Lecture 4 - Text classification

SD-TSIA214

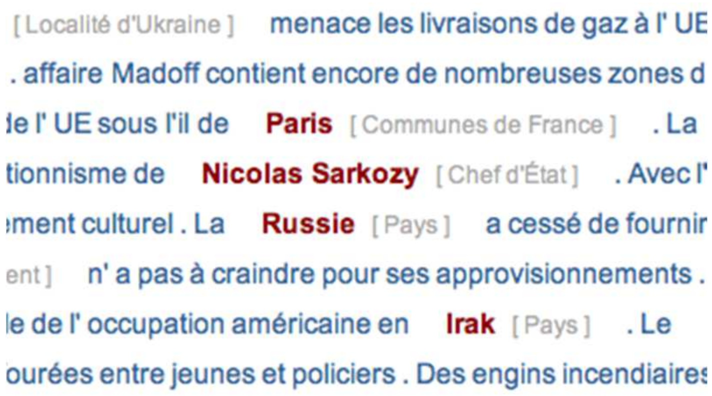
Chloé Clavel

Reminder

NLP tasks

2 kind of tasks:

- **Classify documents by themes, opinions etc...**
  - Supervised learning
    - Ex : SVM (support vector machines), Naive Bayesian
  - Unsupervised learning
    - Ex: Clustering
- **Detect particular expressions**
    - Ex: Named Entities
      - o



[Localité d'Ukraine] menace les livraisons de gaz à l' UE

. affaire Madoff contient encore de nombreuses zones d

le l' UE sous l'il de **Paris** [Communes de France] . La

tionnisme de **Nicolas Sarkozy** [Chef d'État] . Avec l'

ment culturel . La **Russie** [Pays] a cessé de fournir

ent] n' a pas à craindre pour ses approvisionnements .

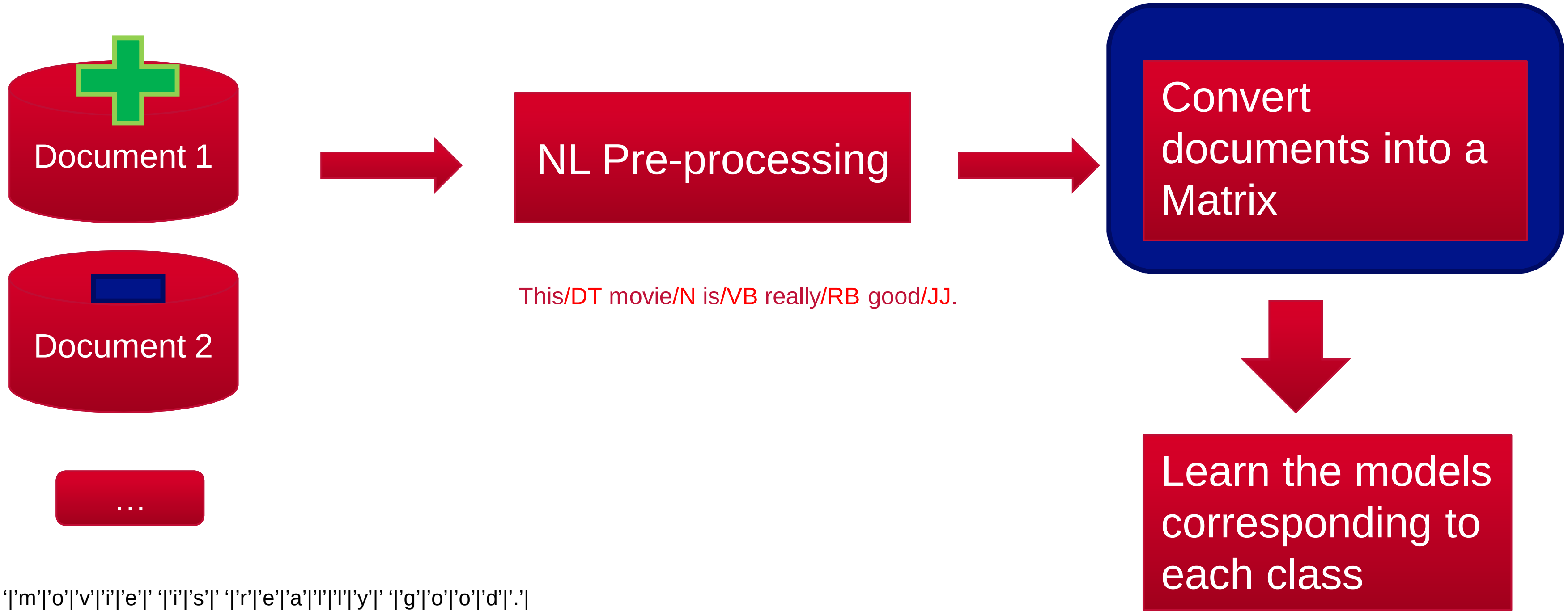
le de l' occupation américaine en **Irak** [Pays] . Le

