CentraleSupélec

Introduction to Natural Language Processing

Part 1: Word Embeddings

Camilo Carvajal Reyes

3rd March 2021

Objectives

After today's lecture and tutorials you should be able to:

- Know what is **Natural Language Processing (NLP)** and why it's important
- Identify the main algorithms in **text representation learning**
- Spot common **challenges** in NLP and **ethical concerns**
- Gain intuition on what kind of information gets encoded in **word representations** and how it gets shown in the embedding space
- Get familiarised with the use of pre-trained **language models**, how to interpret them and implement them in a simple NLP pipeline

Outline

- *8:30 - 9:00* - **Lecture**
  Word Embeddings: challenges and techniques
- *9:00 - 9:30* - **Practical Work**
  Visualisation of semantic relations
- *9:30 - 9:45* - Break
- *9:45 - 10:30* - **Lecture**
  Language Modelling: deep learning methods for NLP
- *10:30 - 11:00* - **Practical Work**
  Transformer LM Part 1: Exploring a pre-trained model
- *11:00 - 11:15* - Break
- *11:15 - 11:45* - **Practical Work**
  Transformer LM Part 2: Fine-tuning for Sentiment Analysis

CentraleSupélec

Table of contents

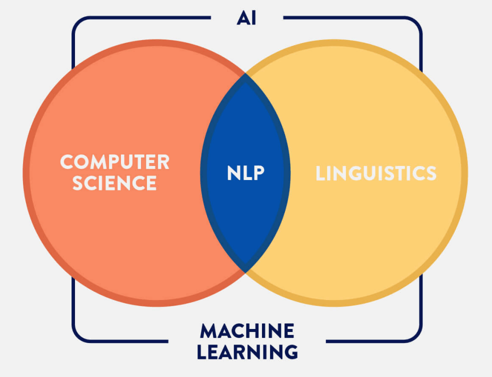
Embedding，嵌入式

CentraleSupélec

What is NLP?

Natural Language Processing is a sub-field of linguistics, computer science and artificial intelligence, that connects computers and human language.

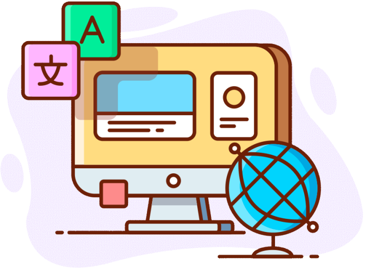
The field has experienced major changes since recent growth of **Deep Learning**.



Why bothering at all with NLP?

Computational encoding of text
opens the gate for:

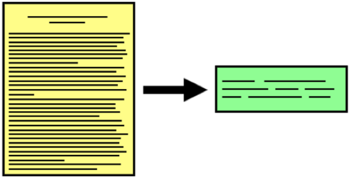
- Machine Translation



Why bothering at all with NLP?

Computational encoding of text opens the gate for:

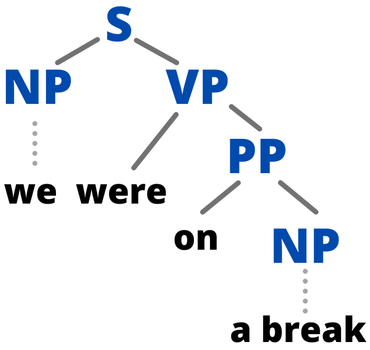
- Machine Translation
- Natural Language Understanding
- Automatic Text Summarisation



Why bothering at all with NLP?

Computational encoding of text opens the gate for:

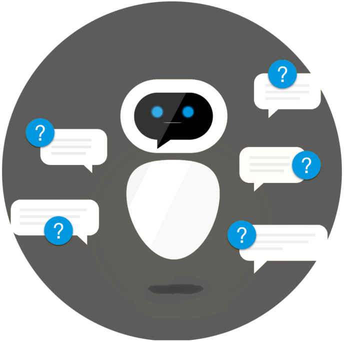
- Machine Translation
- Natural Language Understanding
- Automatic Text Summarisation
- Computational Linguistics



Why bothering at all with NLP?

Computational encoding of text opens the gate for:

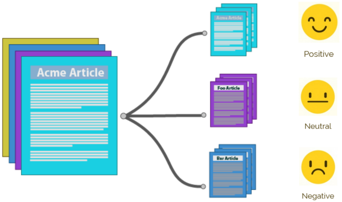
- Machine Translation
- Natural Language Understanding
- Automatic Text Summarisation
- Computational Linguistics
- Natural Language Generation
- Question-Answering



Why bothering at all with NLP?

