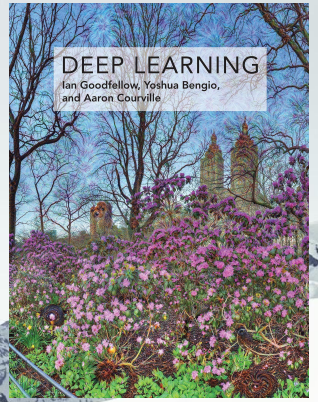
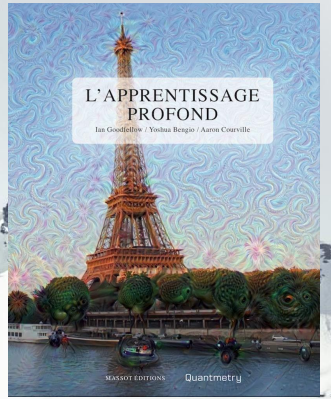
Deep Learning
pour le traitement du langage naturel (TALN)

Emanuela Boros
University of La Rochelle, France

emanuela.boros@univ-lr.fr

February 2021

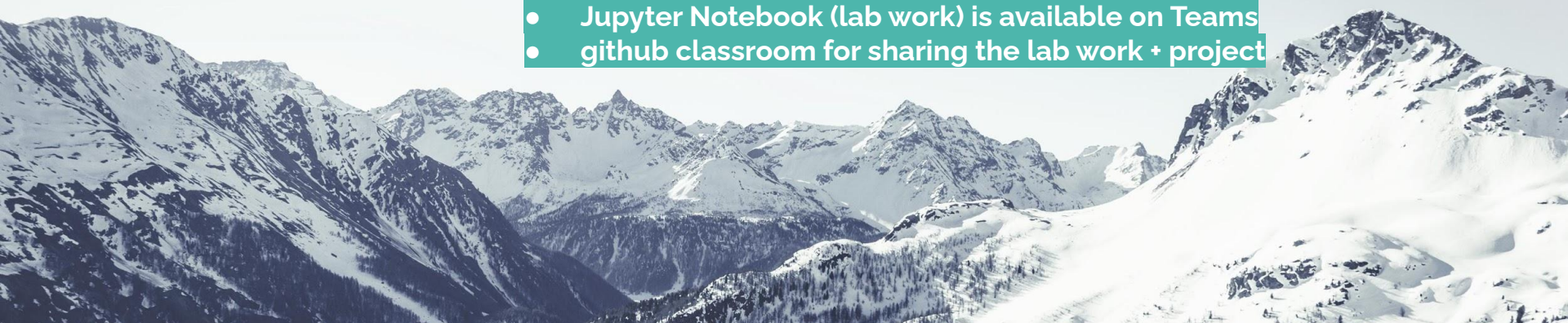




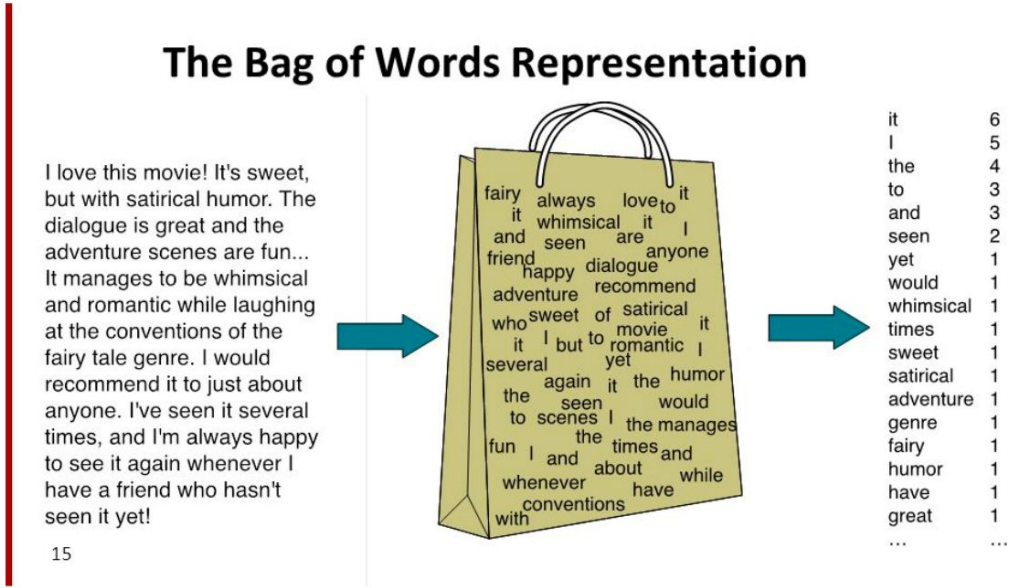
Organization

1. 3h Courses: Word Embeddings & Text Classification
2. 3h Lab work (dataset provided)
3. 3h Project (dataset provided)

- Course (slides) are available on Teams
- Jupyter Notebook (lab work) is available on Teams
- github classroom for sharing the lab work + project



Représentation vectorielle de documents



Représentation vectorielle de documents

Je vous envoie ma nouvelle adresse. Je vous remercie.