ourées entre jeunes et policiers . Des engins incendiaires

From http://www.tal.univ-paris3.fr/plurital/travaux-2009-2010/bao-2009-2010/MarjorieSeizou-AxelCourt/webservices.html

Reminder

- Learning the classes



Document 1

Document 2

…

NL Pre-processing

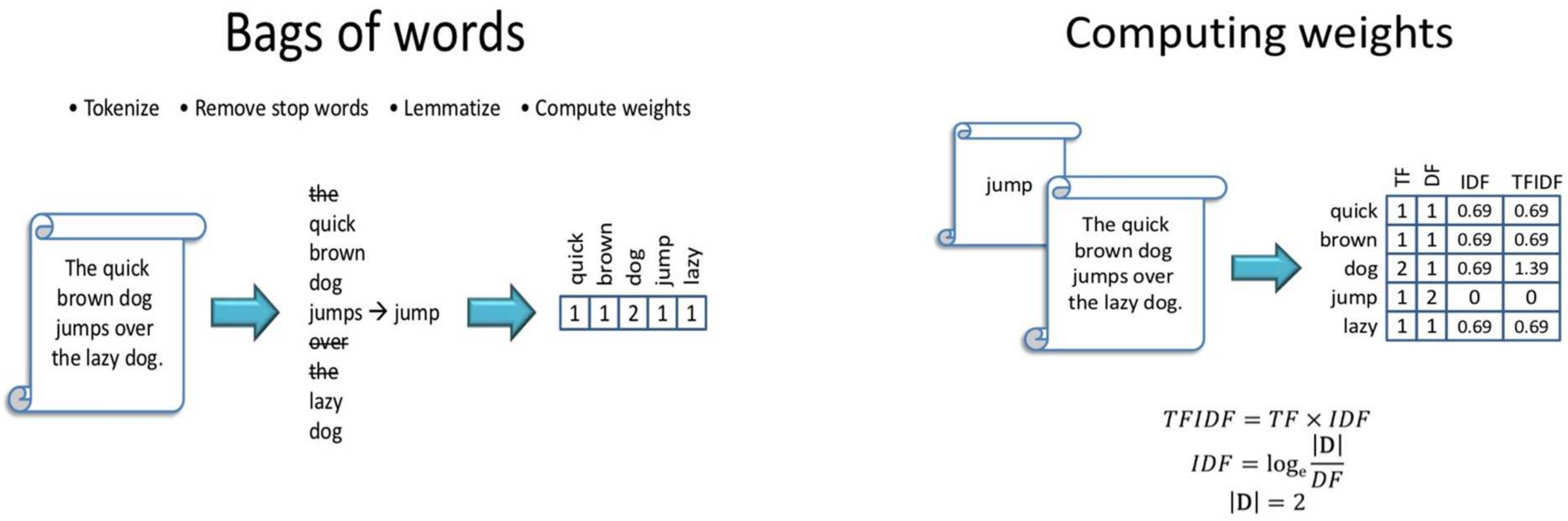
Convert documents into a Matrix

Learn the models corresponding to each class

This/DT movie/N is/VB really/RB good/JJ.

|’T’|’h’|’i’|’s’|’ ‘|’m’|’o’|’v’|’i’|’e’|’ ‘|’i’|’s’|’ ‘|’r’|’e’|’a’|’l’|’l’|’y’|’ ‘|’g’|’o’|’o’|’d’|’.’|

Reminder : Convert documents into a matrix



Bags of words

- Tokenize
- Remove stop words
- Lemmatize
- Compute weights

The quick brown dog jumps over the lazy dog.
~~the~~
quick
brown
dog
jumps → jump
~~over~~
~~the~~
lazy
dog

quick	brown	dog	jump	lazy
1	1	2	1	1

Computing weights

jump
The quick brown dog jumps over the lazy dog.