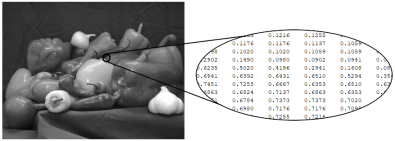
Computational encoding of text opens the gate for:

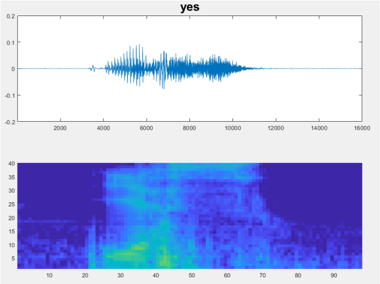
- Machine Translation
- Natural Language Understanding
- Automatic Text Summarisation
- Computational Linguistics
- Natural Language Generation
- Question-Answering
- Sentiment Analysis



Differences with Image and Audio processing

With both images and audio, we start from a **signal**.
That signal is often of little use without some appropriate pre-processing.





CentraleSupélec

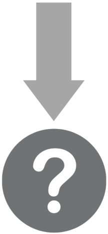
Differences with Image and Audio processing

With both images and audio, we start from a **signal**.
That signal is often of little use without some appropriate pre-processing.

However, this isn't the case for text.

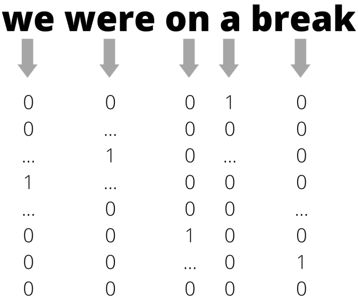
Hence, the **representation learning** process happens from scratch

we were on a break



Differences with Image and Audio processing

==One hot-encoding is one possible approach for giving a vector to a word.== Let's consider a vocabulary of possible words V with n terms. A one-hot representation for the word w, indexed by i in the vocabulary will correspond to the vector that has only zeros, except for the position i. That way, we can distinguish two different words.



Differences with Image and Audio processing

However, in one-hot encoding we encounter the following problems:

- **Sparsity**: For a vocabulary of 10 thousand words we would have 10 thousand dimensional vectors
- **Lack of information**: the vectors wouldn't encode any semantic information.

Therefore, we need a better way to construct our vectors.

we	were	on	a	break
0	0	0	1	0
0	...	0	0	0
...	1	0	...	0
1	...	0	0	0
...	0	0	0	...
0	0	1	0	0
0	0	...	0	1
0	0	0	0	0

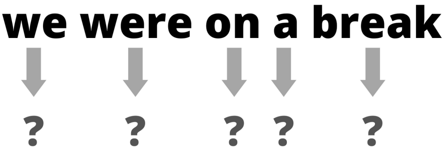
CentraleSupélec

CentraleSupélec

Word Embeddings

The main idea with **word embeddings** is to create a vector for each word that is:

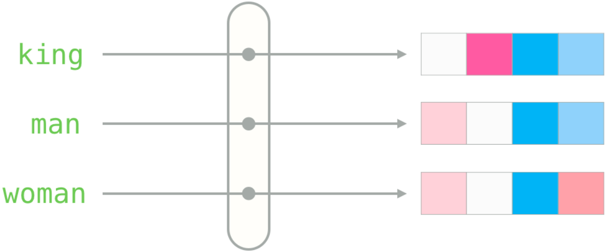
- **informative**: They encode useful information
- **light-weight**: Lower dimension than the size of the vocabulary
- **easy-to-obtain**: It's expensive to label an entire vocabulary, so we must learn representations automatically and with low computational cost.



Word Embeddings

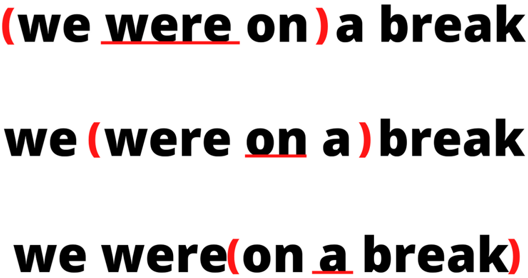
The main idea with **word embeddings** is to create a vector for each word that is:

- **informative**: They encode useful information
- **light-weight**: Lower dimension than the size of the vocabulary
- **easy-to-obtain**: It's expensive to label an entire vocabulary, so we must learn representations automatically and with low computational cost.