Modèle binomial : présence / absence de mot

adresse	commande	envoie	je	ma	nouvelle	réclame	remercie	vous
1	0	1	1	1	1	0	1	1

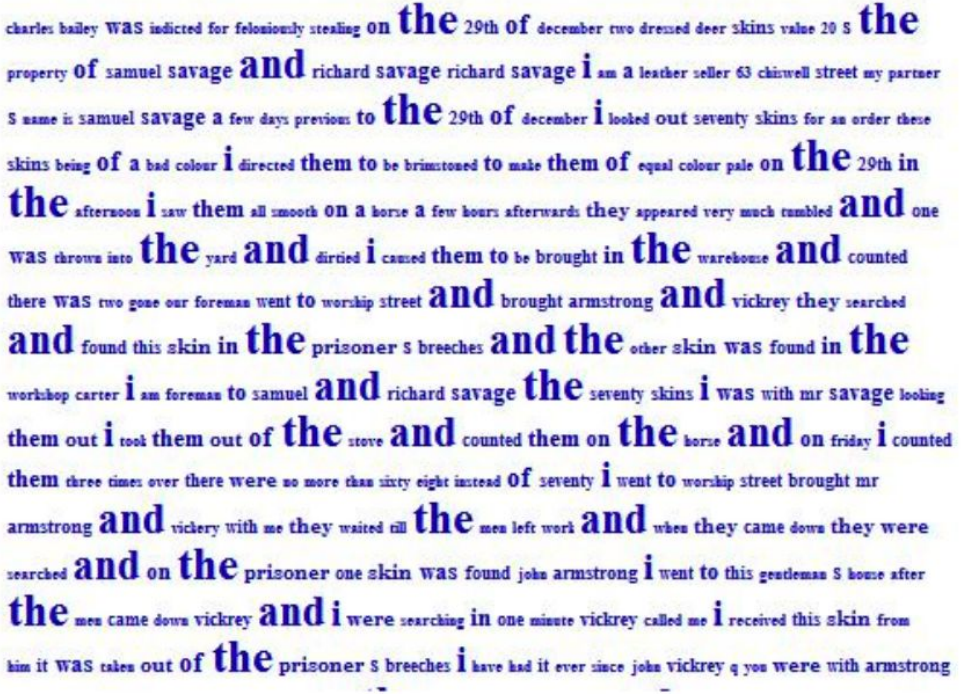
Modèle multinomial : comptage de mots

adresse	commande	envoie	je	ma	nouvelle	réclame	remercie	vous
1	0	1	2	1	1	0	1	2

tf_i (term frequency)

Représentation vectorielle de documents

- Problème du comptage brut : les mots « vides » sont les plus fréquents



Représentation vectorielle de documents

- **Solution 1** : supprimer les mots vides (blacklist)
- **Solution 2** : pénaliser les mots qui apparaissent dans beaucoup de documents

$$\mathrm{idf_i} = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|}$$

idf_i = inverse document frequency

$|D|$: nombre total de documents dans le corpus.
$|\{d_j : t_i \in d_j\}|$: nombre de document où le mot t_i apparaît.
Représentation très utilisée : $tf_i * idf_i$

Représentation vectorielle de documents

Comment représenter les similarités de termes ?

L'altitude du Mont Blanc est 4 810 mètres.
La hauteur du Mont Blanc est de 4 810 mètres.

Dans une représentation vectorielle classique :

```
Distance (altitude, hauteur) = Distance (altitude, lavabo)
```

Aucune prise en compte de la proximité sémantique !

Représentation vectorielle de documents

Comment apprendre le sens des mots ?

1. • Approches par dictionnaires, ontologies
2. • Approches par corpus

Firth (1957): *”You shall know a word by the company it keeps!”*

→ **Apprentissage du sens d’un mot par ses contextes d’usage**

John Rupert Firth (1890-1960) était un linguiste anglais et un chercheur de premier plan en linguistique britannique dans les années 1950.

Représentation vectorielle de documents

“Tesgüino” ?

1. lac finlandais

2. boisson mexicaine

3. manga japonais

Nida, E. A. 1975. Componential analysis of meaning: An introduction to semantic structures.

Représentation vectorielle de documents

Tesgüino ?

Une bouteille de tesgüino est sur la table.

Le tesgüino est produit en Sierra Madre occidentale au Mexique.

Boire du tesgüino rend ivre.

On fabrique le tesgüino à partir de maïs.

Représentation du sens d'un mot par ses contextes d'usage

Nida, E. A. 1975. Componential analysis of meaning: An introduction to semantic structures.

Représentation vectorielle de documents

Matrice terme-document :
représenter les mots par les documents dans lesquels ils apparaissent

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

c1: *Human* machine *interface* for ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management *system*
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user* perceived *response time* to error measurement

m1: The generation of random, binary, ordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

Représentation vectorielle de documents

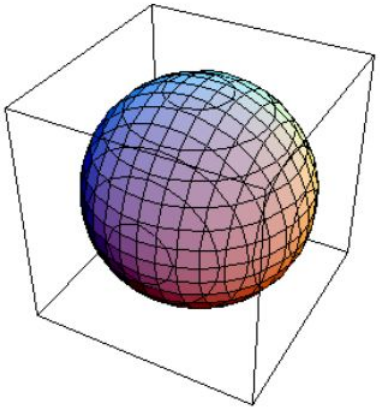
Limite : il faudrait beaucoup de documents pour bien représenter les mots

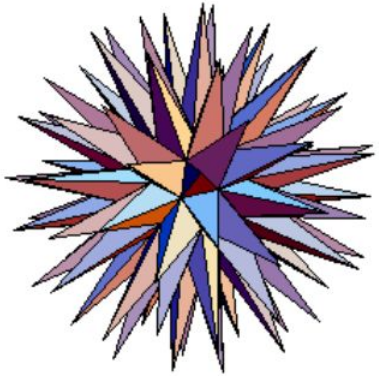
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
minors	0	0	0	0	0	0	0	1	1

- *human* et *user* ne partagent aucun contexte (document)
- *human* et *minors* non plus
- `distance (human, user) > distance (human, minors)`

Malédiction de la dimensionalité

- Dans un espace en haute dimension, tous les points sont loins les uns des autres.
- Le nombre de données nécessaires pour couvrir l'espace augmente de manière exponentielle.

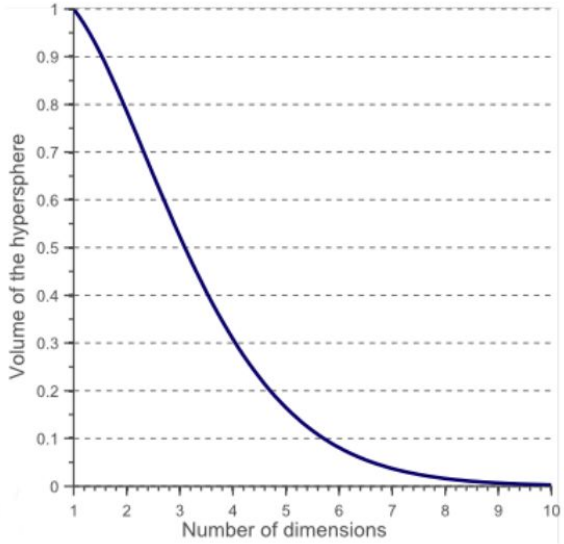




Malédiction de la dimensionalité

- Exemple : volume d'une sphère de rayon 0.5 inscrite dans un cube de coté 1

- dimension 2 :
  V cube = 1 V sphère = π/4
- dimension 3 :
  V cube = 1 V sphère = π/6
- dimension d :
  V cube = 1 V sphère = $\frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}0.5^d.$



