test 1/2: W2V vs BERT - word to word (context)

```
## Warning: package 'data.table' was built under R version 3.4.4
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
## Warning in py_to_r.pandas.core.frame.DataFrame(result): index contains
## duplicated values: row names not set
```

```
## Warning in py_to_r.pandas.core.frame.DataFrame(result): index contains
## duplicated values: row names not set
```

DATA

dataset with mannual annotation of similaraties between pairs of words (wordsim353);

```
head(data_sim)
```

```
##          w1       w2 score
## 1:     love      sex  6.77
## 2:    tiger      cat  7.35
## 3:    tiger    tiger 10.00
## 4:     book    paper  7.46
## 5: computer keyboard  7.62
## 6: computer internet  7.58
```

and dataset with vector representations (w2v) of the set of words appeared (we show only the first two dimensions);

```
head(data_w2v_words[ , c(review_cols, "dim_1", "dim_2"), with = FALSE])
```

```
##    id     w id_token token       dim_1       dim_2
## 1:  1  love        1     i -0.22558594 -0.01953125
## 2:  1  love        2  love  0.10302734 -0.15234375
## 3:  1  love        3   you  0.20410156  0.01318359
## 4:  2 tiger        1     i -0.22558594 -0.01953125
## 5:  2 tiger        2   saw  0.09423828  0.20117188
## 6:  2 tiger        3 tiger -0.06835938  0.18261719
```

we select rows with equal w and token

NOTE: w2v is context-free but we could test it.

Also, we have our BERT representations (free-context) in this way,

```
head(data_bert_words[ , c(review_cols, "dim_w_1", "dim_w_2"), with = FALSE])
```

```
##    id        w id_token    token     dim_w_1     dim_w_2
## 1:  1     love        1     love  0.38649017  0.36187920
## 2:  2    tiger        1    tiger -0.30712840 -0.31644982
## 3:  3     book        1     book  0.41267234 -0.00370523
## 4:  4 computer        1 computer -0.67297429 -0.10599531
## 5:  5    plane        1    plane  0.01826328 -0.21225268
## 6:  6    train        1    train -0.52878708 -1.14276505
```

```
## [1] 428
```

```
## [1] 428
```

Firstly, we remove words that we haven't in both datasets (4 words are lost in w2v dataset because are out-of-vocabulary). Now, we can compute (cosine) similarities between vector representations (w2v and BERT) pairs of words in "data_sim" and compare with the mannual scoring.

```
##          w1       w2 score w2v_cosine bert_cosine
## 1:     love      sex  6.77  0.2639377   0.6756448
## 2:    tiger      cat  7.35  0.5172962   0.7996021
## 3:    tiger    tiger 10.00  1.0000000   1.0000000
## 4:     book    paper  7.46  0.3634626   0.5779281
## 5: computer keyboard  7.62  0.3963916   0.8194257
## 6: computer internet  7.58  0.4068623   0.5341467
```

With this dataset we can compare mannual similarity with cosine metric for w2v and BERT representations.

ANALYSIS

We clean NA puntuations, and we get 328 complete rows.

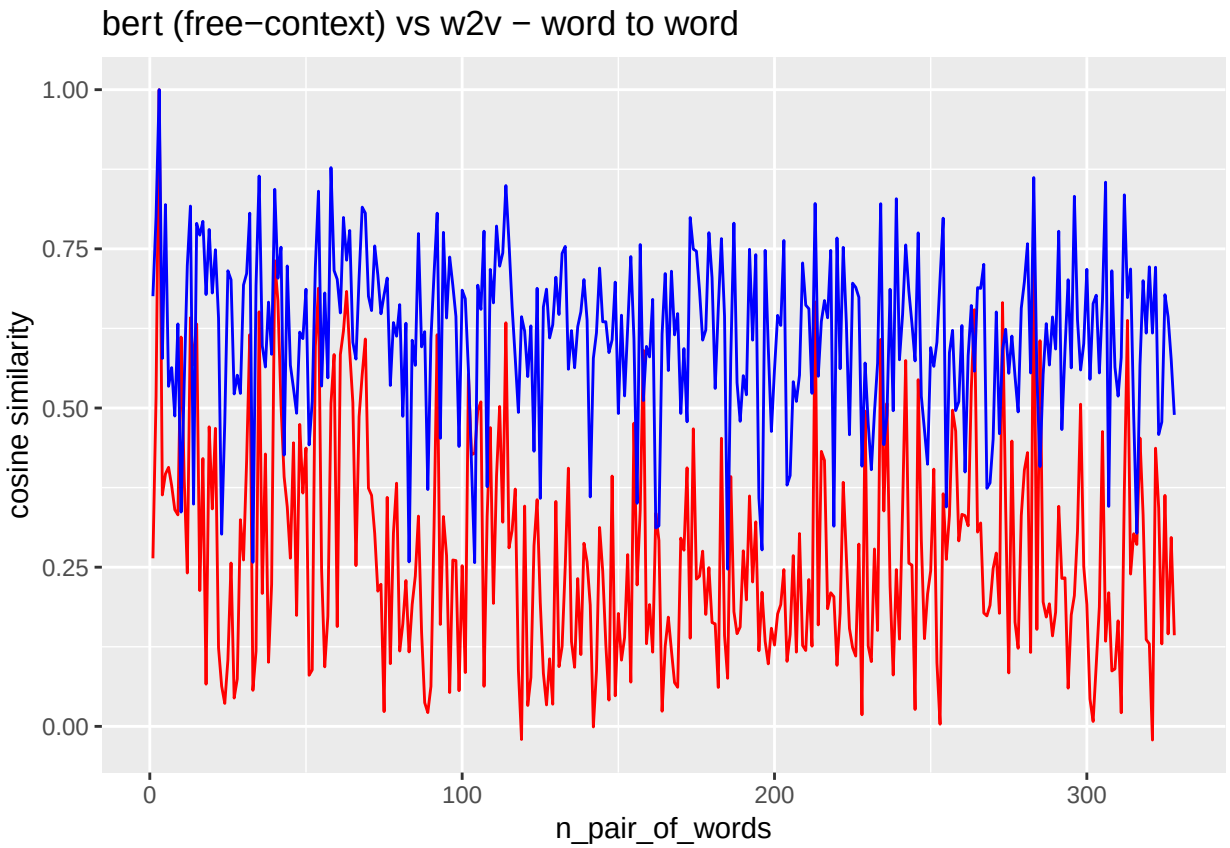
NOTE: we observe 3 values of cosine (w2v) < 0

```
##              w1        w2 score    w2v_cosine bert_cosine
## 1:    precedent cognition  2.81 -0.0205696013   0.6433847
## 2:         mars     water  2.94 -0.0009222099   0.5781125
## 3: preservation     world  6.19 -0.0214706240   0.6177243
```

```
## Empty data.table (0 rows) of 5 cols: w1,w2,score,w2v_cosine,bert_cosine
```