	TF	DF	IDF	TFIDF
quick	1	1	0.69	0.69
brown	1	1	0.69	0.69
dog	2	1	0.69	1.39
jump	1	2	0	0
lazy	1	1	0.69	0.69

$$TFIDF = TF \times IDF$$

$$IDF = \log_e \frac{|\mathrm{D}|}{DF}$$

$$|\mathrm{D}| = 2$$

Sparse vs. Dense representations (word2vec)

Objective of the lecture

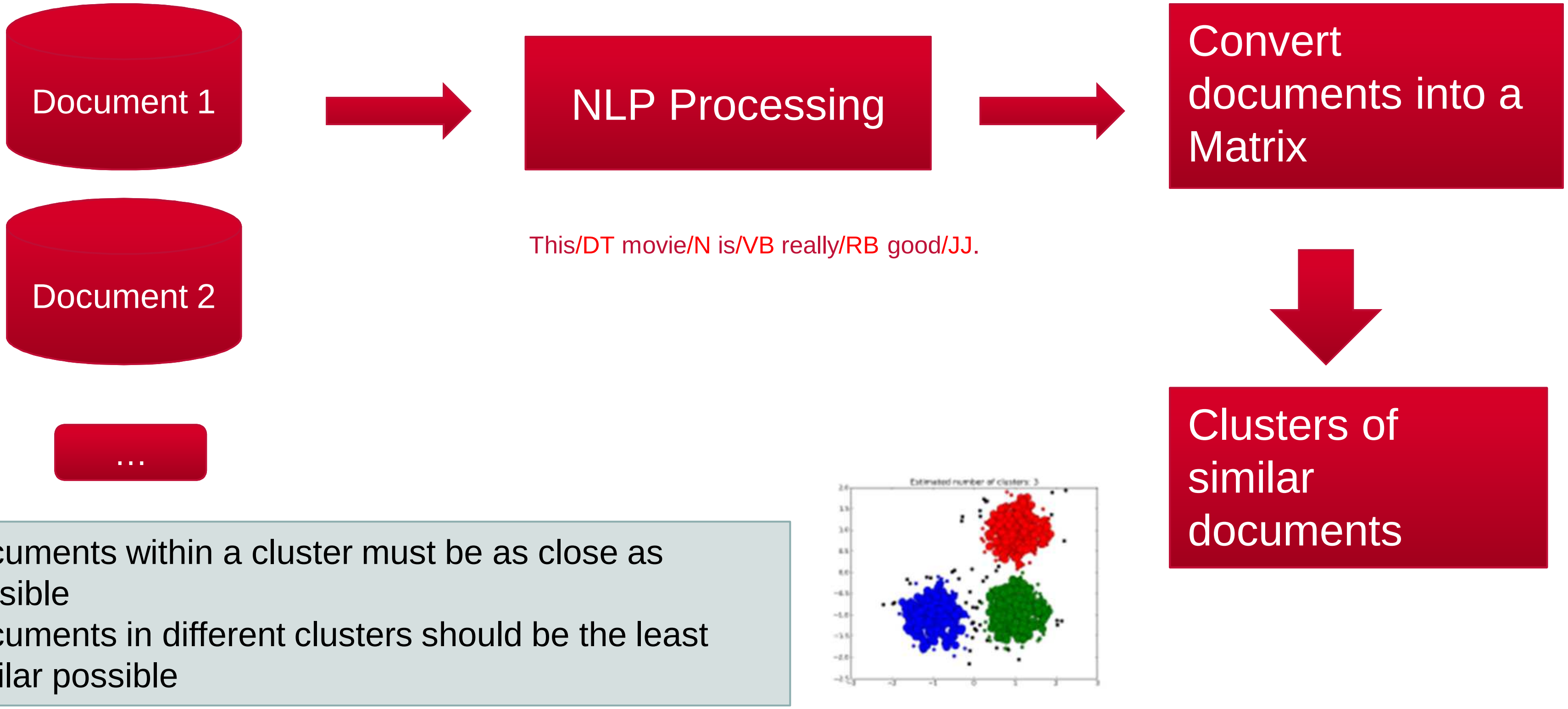
- **Get familiar with:**
  - Text Clustering
  - Supervised text classification