word2vec: Context-Window Methods

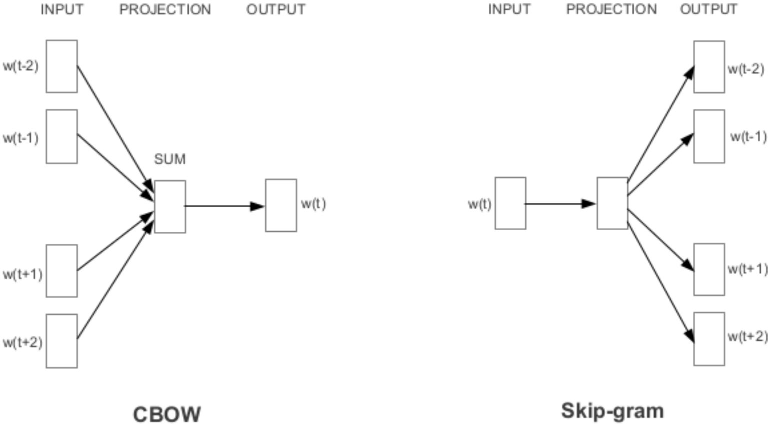
This section is based on **word2vec** by Mikolov et al. 2013 [1]. It is one of the most widely used pre-trained models for creating word vectors. They propose two architectures that are based on **context windows**[1].



[1]Figure shows examples of context windows of length 3 with the centre word underlined

word2vec: Context-Window Methods

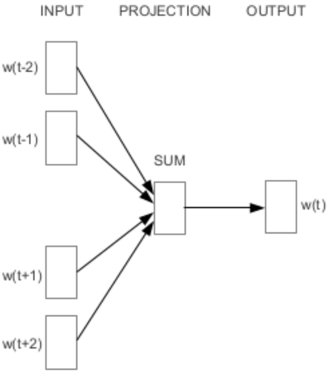
They are also called **Log-linear** models since they use a Log-linear classifier as training objective.



Continuous Bag-of-Words

In **CBOW**:

1. We take the **context words**. It's a bag-of-words, so order doesn't matter.
2. We project them in the **embedding space**
3. We **sum** the representations
4. We use that continuous representation of context to **predict the centre word.**

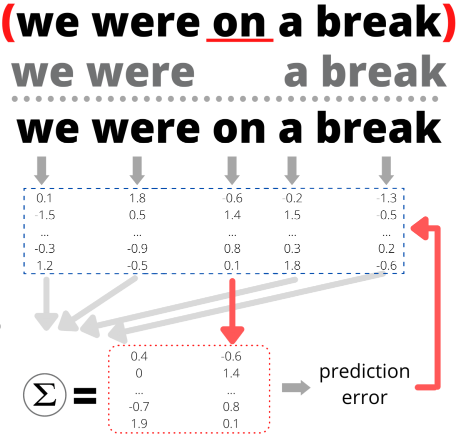


CBOW

Continuous Bag-of-Words

In **CBOW**:

1. We take the **context words.** It's a bag-of-words, so order doesn't matter.
2. We project them in the **embedding space**
3. We **sum** the representations
4. We use that continuous representation of context to **predict the centre word.**



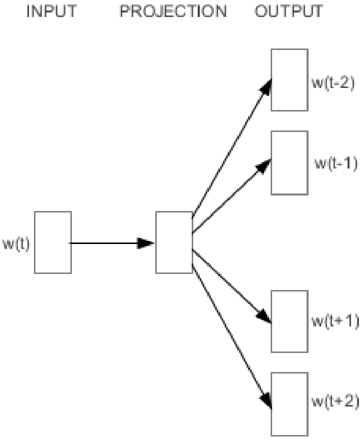
For our previous example, with context window of length 5, we would have

Skip-Gram

In **Skip-Gram**:

1. We take the **centre word**.
2. We project it in the **embedding space**
3. We **predict the context words** given the centre word representation.

This is roughly equivalent to compare centre and context words pairwise.



Skip-gram

CentraleSupélec

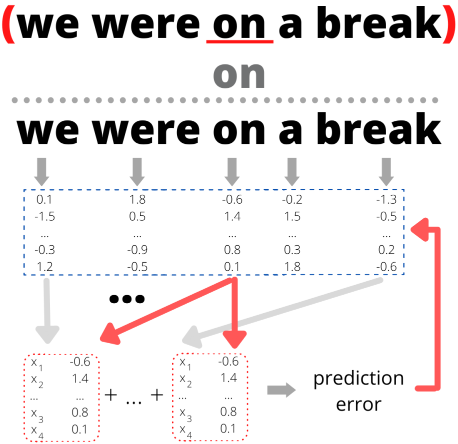
Skip-Gram

In **Skip-Gram**:

1. We take the **centre word.**
2. We project it in the **embedding space**
3. We **predict the context words** given the centre word representation.