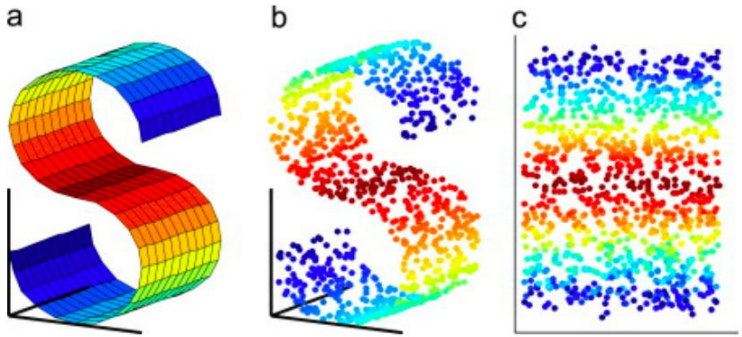
Malédiction de la dimensionalité

- Lorsque la dimension augmente, le volume de la sphère devient négligeable par rapport au volume du cube : tous les points de l'espace sont dans les coins, éloignés les uns des autres.

- Plus la dimension augmente, plus il faut de points pour couvrir l'espace et estimer les modèles

Réduction de dimension

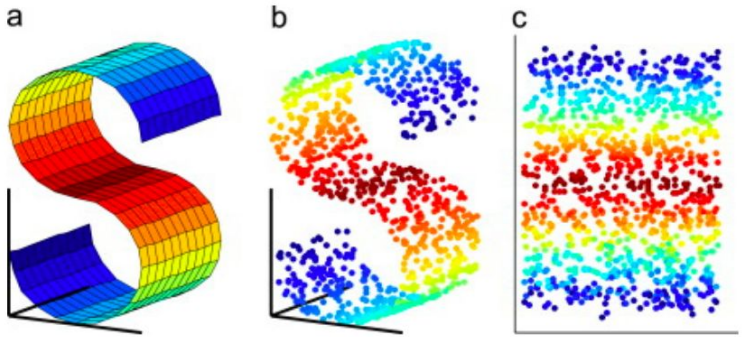
Intuition : en réalité, les points n'occupent pas uniformément tout l'espace



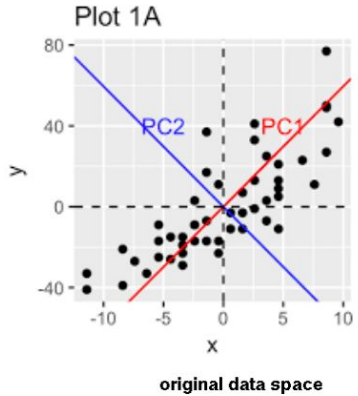
- les points sont représentés en 3 dimensions (Figures a. et b.)
- mais en réalité, ils sont disposés sur un sous-espace de dimension 2 (Figure c.)

Réduction de dimension

On peut donc conserver les relations entre les points tout en réduisant la dimension de l'espace de représentation

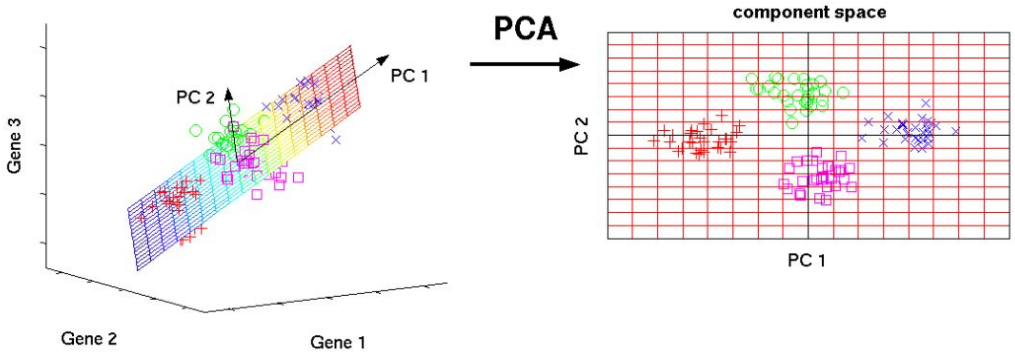


Réduction de dimension



Analyse en composantes principales (PCA)

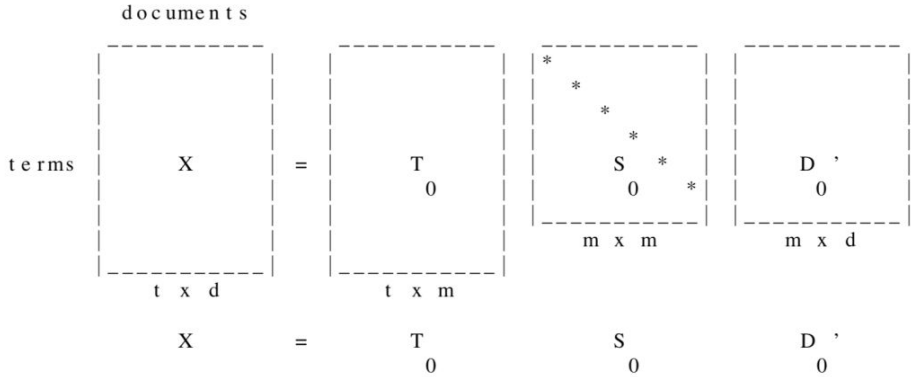
- Recherche des axes des composantes principales
- Sélection des composantes
- Projection des données sur les axes des composantes



Réduction de dimension

Latent semantic indexing/analysis (LSI/LSI)

- **Singular Value Decomposition (SVD)** : extension de l'ACP aux matrices non carrés
- **Latent Semantic Indexing (LSI)** : SVD appliqué à la matrice termes-documents