Clustering

Unsupervised learning

Text clustering

Unsupervised learning : no labelling based on human expertise


Document 1
Document 2
…
NLP Processing
Convert documents into a Matrix
Clusters of similar documents
Estimated number of clusters: 3

This/DT movie/N is/VB really/RB good/JJ.

Documents within a cluster must be as close as possible
Documents in different clusters should be the least similar possible

Text clustering

Unsupervised learning : no labelling based on human expertise

Principles:

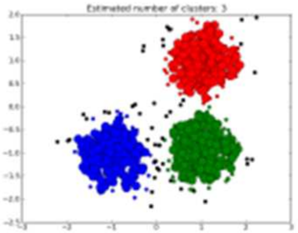
- Methods for grouping similar textual documents
- Problem of partitioning documents
- Require the definition of criteria to evaluate the quality of the partitionning

Documents within a cluster must be as close as possible
Documents in different clusters should be the least similar possible

Text clustering

Unsupervised learning : no labelling based on human expertise

- **The cluster membership is determined by :**
  - the distribution of the data
  - the make-up of the data



In this figure, it is visually clear that there are three distinct clusters of points

=> Clustering methods are algorithms that find such clusters in an unsupervised fashion

Clustering vs. classification

- **Classification is a form of supervised learning**
  - The goal is to replicate a categorical distinction that a human supervisor imposes on the data
- **Clustering is a form of unsupervised learning**
  - We have no teacher (human labeller) to guide the clustering

Text clustering

- **The different types of clustering methods**
  - Hierarchical Clustering:creates a hiearchy of clusters
    - Graphs, Trees
  - Non hierarchical methods/Flat clustering:creates a flat set of clusters without any explicit structure that would relate clusters to each other
    - k-means, ISODATA,

But not all the clustering methods are relevant for TEXT clustering

ex: hierarchical-agglomerative clustering

Key input to clustering algorithms

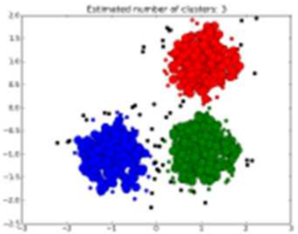
- **distance / similarity measure**
  - Will influence clustering outputs
    - Different distance measures give rise to different clustering
    - => make up your vector space model and your distance according to your clustering task:
      - Topic similarity for topic clustering
      - Language similarity for language clustering

EXAMPLE : when computing topic similarity, stop words can be safely ignored but not for language similarity

« the » and « la » are useful for langague similarity

Key input to clustering algorithms

- **distance / similarity measure**
  - Will influence clustering outputs
    - Different distance measures give rise to different clustering
    - => make up your distance according to your clustering task:
      - Topic similarity for topic clustering
      - Language similarity for language clustering
  - Some distances :
    - Euclidean distance

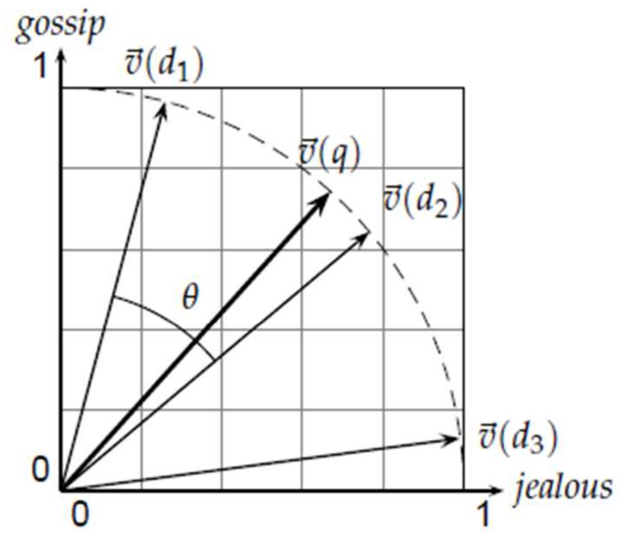


In this figure, the euclidean distance in the 2d-plane suggests three different clusters