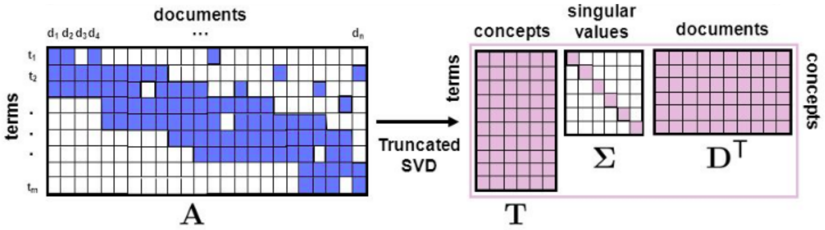
This is roughly equivalent to compare centre and context words pairwise.

For our previous example, with context window of length 5, we would have



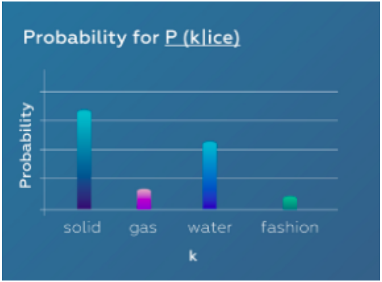
Matrix Factorisation: count-based models

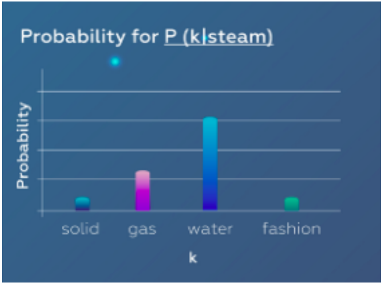
Count-based models take advantage of **word co-occurrence matrices**. Latent semantic analysis is one example in which matrix decomposition is used to grasp **statistical information** of terms and documents [2]. However, these methods generally **do not reflect semantic relations** in the same way word2vec does.



GloVe: a Hybrid Method

Global Vectors (**GloVe**) proposed by Pennington et al. seek to use statistical information while keeping the benefits of Skip-Gram [3]. For two words, they proceed by making the dot product of their embeddings equal to the logarithm of the **words' probability of co-occurrence** from the count matrix. To understand this, let's consider an example[2] with the words ICE and STEAM:

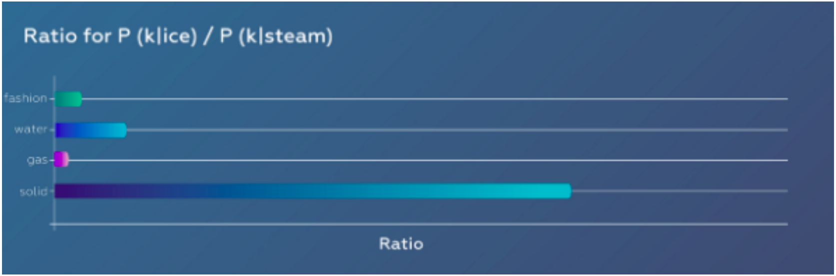




[2] Illustrations from: https://medium.com/@sciforce

GloVe: a Hybrid Method

Since both ICE and STEAM are related to WATER and unrelated to FASHION, these two words might be considered as **noise** when comparing them. The important information rather relies in the respective thermodynamic states.



GloVe implicitly causes the **vector differences** to correlate with **ratios of probabilities of co-occurrence.** This encodes more fine-grained **semantic relations** than pure context sharing.

CentraleSupélec

Incorporating Knowledge

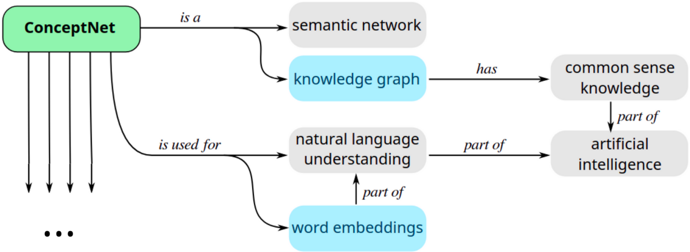
It is also possible to learn vectors in a (semi) supervised way. Here are two examples:

- **Retrofitting** is an algorithm that **fine tunes** pre-trained embeddings by minimising the distance between representations of words that are adjacent to each other in a **semantic graph** [4].

Incorporating Knowledge

It is also possible to learn vectors in a (semi) supervised way. Here are two examples:

- **Numberbatch** embeddings incorporate common sense knowledge from **ConceptNet**[3] by incorporating it using Retrofitting [5]. It is also partly built using a combination of word2vec and GloVe algorithms.