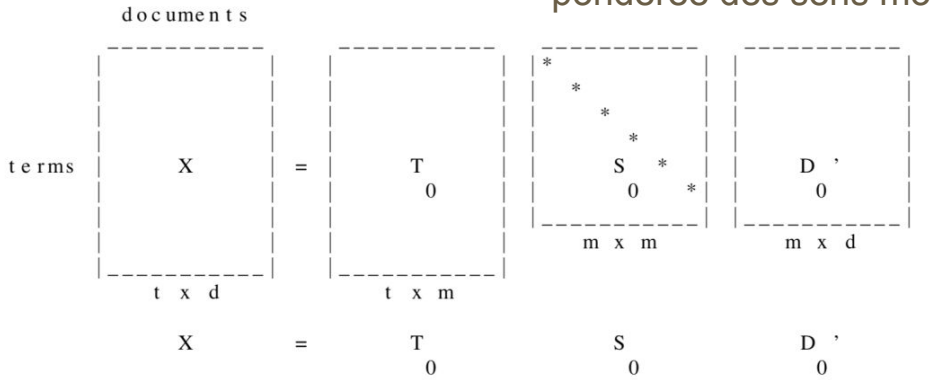


Deerwester et al., Indexing by latent semantic analysis, 1990

Réduction de dimension

Latent semantic indexing/analysis (LSI/LSI)

- Approximation de rang inférieur : k < m (rang de X, k ≈ 300)
- LSA/LSI représente le sens d'un mot comme une moyenne pondérée du sens des documents dans lesquels il apparaît et en même temps le sens d'un document comme une moyenne pondérée des sens mots qu'il contient.



Deerwester et al., Indexing by latent semantic analysis, 1990

Réduction de dimension

Latent semantic indexing/analysis (LSI)

Limites de LSI :

- Les mots les plus fréquents ont un poids très important dans la matrice de co-occurrence
- LSI modélise la relation statistiques des mots et des documents : difficulté de passage à l'échelle si le nombre de mots ou documents augmente
- LSI ne prend en compte ni l'ordre des mots, ni leur relations syntaxiques ou logiques, ni la morphologie

Solutions proposées :

- Normalisation de la matrice basée sur l'entropie ou la corrélation (COALS, Rohde et al., 2006)
- Pointwise mutual information (PPMI, Bullinaria and Levy, 2007)
- ...

Word Embeddings: GloVe (Global Vectors for Word Representation)

- Prise en compte des co-occurrences des mots pour la création de représentations vectorielles
- **Objectif** : trouver une représentation vectorielle qui conserve les ratios de fréquence de co-occurrence

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

$$P_{ij} = P(j|i) = \frac{X_{ij}}{Xi}$$

Word Embeddings: GloVe (Global Vectors for Word Representation)

- **Apprentissage** : minimisation de `J`

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

- `V` : taille du vocabulaire
- `f` : fonction de pondération
- b_i : permet de prendre en compte les différences de fréquence X_i
- W_i : représentation vectorielle

Recherche des représentation vectorielles des mots qui approchent le mieux $\log(X_{ij})$

- Optimisation par descente de gradient

Word Embeddings:

A Neural Probabilistic Language Model
Bengio et al., 2003

Modélisation statistique de la langue
`(running, walking)`, we could naturally generalize (i.e. transfer probability mass) from

The cat is walking in the bedroom

to

A dog was running in a room

and likewise to

The cat is running in a room
A dog is walking in a bedroom
The dog was walking in the room
...

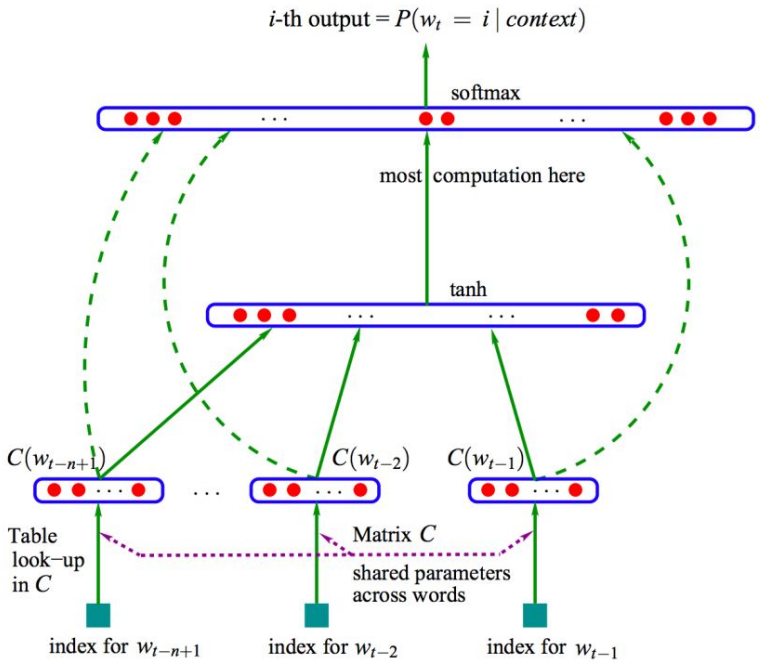
1.1 Fighting the Curse of Dimensionality with Distributed Representations

In a nutshell, the idea of the proposed approach can be summarized as follows:

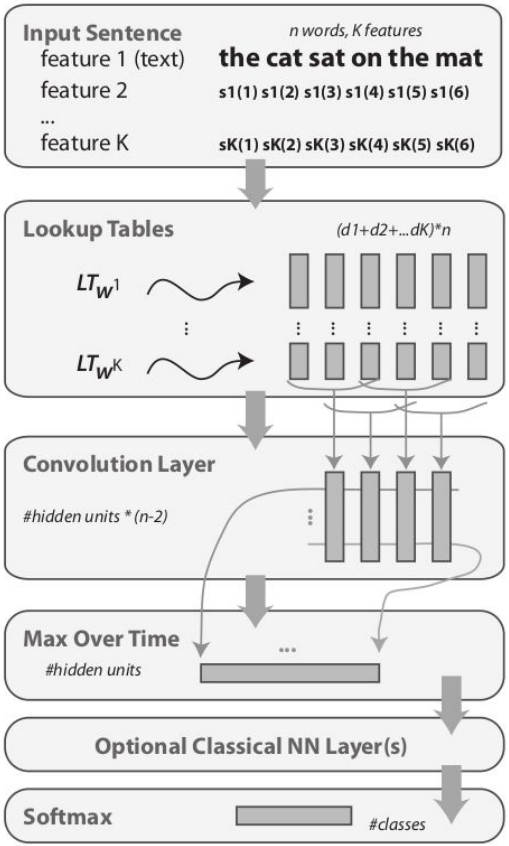
1. associate with each word in the vocabulary a distributed *word feature vector* (a real-valued vector in $\mathbb{R}^m$),
2. express the joint *probability function* of word sequences in terms of the feature vectors of these words in the sequence, and
3. learn simultaneously the *word feature vectors* and the parameters of that *probability function*.

