ 0.35276854  0.28298452 -0.48434106  0.20892495 -0.08275152 -0.51659137
  0.5495379   0.04600057 -0.9030283   0.263558   -0.16316102  1.2187335
 -0.67125636  0.03081998 -0.1683466   0.10639236 -0.62544584  0.7386577
  0.36899188 -0.53658295  0.07768735  0.32070884 -0.58006316 -0.44428518
  0.22070138 -0.08169428 -0.40921476 -0.58205694 -0.34174064 -0.26027617
  0.06529167 -0.44196263  0.03659781 -0.63192016  0.17569384 -0.77974784
  0.25427294  0.20746177 -0.5014332   0.04405181  0.05814064 -0.34767535
  0.13279924  0.5230738  -0.24471463  0.47055876 -0.03442214  0.87062186
 -0.18592583  0.07998727  0.2648435   0.13162348 -0.57263315 -0.49423268
 -0.31795162 -0.17445585  0.8001803  -0.14122763  0.48523104  0.3291617
 -0.72325754  0.05060219  0.34463784 -0.11068175 -0.3578359   0.05756868
  0.18872626  0.13669431  0.8671354   0.8354314   0.19900392  0.31357726
  0.30316603 -0.1131751  -0.02332059  0.23904848 -0.04977218  0.03137538
 -0.5032374   0.09483157  0.29900625  0.06944838 -0.08434725  0.5731952
 -0.2739774   0.36629736 -0.9495105   0.3790426  -0.6206464   0.1812207
  0.20094869 -0.7154701  -0.5788972   0.251251   -0.15352759  0.501799
 -0.00680109 -0.14573407 -0.08576518  0.50211656]
```

2. Hitung nilai *similarity* antar kata dengan memilih 3 pasang kata yang Anda perkirakan nilai similaritynya.

3 pasang kata yang digunakan yaitu ‘pemerintah’ dengan ’corona’ , ‘vaksin’ dengan ‘corona’ , dan ‘masyarakat’ dengan ‘jakarta’. Akurasi yang didapatkan dari min_count=5 memiliki nilai similarity yang lebih besar dari min_count = 1. Karena jika kita melakukan setting dimana min_count = 5 maka kata-kata yang kemunculannya kurang dari 5 tidak akan disimpan pada konteks data. Sehingga akurasi yang dihasilkan akan lebih besar dari min_count =1.

```
[9] print(model.wv.similarity('pemerintah','corona'))

0.9997932
/usr/local/lib/python3.6/dist-packages/gensim/matutils.
  if np.issubdtype(vec.dtype, np.int):

print(model.wv.similarity('vaksin','corona'))

0.99983054
/usr/local/lib/python3.6/dist-packages/gensim/matutils.
  if np.issubdtype(vec.dtype, np.int):

[11] print(model.wv.similarity('masyarakat','jakarta'))

0.999783
/usr/local/lib/python3.6/dist-packages/gensim/matutils.
  if np.issubdtype(vec.dtype, np.int):
```

Gambar 2 min_count = 1



```
Tes nilai similarity antar 2 kata

[26] print(model.wv.similarity('pemerintah','corona'))

0.99987453
/usr/local/lib/python3.6/dist-packages/gensim/matutils.p

Conversion of the second argument of issubdtype from `in

[27] print(model.wv.similarity('vaksin','corona'))

0.9998971
/usr/local/lib/python3.6/dist-packages/gensim/matutils.p

Conversion of the second argument of issubdtype from `in

[28] print(model.wv.similarity('masyarakat','jakarta'))

0.9998871
```

Gambar 1 min_count=5

3. Cari top-5 kata yang similar dengan sebuah kata tertentu, amati hasilnya, berikan analisis terhadap hasil yang Anda peroleh. Bandingkan hasil

```
print(model.wv.most_similar(positive=['covid'], topn=5))

[('yang', 0.9999521970748901), ('dan', 0.9999505877494812), ('dengan', 0.9999473094940186), ('akan', 0.999931812286377), ('di', 0.9999316930770874)]
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning: Conversion of the second argument of issubdtype from `int` to `np.signed
  if np.issubdtype(vec.dtype, np.int):