We can plot similarities scores,

```
ggplot() + geom_line(aes(x = seq(1:length(data_sim$w1)), y = data_sim$w2v_cosine),color='red') +
           geom_line(aes(x = seq(1:length(data_sim$w1)), y = data_sim$bert_cosine),color='blue') +
           ylab('cosine similarity') + xlab('n_pair_of_words') + ggtitle("bert (free-context) vs w2v - w
```



We can observe the best results for BERT representations.

Also we can compute the Pearson coefficient in both case respect to the mannual scores in dataset. For w2vec similarities,

```
cor(data_sim$score, data_sim$w2v_cosine, method = c("pearson"))
```

```
## [1] 0.655213
```

and for BERT cosine similarities,

```
cor(data_sim$score, data_sim$bert_cosine, method = c("pearson"))
```

```
## [1] 0.2467143
```

We can observe highest similarity metric for BERT representations, but has a worst correlation with mannual scoring.

And corrlation between both vector representations scoring is,

```
cor(data_sim$w2v_cosine, data_sim$bert_cosine, method = c("pearson"))
```

```
## [1] 0.3178288
```

BERT non-free-context

Now, we can use the BERT vector representation of the same word got from word in a phrase (context).

In the next dataset we have a phrase containing the word and we have the vector representation got in this case,

```
## Warning in py_to_r.pandas.core.frame.DataFrame(result): index contains
```

```
## duplicated values: row names not set
```

```
##    id     w id_token token dim_context_1 dim_context_2
## 1:  1  love        1     i     0.2335791    0.24898028
## 2:  1  love        2  love     0.9315368    0.91580886
## 3:  1  love        3   you    -0.1724851   -0.77048999
## 4:  2 tiger        1     i     0.1126512   -0.53610945
## 5:  2 tiger        2   saw    -0.1356604   -0.20776120
## 6:  2 tiger        3     a    -0.2600941   -0.07189066
```

and we select the corresponding vector,

```
##    id        w dim_context_1 dim_context_2
## 1:  1     love     0.9315368     0.9158089
## 2:  2    tiger     0.3108854     0.1306733
## 3:  3     book     0.1590683    -0.4085730
## 4:  4 computer    -0.4819463     0.2318698
## 5:  5    plane     1.0386617    -0.6260117
## 6:  6    train     1.0346285    -0.8108625
```

```
##          w1       w2 score w2v_cosine bert_cosine bert_context_cosine
## 1:     love      sex  6.77  0.2639377   0.6756448           0.2499513
## 2:    tiger      cat  7.35  0.5172962   0.7996021           0.5943350
## 3:    tiger    tiger 10.00  1.0000000   1.0000000           1.0000000
## 4:     book    paper  7.46  0.3634626   0.5779281           0.6570200
## 5: computer keyboard  7.62  0.3963916   0.8194257           0.4377427
## 6: computer internet  7.58  0.4068623   0.5341467           0.4450932
```

in this case the before results are,

```
cor(data_sim$score, data_sim$bert_context_cosine, method = c("pearson"))
```

```
## [1] 0.3516758
```

we can observe a bit improve respect to results with BERT free-context.