Word Embeddings:

A Neural Probabilistic Language Model
Bengio et al., 2003



Word Embeddings:



A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning, ICML 2008

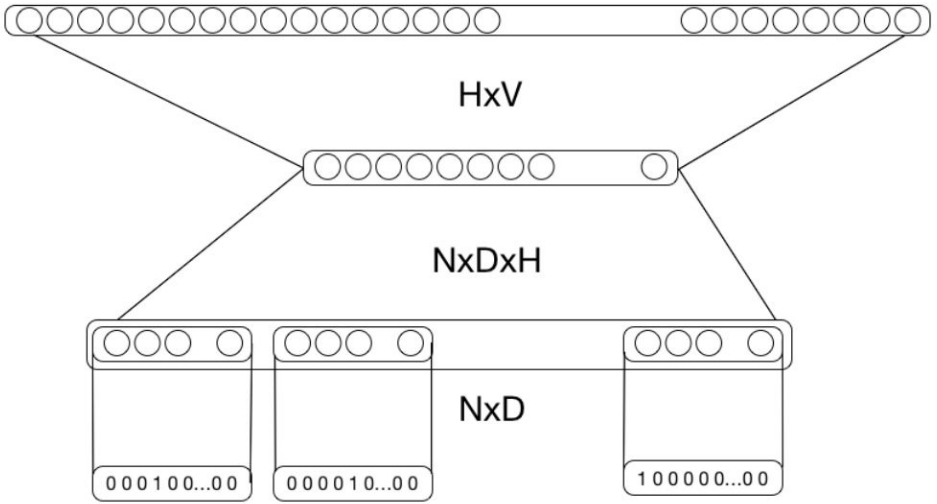
Ronan Collobert et Jason Weston

- Apprentissage des représentations (deep learning versus shallow features)
- Entrainement end-to-end versus features engineering + classifieur
- Pré-entraînement non supervisé (modèle de langue)
- Apprentissage supervisé mutlitâche

2018 International Conference on Machine Learning (ICML) “Test of Time Award”

Word Embeddings:

A Neural Probabilistic Language Model
Bengio et al., 2003



V = 100 000
D = 50 à 200
H = 500 à 1000

- HxV est énorme mais il existe des techniques d'accélération
- NxDxH reste problématique

Word Embeddings: Word2Vec

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

1 Introduction

Many current NLP systems and techniques treat words as atomic units - there is no notion of similarity between words, as these are represented as indices in a vocabulary. This choice has several good reasons - simplicity, robustness and the observation that simple models trained on huge amounts of data outperform complex systems trained on less data. An example is the popular N-gram model used for statistical language modeling - today, it is possible to train N-grams on virtually all available data (trillions of words [3]).

However, the simple techniques are at their limits in many tasks. For example, the amount of relevant in-domain data for automatic speech recognition is limited - the performance is usually dominated by the size of high quality transcribed speech data (often just millions of words). In machine translation, the existing corpora for many languages contain only a few billions of words or less. Thus, there are situations where simple scaling up of the basic techniques will not result in any significant progress, and we have to focus on more advanced techniques.

With progress of machine learning techniques in recent years, it has become possible to train more complex models on much larger data set, and they typically outperform the simple models. Probably the most successful concept is to use distributed representations of words [10]. For example, neural network based language models significantly outperform N-gram models [1, 27, 17].

1.1 Goals of the Paper

The main goal of this paper is to introduce techniques that can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary. As far as we know, none of the previously proposed architectures has been successfully trained on more

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Kai Chen
Greg Corrado
Jeffrey Dean

Proceedings of Workshop at ICLR, 2013.

Word Embeddings: Word2Vec

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

1 Introduction

Many current NLP systems and techniques treat words as atomic units - there is no notion of similarity between words, as these are represented as indices in a vocabulary. This choice has several good reasons - simplicity, robustness and the observation that simple models trained on huge amounts of data outperform complex systems trained on less data. An example is the popular N-gram model used for statistical language modeling - today, it is possible to train N-grams on virtually all available data (trillions of words [3]).

However, the simple techniques are at their limits in many tasks. For example, the amount of relevant in-domain data for automatic speech recognition is limited - the performance is usually dominated by the size of high quality transcribed speech data (often just millions of words). In machine translation, the existing corpora for many languages contain only a few billions of words or less. Thus, there are situations where simple scaling up of the basic techniques will not result in any significant progress, and we have to focus on more advanced techniques.

With progress of machine learning techniques in recent years, it has become possible to train more complex models on much larger data set, and they typically outperform the simple models. Probably the most successful concept is to use distributed representations of words [10]. For example, neural network based language models significantly outperform N-gram models [1, 27, 17].

1.1 Goals of the Paper

The main goal of this paper is to introduce techniques that can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary. As far as we know, none of the previously proposed architectures has been successfully trained on more

Objectif : étant donné un mot w_t dans un corpus de taille T, prédire les mots w_c qui peuvent apparaître dans son contexte :

$$p(w_c \mid w_t) = \frac{e^{s(w_t,\, w_c)}}{\sum_{j=1}^{W} e^{s(w_t,\, j)}}$$

Si s est une fonction de similarité entre mots, la probabilité d'un mot w_c conditionnellement à un autre mot w_t peut être calculée par :

$$\sum_{t=1}^{T} \sum_{c \in \mathcal{C}_t} \log p(w_c \mid w_t)$$

Word Embeddings: Word2Vec

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

1 Introduction

Many current NLP systems and techniques treat words as atomic units - there is no notion of similarity between words, as these are represented as indices in a vocabulary. This choice has several good reasons - simplicity, robustness and the observation that simple models trained on huge amounts of data outperform complex systems trained on less data. An example is the popular N-gram model used for statistical language modeling - today, it is possible to train N-grams on virtually all available data (trillions of words [3]).

However, the simple techniques are at their limits in many tasks. For example, the amount of relevant in-domain data for automatic speech recognition is limited - the performance is usually dominated by the size of high quality transcribed speech data (often just millions of words). In machine translation, the existing corpora for many languages contain only a few billions of words or less. Thus, there are situations where simple scaling up of the basic techniques will not result in any significant progress, and we have to focus on more advanced techniques.

