# Lab report

## Exercise 2: Word Embeddings

# Preprocessing

Preprocessing was an important step to obtain the best possible model. A few first typical steps were chosen: lowercasing and tokenizing the text.
After that, we examined the text to determine the next steps to take.
We chose to keep only the texts written by Shakespeare himself, not all the texts around (publishers, conditions of use, copyrights, etc).
Removing the numerals was an important step, as even if they appear in a similar context, they are arbitrary. They could also have been replaced with a single word.
After that, we decided to also get rid of all the [_*_] occurrences (they are the *blockings* of the theatre plays). We also removed the SCENE+Roman numerals and all the special punctuation.

# Word embedding and training

The next steps were the coding of the different classes. For those, we took inspiration from the different books we had (particularly Rao and McMahan) as well as blog posts dealing with word embeddings. The classes we chose were:

- **Vocabulary** (to create dictionaries to look-up the words and their index),
- **Vectorizer** (to vectorize the context of the target word)
- **ShakespeareDataset** (which loads the data and preprocesses it)
- **CBOW** (inspired by the one in the preliminary exercise)
- **TrainingRoutine** (which does the training of a single model)
- And also some honorable mentions: GridSearch, CBOWEvaluator, …

Even though this class setup might be a bit excessive for this given task, we have decided to include most of our code. Not just the evaluation part. Initially, we have developed the codebase via jupyter notebooks and later merged it into one *.py* file. We recommend reading the code from the program's entrypoint (the "main()" function).

From there, as a reviewer, you are able to change the constants (line 906-909). These constants modify the behaviour of the main loop. You can leverage the fact that our code is also capable of loading previously trained models. We have decided to include our best models in the final *.zip.*

Each model is 22MB (sorry) and took 12-14 hours to train on a 1080Ti / K80 GPU. Please understand our frustration that led to including these models here. If `TRAIN=False` the program won't begin the training but rather loads these previously trained models and

prepares them for evaluating. After a few minutes the evaluator will ask for your input (a word to evaluate).

If `TRAIN=True` the program starts a quick gridsearch with the `data_frac` parameter being set to a very low amount intentionally. This is just to showcase that the training / gridsearch part works as intended. The evaluation part starts right after the training is done. Please note, that in this case the vocabulary size corresponds to the full vocabulary size (which is len(set(~1.2M words))) times the `data_frac` parameter. Thus, most of the words you would expect are not present in the trained embeddings.

# Computation power and issues

One of the biggest issues we had was how expensive the calculations were. The data was pretty big, and deep neural networks need a lot of power. Without GPUs and no Google Cloud or other services (friends with GPUs), it was very difficult to run the whole training. It was easily done with fractions of the data to test our code and see how it was running, but training the whole dataset was a whole other story: it took time, made our computers overheat. Those were issues that needed to be dealt with and combined with the stress of all the other issues that needed to be solved, it created some unnecessary trouble.

# Results CBOW2

## Common words (frequency)

### King (3092):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[43.36] - fait
...[44.55] - donnerai
...[45.66] - meilleur
...[46.77] - claudio
...[46.93] - caesar
===Cosine Similarity (higher better)===
...[0.73] - claudio
...[0.73] - brutus
...[0.70] - flowed
...[0.68] - caesar
...[0.67] - three-and-twenty
```

### Good (2946):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[53.68] - if
...[54.01] - let
...[55.95] - my
...[57.11] - pray
...[58.50] - there
===Cosine Similarity (higher better)===
...[0.82] - consort
...[0.79] - esperance
...[0.79] - my
...[0.78] - if
...[0.78] - cauldron
```

### Come(2620):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[38.48] - well
...[43.27] - now
...[46.50] - what
...[49.92] - peace
...[53.88] - away
===Cosine Similarity (higher better)===
...[0.85] - well
...[0.79] - ah
...[0.79] - now
...[0.79] - ay
...[0.77] - what
```

## Mid-Common words (frequency)

### Life (946):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[51.99] - heart
...[54.22] - valour
...[56.50] - eye
...[57.16] - time
...[57.72] - state
===Cosine Similarity (higher better)===
...[0.84] - eye
...[0.79] - valour
...[0.77] - soul
...[0.77] - blood
...[0.77] - spirit
```

### Sweet (856):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[57.84] - 't
...[57.96] - thank
...[58.72] - who
...[59.38] - pardon
...[59.83] - how
===Cosine Similarity (higher better)===
...[0.82] - 't
...[0.81] - thank
...[0.81] - hither
...[0.80] - who
...[0.80] - pardon
```

### Think (1073):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[54.05] - nothing
...[54.65] - warrant
...[56.04] - like
...[57.11] - out
...[58.19] - them
===Cosine Similarity (higher better)===
...[0.85] - mean
...[0.81] - near
...[0.81] - heard
...[0.81] - return
...[0.80] - spoke
```