- Distance / similarity cosine
- Distance from Jaccard

Key input to clustering algorithms

- **Cosine similarity**



$$\mathrm{sim}(d_1,d_2) = \frac{\vec{V}(d_1)\cdot\vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|},$$

▶ **Figure 6.10** Cosine similarity illustrated. $\mathrm{sim}(d_1,d_2) = \cos\theta$.

Key input to clustering algorithms

- **Distance based on Jaccard index**

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

$$J_\delta(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

Focus on flat clustering

Problem statement

- Inputs
  - a set of N documents D = {D1,…, DN}
  - A desired number of clusters K
  - An objective function that evaluates the quality of the clustering
- Outputs
  - An assignment function f: D -> {1,…,K} that minimizes/maximizes the objective function
- NB : the algo has also to identify the best K

Focus on k-means

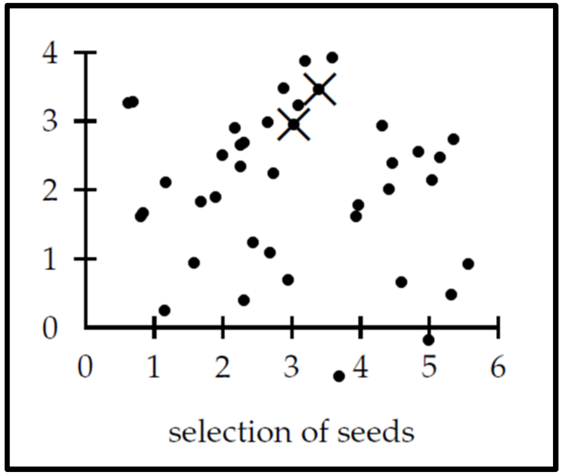
General principle

- Distance measure :
  - euclidean distance
- Objective function to minimize
  - Intra-cluster inertial : average squared Euclidean distance of documents from their cluster centers μ_k

$$\boxed{\sum_{k\in\{1,\ldots,K\}}\sum_{i\in\mathcal{C}_k}\|x_i-\mu_k\|_2^2}$$

Focus on k-means

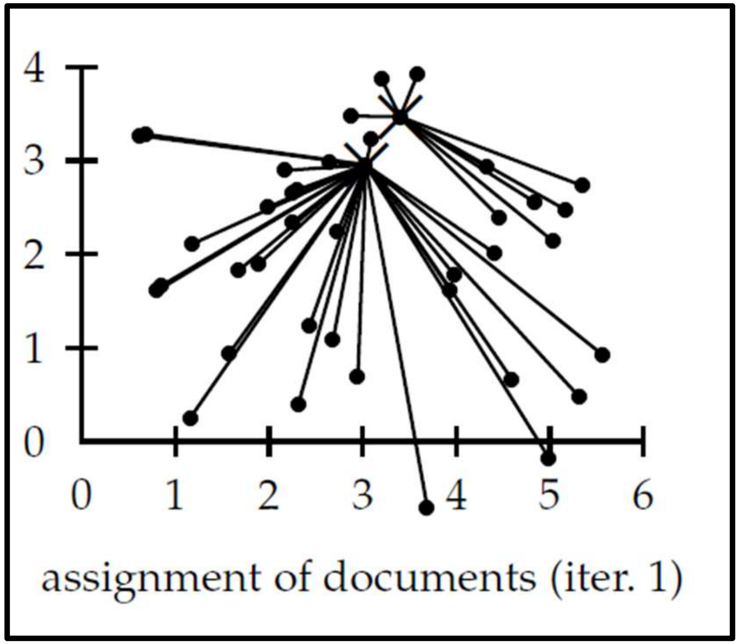
- **ALGO**
  - INPUT: D set of N documents = points of a multi-dimensional space, provided with a distance d.
  - Initialization:
    - Select randomly K documents in D
      - to define the K initial cluster centers = the *seeds*