With progress of machine learning techniques in recent years, it has become possible to train more complex models on much larger data set, and they typically outperform the simple models. Probably the most successful concept is to use distributed representations of words [10]. For example, neural network based language models significantly outperform N-gram models [1, 27, 17].

1.1 Goals of the Paper

The main goal of this paper is to introduce techniques that can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary. As far as we know, none of the previously proposed architectures has been successfully trained on more

Le **problème** avec cette fonction objectif est le terme de normalisation : il nécessite de calculer s sur tous les mots

$$p(w_c \mid w_t) = \frac{e^{s(w_t,\, w_c)}}{\sum_{j=1}^{W} e^{s(w_t,\, j)}}$$

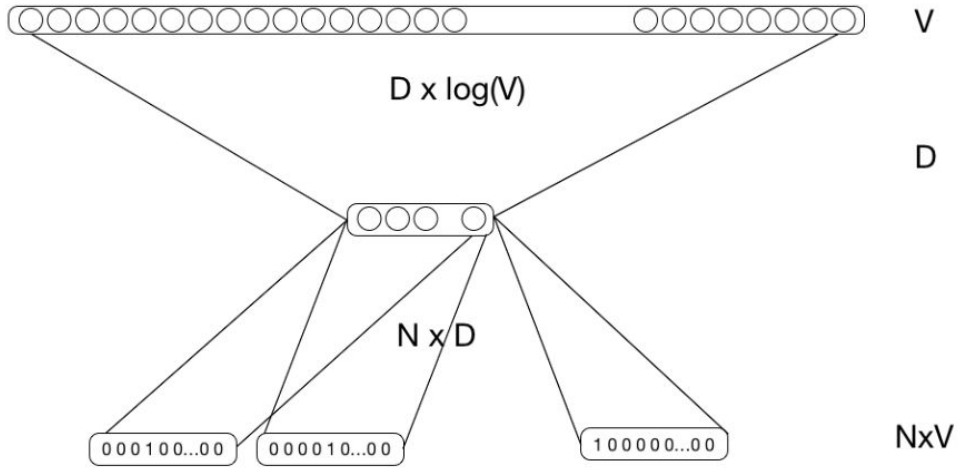
Word2Vec remplace donc cette fonction objectif par une tâche de classification : prédire si oui ou non un mot apparaît dans un contexte d'un mot donné.

$$\log\left(1 + e^{-s(w_t,\, w_c)}\right) + \sum_{n \in \mathcal{N}_{t,c}} \log\left(1 + e^{s(w_t,\, n)}\right)$$

Word Embeddings: Word2Vec

Approche 1 : Continuous Bag of word (CBOW)

- Supprimer la couche cachée
- Sommer les contextes

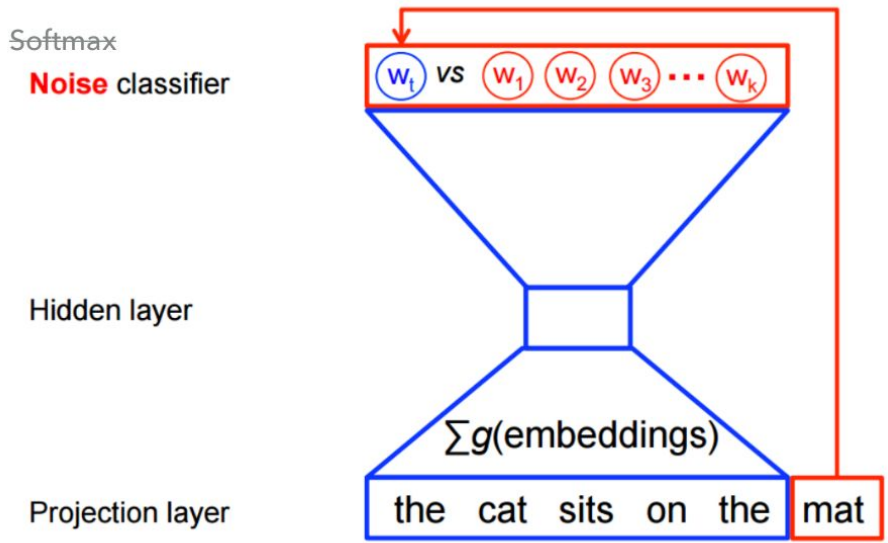


Prédiction du mot courant en fonction du contexte droit et gauche

Word Embeddings: Word2Vec

Approche 1 : Continuous Bag of word (CBOW); Paramètres du modèles :

- Exemples négatifs : nombre ?
- Embedding : taille ?
- Contexte : gauche/droite, taille ?

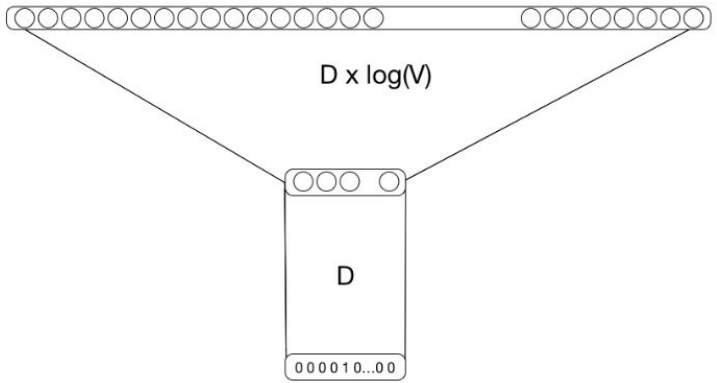


Prédiction du mot courant en fonction du contexte droit et gauche

Word Embeddings: Word2Vec

Approche 2 : Continuous Skip-Gram

- Supprimer la couche cachée
- Prédire chaque mot du contexte à partir du mot courant