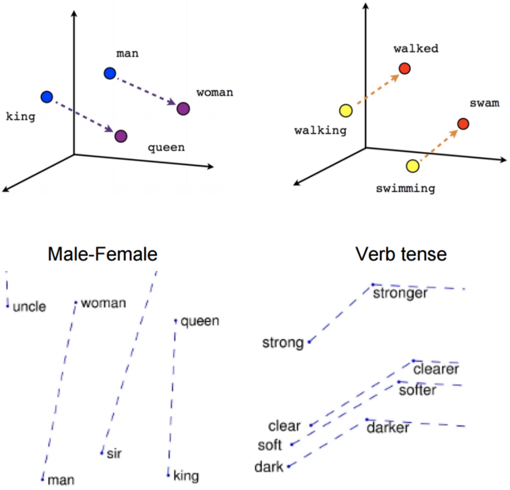


[3] http://www.conceptnet.io/

Semantic Regularities

Due to the way they are constructed, both word2vec (top) and GloVe (bottom) have shown to exhibit **linear substructures** [3, 6].

We'll further explore their embedding capacity in the practical session using Principal Component Analysis (**PCA**)



Drawbacks

Learning word representations comes at a cost:
Representing words as a point in an Euclidean space might not fully encode natural uncertainties in language such as **multiple senses** and **affective semantics**. Some works have tried with more complex embedding spaces.

Furthermore, drawing vectors from large corpora is dangerous since they can hide **stereotyped representations** [7]. A possible solution is debiasing vectors[8], which is done with Numberbatch[a].

[a] http://blog.conceptnet.io/posts/2017/conceptnet-numberbatch-17-04-better-less-stereotyped-word-vectors/

CentraleSupélec

In a Nutshell

- word2vec uses context windows for learning word representations by either predicting the centre word (CBOW) or the context words (Skip-Gram)
- GloVe also incorporates co-occurrence probabilities from a term-document count matrix.
- Vectors can then be fine-tuned using semantic information, like Numberbatch does.
- NLP consists of many useful applications for which representation learning is the departing point
- However, we need to be aware of the possible harmful data encoded in pre-trained vectors

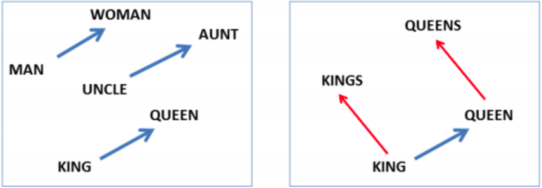
CentraleSupélec

Visualisation of semantic relations

Objectives: You should gain intuition on what kind of information gets encoded in word representations and how to visualise it.

To do: You will code a **PCA visualisation of semantic relations** using Numberbatch. They can base their algorithm in an example given with countries and capitals.

Bonus task: Interested students may implement a simple semantic evaluation for word embeddings.



References I

[1] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. ArXiv:1301.3781 [Cs]. http://arxiv.org/abs/1301.3781

[2] Deerwester, S. C., Dumais, S. T., Furnas, G. W., Harshman, R. A., CA, Landauer, T. K., Lochbaum, K. E., Streeter, L. A. (1989). United States Patent: 4839853 - Computer information retrieval using latent semantic structure (Patent No. 4839853).

[3] Pennington, J., Socher, R., Manning, C. (2014). Glove: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532–1543. https://doi.org/10.3115/v1/D14-1162

[4] Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., Smith, N. A. (2015). Retrofitting Word Vectors to Semantic Lexicons. Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 1606–1615. https://doi.org/10.3115/v1/N15-1184

[5] Speer, R., Chin, J., Havasi, C. (2017). ConceptNet 5.5: An open multilingual graph of general knowledge. Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4444–4451.

[6] Mikolov, T., Yih, W., Zweig, G. (2013). Linguistic Regularities in Continuous Space Word Representations. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 746–751. https://www.aclweb.org/anthology/N13-1090

[7] Caliskan, A., Bryson, J. J., Narayanan, A. (2017). Semantics derived automatically from language corpora contain human-like biases. Science, 356(6334), 183–186. https://doi.org/10.1126/science.aal4230

[8] Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., Kalai, A. T. (2016). Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29 (pp. 4349–4357). Curran Associates, Inc. http://papers.nips.cc/paper/6228-man-is-to-computer-programmer-as-woman-is-to-homemaker-debiasing-word-embeddings.pdf

CentraleSupélec

Introduction to Natural Language Processing

Part 1: Word Embeddings

Camilo Carvajal Reyes

3rd March 2021