## Rare words (frequency)

### Fact (13):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[66.75] - son-in-law
...[67.90] - grim
...[68.39] - extend
...[69.47] - deceived
...[69.94] - eternally
===Cosine Similarity (higher better)===
...[0.92] - enrich
...[0.92] - fain
...[0.92] - twere
...[0.91] - convenient
...[0.91] - trusted
```

### Comfortable (13):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[62.32] - airs
...[64.51] - diet
...[67.91] - supremacy
...[68.59] - milks
...[69.31] - curb
===Cosine Similarity (higher better)===
...[0.92] - chose
...[0.92] - yond
...[0.91] - diet
...[0.91] - airs
...[0.91] - fraught
```

### Oppress (13):

```
--------------------------------------------------
Model (Learning Rate: 0.01, Epochs: 200): CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[84.12] - slumbers
...[85.72] - grudge
...[89.58] - faintly
...[89.79] - prompt
...[90.92] - clos
===Cosine Similarity (higher better)===
...[0.95] - prompt
...[0.95] - slumbers
...[0.95] - grudge
...[0.95] - restor
...[0.95] - faintly
```

# Discussion[1]

The very common words are not the one that output the best results, interestingly.
For **King**, “Claudio”, “Brutus” and “Caesar” seem to make sense (famous king names).
**Good**, however, is completely wrong: the outputted words aren’t even of the same Part-of-Speech! **Come** is also quite bad.
The mid-common words seem to have better results. This might be attributable to the fact that they are seen in a smaller variety of contexts than the more common ones (just a hypothesis). **Life**, with words such as “blood”, “eye” and “valour” is a good example of successful embedding. **Sweet** has very bad results, but think has the merit to have 4 out 5 words that are indeed verbs. “Mean” comes close to it, as does “spoke”, although the tense is not the same.
The rare words have very bad results, and we can’t really explain how they are coming close to the word we researched.
Generally speaking, there are two trends to be noticed:
-mid-common words seem to obtain the most accurate embeddings
-nouns seem to obtain the most accurate embeddings generally speaking
These trends, however, should be taken with caution: 9 words is certainly not a sample big enough to make relevant statements.

[1] NB: for both discussions, we tended to focus on the word outputted by cosine similarity because they tended to be, in our opinion, slightly better. There are remarks about pairwise similarity too, though, especially when they were notable.

# Results CBOW5

## Common words (frequency)

### King (3092):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[3.67] - queen
...[3.70] - prince
...[4.59] - gentleman
...[4.98] - servant
...[5.16] - fool
===Cosine Similarity (higher better)===
...[0.90] - queen
...[0.88] - prince
...[0.78] - gentleman
...[0.77] - cardinal
...[0.77] - soldier
```

### Good (2946):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[3.95] - sweet
...[4.18] - noble
...[4.32] - me
...[4.43] - come
...[4.49] - great
===Cosine Similarity (higher better)===
...[0.73] - come
...[0.72] - then
...[0.70] - sweet
...[0.69] - fellow
...[0.68] - why
```

### Come (2620):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[4.09] - speak
...[4.43] - good
...[4.43] - look
...[4.51] - go
...[4.82] - then
===Cosine Similarity (higher better)===
...[0.78] - speak
...[0.76] - look
...[0.74] - why
...[0.74] - go
...[0.74] - well
```

## Mid-Common words (frequency)

### Life (946):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[4.84] - heart
...[5.25] - face
...[5.27] - father
...[5.38] - love
...[5.48] - soul
===Cosine Similarity (higher better)===
...[0.76] - mother
...[0.76] - heart
...[0.74] - son
...[0.74] - daughter
...[0.73] - father
```

### Sweet (856):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[3.95] - good
...[4.03] - noble
...[4.13] - dear
...[4.79] - great
...[4.80] - master
===Cosine Similarity (higher better)===
...[0.72] - noble
...[0.71] - dear
...[0.70] - cousin
...[0.70] - good
...[0.68] - master
```

### Think (1073):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[4.44] - for
...[4.54] - see
...[4.56] - know
...[4.56] - and
...[4.57] - say
===Cosine Similarity (higher better)===
...[0.75] - say
...[0.71] - know
...[0.70] - and
...[0.69] - for
...[0.66] - see
```