Prédiction pour C contextes

Word Embeddings: Word2Vec

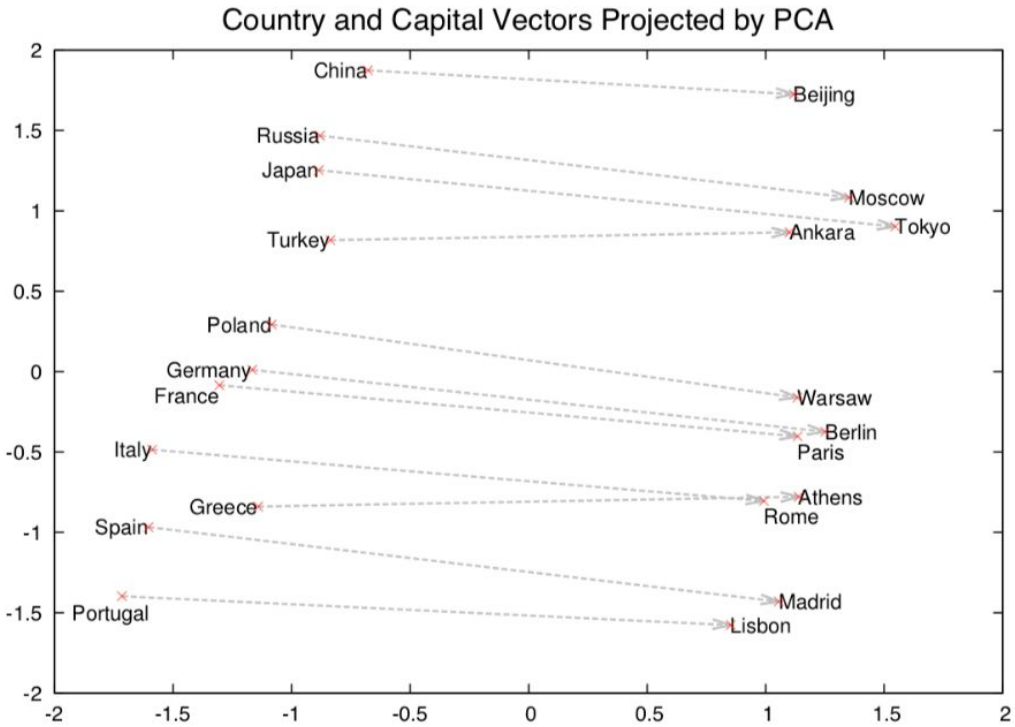
Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Ilya Sutskever
Kai Chen
Greg Corrado
Jeffrey Dean

NIPS, 2013

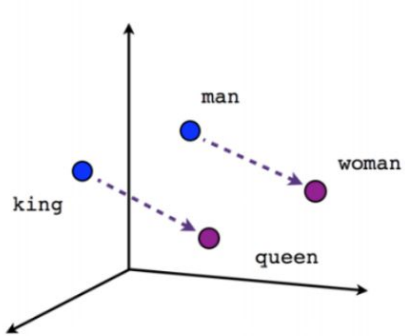
Word Embeddings: Word2Vec

Relations sémantiques et géométriques

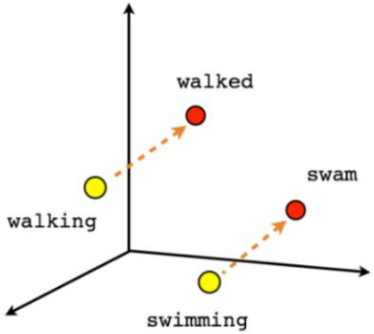


Word Embeddings: Word2Vec

Relations sémantiques et géométriques



Male-Female



Verb tense

Word Embeddings: Word2Vec

Arithmétique vectorielle et sémantique

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Word Embeddings: Word2Vec

Limites :

- Pas de représentation pour les mots inconnus :
  mots rares, nom propres, néologismes, fautes d'orthographe, argot, jargon, erreur de reconnaissance (OCR, parole)
- Pas de paramètres partagés pour les différentes formes fléchies d'un mot

```
mange / mangerai
cheval /chevaux
```

Word Embeddings: FastText

Enriching Word Vectors with Subword Information

Piotr Bojanowski* and Edouard Grave* and Armand Joulin and Tomas Mikolov
Facebook AI Research
{bojanowski,egrave,ajoulin,tmikolov}@fb.com

Abstract

Continuous word representations, trained on large unlabeled corpora are useful for many natural language processing tasks. Popular models that learn such representations ignore the morphology of words, by assigning a distinct vector to each word. This is a limitation, especially for languages with large vocabularies and many rare words. In this paper, we propose a new approach based on the skipgram model, where each word is represented as a bag of character n-grams. A vector representation is associated to each character n-gram; words being represented as the sum of these representations. Our method is fast, allowing to train models on large corpora quickly and allows us to compute word representations for words that did not appear in the training data. We evaluate our word representations on nine different languages, both on word similarity and analogy tasks. By comparing to recently proposed morphological word representations, we show that our vectors achieve state-of-the-art performance on these tasks.

1 Introduction

Learning continuous representations of words has a long history in natural language processing (Rumelhart et al., 1988). These representations are typically derived from large unlabeled corpora using co-occurrence statistics (Deerwester et al., 1990; Schütze, 1992; Lund and Burgess, 1996). A large body of work, known as distributional semantics, has studied the properties of these methods (Turney et al., 2010; Baroni and Lenci, 2010). In the neural network community, Collobert and Weston (2008) proposed to learn word embeddings using a feed-forward neural network, by predicting a word based on the two words on the left and two words on the right. More recently, Mikolov et al. (2013b) proposed simple log-bilinear models to learn continuous representations of words on very large corpora efficiently.

Most of these techniques represent each word of the vocabulary by a distinct vector, without parameter sharing. In particular, they ignore the internal structure of words, which is an important limitation for morphologically rich languages, such as Turkish or Finnish. For example, in French or Spanish, most verbs have more than forty different inflected forms, while the Finnish language has fifteen cases for nouns. These languages contain many word forms that occur rarely (or not at all) in the training corpus, making it difficult to learn good word representations. Because many word formations follow rules, it is possible to improve vector representations for morphologically rich languages by using character level information.

In this paper, we propose to learn representations for character n-grams, and to represent words as the sum of the n-gram vectors. Our main contribution is to introduce an extension of the continuous skipgram model (Mikolov et al., 2013b), which takes into account subword information. We evaluate this model on nine languages exhibiting different morphologies, showing the benefit of our approach.

*The two first authors contributed equally.