[13] print(model.wv.most_similar(positive=['jakarta'], topn=5))

/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning: Conversion of the second argument of issubdtype from `int` to `np.signed
  if np.issubdtype(vec.dtype, np.int):
[('dan', 0.9998534321784973), ('di', 0.9998509287834167), ('kata', 0.9998445510864258), ('akan', 0.9998432397842407), ('dengan', 0.9998422861099243)]

[14] print(model.wv.most_similar(positive=['pemerintah'], topn=5))

[('di', 0.9999016523361206), ('dan', 0.9999012351036072), ('pada', 0.9999009370803833), ('yang', 0.9998890161514282), ('ke', 0.9998867511749268)]
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning: Conversion of the second argument of issubdtype from `int` to `np.signed
  if np.issubdtype(vec.dtype, np.int):
```

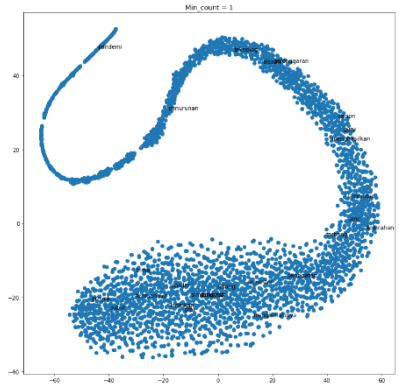
Gambar 4 min_count = 1

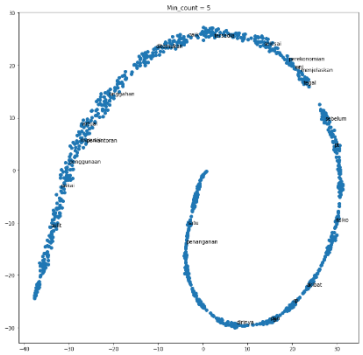
```
[29] print(model.wv.most_similar(positive=['covid'], topn=5))

[('yang', 0.9999622106552124), ('dengan', 0.9999610185623169), ('dan', 0.9999597668647766), ('akan', 0.9999531507492065), ('tidak', 0.9999496936798096)]
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning:

Conversion of the second argument of issubdtype from `int` to `np.signedinteger` is deprecated. In future, it will be treated as `np.int64 == np.dtype(i

[30] print(model.wv.most_similar(positive=['jakarta'], topn=5))

[('dan', 0.9999205470085144), ('di', 0.9999187588691711), ('akan', 0.9999154806137085), ('kata', 0.9999152421951294), ('melakukan', 0.9999145269393921)]
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning:

Conversion of the second argument of issubdtype from `int` to `np.signedinteger` is deprecated. In future, it will be treated as `np.int64 == np.dtype(i

[31] print(model.wv.most_similar(positive=['pemerintah'], topn=5))

[('pada', 0.9999227523803711), ('dan', 0.9999213814735413), ('di', 0.9999204277992249), ('dengan', 0.9999160170555115), ('dalam', 0.9999129176139832)]
/usr/local/lib/python3.6/dist-packages/gensim/matutils.py:737: FutureWarning:
```

Gambar 3. Min_count=5

Berdasarkan hasil akurasi min_count = 5 nilai similaritynya lebih besar dari min_count = 1 walaupun perbedanyaannya sangat kecil. Dan juga terdapat perbedaan kata yang similar antara min_count= 1 dengan min_count = 5 ,karena jika kita setting min_count = 5, maka kata-kata yang muncul kurang dari 5 akan dihilangkan dari konteks data. Sehingga hasil similarity dari sebuah kata juga akan berbeda.

4. Lakukan visualisasi embedding dengan matplotlib





Berdasarkan hasil visualisasi embedding dengan matplotlib visualisasi dari min_count = 1 data data lebih banyak yang berkelompok/cluster dibandingkan pada min_count = 1. Karena min_count = 1 menyebabkan semua data bisa menjadi data latih sedangkan min_count 5 data yang kemunculannya kurang dari 5 akan dihapus dari konteks data . Sehingga kata pada min_count = 1 lebih banyak dari pada kata pada min_count = 5.