From IR book

Focus on k-means

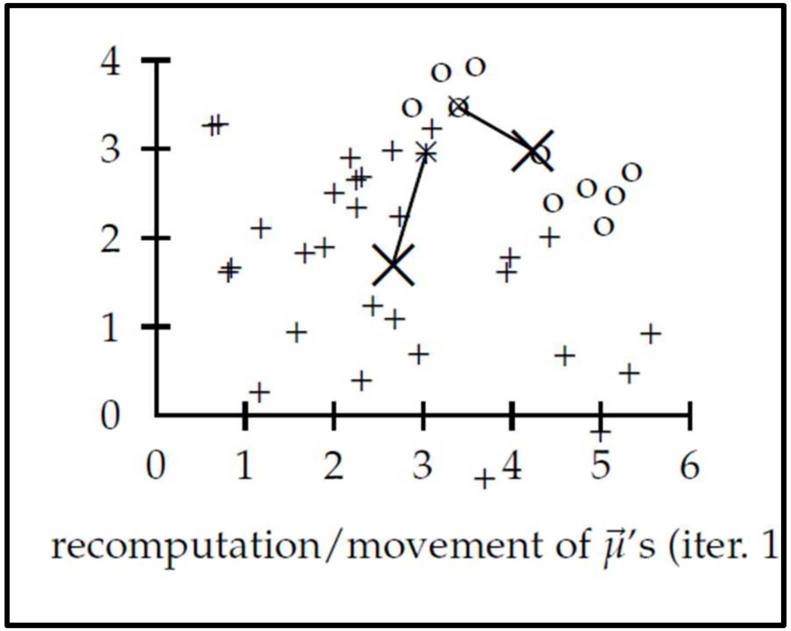
- ith Iteration
  - Assign the N documents to the cluster with the closest cluster center (assignment function f_i: D -> {1,…,K})



Focus on k-means

- ith Iteration
  - Calculation of the centroid of each cluster as the barycenter of the current members of the cluster:

$$\mu_k = \frac{1}{|\mathcal{C}_k|} \sum_{i \in \mathcal{C}_k} x_i, \quad \forall k \in \{1, \dots, K\}$$