## Rare words (frequency)

### Fact (13):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[13.83] - drown
...[14.40] - sap
...[14.60] - cloud
...[14.67] - succeeds
...[14.73] - tale
===Cosine Similarity (higher better)===
...[0.83] - forbidden
...[0.80] - laughing
...[0.79] - dower
...[0.79] - him—
...[0.78] - buzz
```

### Comfortable (13):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[14.57] - trod
...[15.49] - hangs
...[15.57] - prohibition
...[15.59] - inviolable
...[15.74] - jot
===Cosine Similarity (higher better)===
...[0.79] - trod
...[0.77] - slowly
...[0.76] - absey
...[0.76] - doubled
...[0.75] - sakes
```

### Oppress (13):

```
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[13.12] - needle
...[14.24] - belov
...[14.53] - steeds
...[14.66] - r
...[14.69] - condemn
===Cosine Similarity (higher better)===
...[0.85] - needle
...[0.84] - atomies
...[0.84] - steeds
...[0.83] - new-made
...[0.83] - whate
```

## Discussion

As one can see, CBOW5 gives results that are quite relevant for most words. The word **King** gives especially good results (in both cosine similarity and pairwise similarity), while the word Good has its best results in the pairwise similarity list. **Come** has two verbs at the top ("speak" and "look"), and "go" some place further, but the other words doesn't seem to make that much sense. In the mid-common words, "**life**" also outputted same PoS and they make much sense, especially given the usually quite dramatic atmosphere in Shakespeare's writings. For **Sweet**, "noble", "dear", "cousin" and "good" are seemingly good fits. For the verb **Think**, "say", "know" and "see" are very relevant answers.
In the least common words, all three words have quite bad results.

To sum it up, words with a relatively high frequency do usually have meaningful embeddings, especially nouns, as it seems. However, once the frequency becomes low, the words seem to be chosen almost by chance, which is probably a little the case, as the model hasn't seem them enough to really know what they should mean.

# Comparison of both models

A comparison of both model lets the CBOW5 model win. Its results are usually more meaningful and they also respect the PoS of the given word better. Those are two important things. However, especially with nouns, CBOW2 also did a great job. There are a few other instances of a very good word outputted by CBOW2 that wasn't present in CBOW5.
As for everything, context size probably has a sweet spot: all context sizes smaller or bigger than this one are giving worse results. This can be explained by the fact that smaller context sizes might allow for many more different words, some not related to the target word or not sharing the same PoS. On the other hand, if the context size is too big, the choice of possible options might be very limited: as the context becomes more and more specific, the model will have a harder time to generalize.
A context size of 5 seems to have done a great job, it would be interesting to experiment a little to see if context sizes close to it would be somewhat better or not.

# References


Delip Rao and Brian McMahan. *Natural language processing with PyTorch: build intelligent language applications using deep learning*. O'Reilly, First edition ed., 2019.

https://iksinc.online/tag/continuous-bag-of-words-cbow/
http://mccormickml.com/assets/word2vec/Alex_Minnaar_Word2Vec_Tutorial_Part_II_The_Continuous_Bag-of-Words_Model.pdf
https://stackoverflow.com/questions/48479915/what-is-the-preferred-ratio-between-the-vocabulary-size-and-embedding-dimension
https://github.com/FraLotito/pytorch-continuous-bag-of-words/blob/master/cbow.py
https://stackoverflow.com/questions/50792316/what-does-1-mean-in-pytorch-view

https://www.tensorflow.org/tutorials/text/word_embeddings
https://pytorch.org/docs/stable/nn.html
https://pytorch.org/tutorials/beginner/blitz/autograd_tutorial.html
https://github.com/ChristophAlt/embedding_vectorizer/blob/master/embedding_vectorizer.py
https://pytorch.org/tutorials/beginner/saving_loading_models.html

# Example

An example of both models evaluating at the same time can be found below:

```
============================ Evaluating 'king' on: ==================
Model (Context: 2, LR: 0.01, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[43.36] - fait
...[44.55] - donnerai
...[45.66] - meilleur
...[46.77] - claudio
...[46.93] - caesar
===Cosine Similarity (higher better)===
...[0.73] - claudio
...[0.73] - brutus
...[0.70] - flowed
...[0.68] - caesar
...[0.67] - three-and-twenty
============================================================
Model (Context: 5, LR: 0.001, Epochs: 200):
CBOW(
  (embeddings): Embedding(31268, 50)
  (linear1): Linear(in_features=50, out_features=128, bias=True)
  (linear2): Linear(in_features=128, out_features=31268, bias=True)
)

===Pairwise Distance (lower better)===
...[3.67] - queen
...[3.70] - prince
...[4.59] - gentleman
...[4.98] - servant
...[5.16] - fool
===Cosine Similarity (higher better)===
...[0.90] - queen
...[0.88] - prince
...[0.78] - gentleman
...[0.77] - cardinal
...[0.77] - soldier
============================================================
```