Enriching Word Vectors with Subword Information

Piotr Bojanowski
Edouard Grave
Armand Joulin
Tomas Mikolov

Transactions of the Association for Computational Linguistics, 2013

Word Embeddings: FastText

« Our main contribution is to introduce an extension of the continuous skip-gram model (Mikolov et at., 2013), which takes into account subword information»

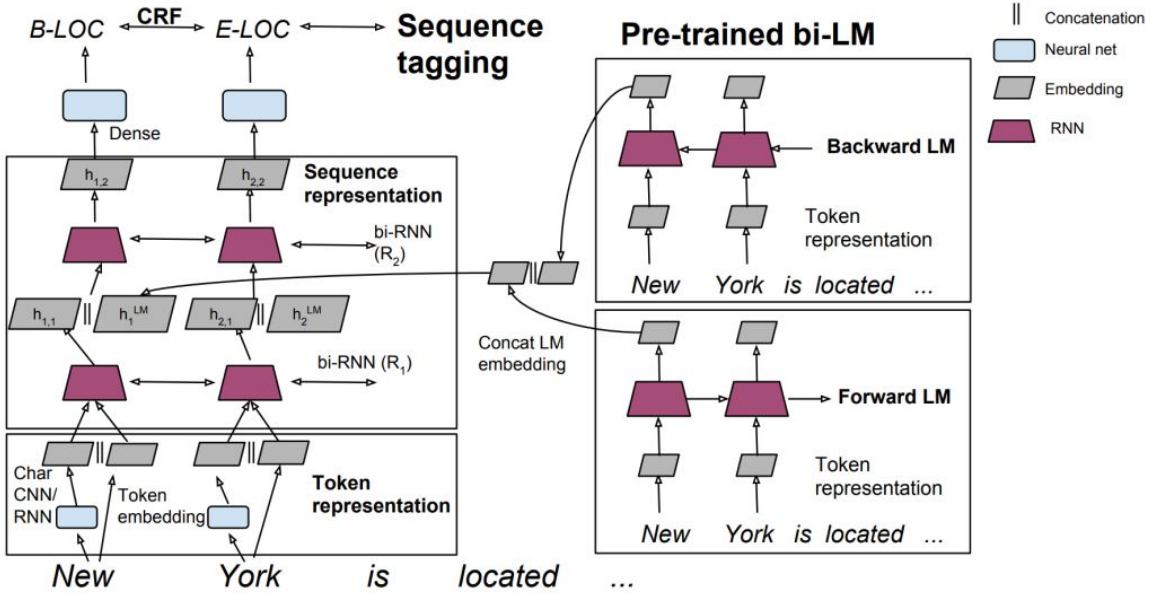
« Learn representations for character n-grams, and to represent words as the sum of the n-gram vectors.»

$$\sum_{t=1}^{T} \left[\sum_{c \in \mathcal{C}_t} \ell(s(w_t, w_c)) + \sum_{n \in \mathcal{N}_{t,c}} \ell(-s(w_t, n)) \right]$$

$$\ell : x \mapsto \log(1 + e^{-x})$$

$N_{t,c}$: ensemble d'exemples négatifs tirés du vocabulaire
C_t : mots du contexte du mot cible t

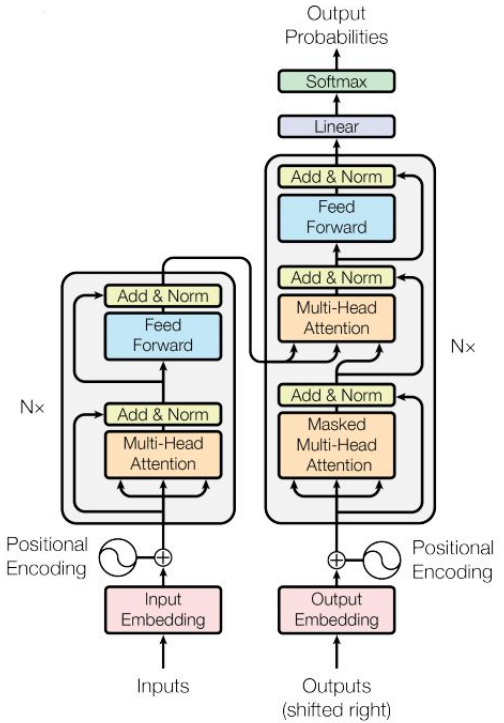
Word embeddings : modèles plus complexes



- Semi-supervised sequence tagging with bidirectional language models, Peters et al., ACL 2017
- Deep Contextualized word representation, Peters et al., NAACL 2018

Word embeddings : modèles plus complexes (+ des modèles d'attention)

Attention is all you need
Vaswani et al., NIPS 2017



→ Transformer

Word embeddings : problème de biais

Man is to computer programmer as woman is to homemaker? Debiasing word embeddings

Bolukbasi et al., NIPS 2016

Biais de position sur l'axe homme-femme

Extreme *she* occupations

1. homemaker	2. nurse	3. receptionist
4. librarian	5. socialite	6. hairdresser
7. nanny	8. bookkeeper	9. stylist
10. housekeeper	11. interior designer	12. guidance counselor

Extreme *he* occupations

1. maestro	2. skipper	3. protege
4. philosopher	5. captain	6. architect
7. financier	8. warrior	9. broadcaster
10. magician	11. figher pilot	12. boss

Biais d'analogie

Gender stereotype *she-he* analogies.

sewing-carpentry	register-nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	hairdresser-barber

Gender appropriate *she-he* analogies.

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Links

https://www.kaggle.com/competitions

Machine learning Coursera famous courses, Andrew Ng, https://www.coursera.org/learn/machine-learning

Machine learning Coursera (on youtube), Andrew Ng, https://www.youtube.com/watch?v=PPLop4L2eGk

The most famous book on deep learning: https://www.deeplearningbook.org/ (Ian Goodfellow, Yoshua Bengio and Aaron Courville)

ML and DL People

Andrew Ng, Founder and CEO of Landing AI, Founder of deeplearning.ai.
Fei-Fei Li, Professor of Computer Science at Stanford University.
Andrej Karpathy, Senior Director of Artificial Intelligence at Tesla.
Demis Hassabis, Founder and CEO of DeepMind.
Ian Goodfellow, Director of Machine Learning at Apple.
Yann LeCun, Vice President and Chief AI Scientist at Facebook.
Jeremy P. Howard, Founding Researcher at fast.ai, Distinguished Research Scientist at the University of San Francisco.
Ruslan Salakhutdinov, Associate Professor at Carnegie Mellon University, Director of AI Research at Apple.
Geoffrey Hinton, Professor of Computer Science at the University of Toronto, VP and Engineering Fellow at Google
Rana el Kaliouby, CEO and Co-Founder of Affectiva.
Daphne Koller, Founder and CEO of insitro, Co-Founder of Coursera, Adjunct Professor of Computer Science and Pathology at Stanford.
Alex Smola, Director, Amazon Web Services.