recomputation/movement of $\vec{\mu}$'s (iter. 1

Focus on k-means

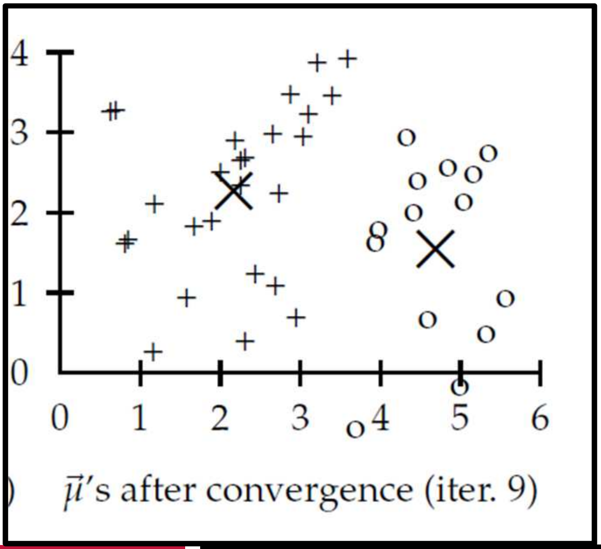
- calculation of intra-class inertia

$$\sum_{k\in\{1,\ldots,K\}} \sum_{i\in\mathcal{C}_k} \|x_i - \mu_k\|_2^2$$

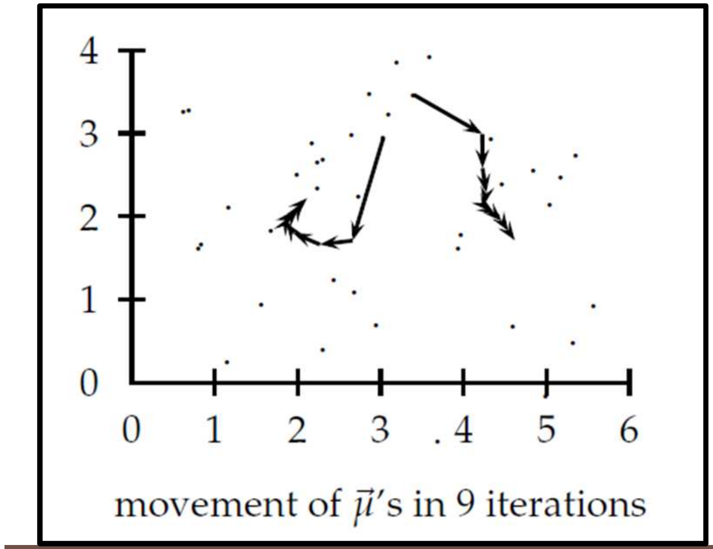
- i-> i+1

Focus on k-means

- Termination options
  - Stop after a fixed number of iterations
  - Stop when the assignment function or centroids do not change between iterations
  - Stop when inertia falls below a threshold
  - Stop when inertia converges (the decrease of inertia falls below a small threshold)



$\vec{\mu}$'s after convergence (iter. 9)



movement of $\vec{\mu}$'s in 9 iterations

Focus on flat clustering

Mix of Multinomial Laws

General principle :

- Looking for a description of classes / clusters by:
  - By their probability density:
- we know :
  - the shapes of probability densities (ex: mixture of multinomial laws)
- we look for :
  - the parameters of the densities (ex: parameters of the gaussians)
  - … that maximize a criterion of grouping documents according to these classes

Flat clustering

Mix of Multinomial Laws

- initialization:
  - consider a set of K clusters and initialize the parameters of the law associated with each cluster
  - Assign each document to a cluster based on the probability of the document to belong to a class (most likely class) -> initial partitioning
- iteration:
  - Recalculate model parameters based on current partitioning clusters
  - Redistribute documents in clusters from this new template.

Text classification

Rule-based and supervised learning

EXAMPLE : Is this e-mail spam?

Good Day,
My name is Dr William Monroe, a staff in the Private Clients Section of a well-known bank, here in London, England. One of our accounts, with holding balance of £15,000,000 has been dormant and last operated three years ago. From my investigations, the owner of the said account, John Shumejda died on the 4th of January 2002 in a plane crash.

I have decided to find a reliable foreign partner to deal with. I therefore propose to do business with you, standing in as the next of kin of these funds from the deceased. This transaction is totally free of risk and troubles as the fund is legitimate and does not originate from drug, money laundry or terrorism. On your interest, let me hear from you URGENTLY.

Best Regards,

Dr William Monroe Financial Analysis and Remittance Manager

Classification Tasks - example

- Is this e-mail spam?
- Positive or negative review?
- What is the topic of this article?
- Predict hashtags for a tweet
- Age/gender identification
- Language identification
- Sentiment analysis

Types of Classification Tasks

- Binary classification (true, false)
- Multi-class classification (politics, sports, gossip)
- Multi-label classification (#party #FRIDAY #fail)
- Clustering (labels unknown)

Classification Methods

- **By hand**
  - E.g. Yahoo in the old days
    - ✔ Very accurate and consistent assuming experts
    - ✘ Super slow, expensive, does not scale
- **Rule-based**
  - E.g. Advanced search criteria (”site:ox.ac.uk”)
    - ✔ Accuracy high if rule is suitable
    - ✘ Need to manually build and maintain rule-based system.
- **Machine learning**
    - ✔ Scales well, can be very accurate, automatic
    - ✘ Requires classified training data. Sometimes a lot!

Rule-based methods

- **Objectif :**
  - décrire l'information à extraire pour un métier, un domaine spécifique ou une thématique en modélisant l'information sous forme de lexiques/ontologies et patrons/règles linguistiques/grammaires/automates.

« manque de qualité de service » → Concept **INSATISFACTION**

« il n'y a vraiment pas eu de contact », …

Rule-based methods

- **Modélisation sémantique :**
  - Utilisation de lexiques et de règles
  - Règles qui répertorient toutes les formulations possibles d'une même information
    - – langage d'expressions régulières
      - Appel de lemmes : ex. « *avoir* »
      - Appel de catégories grammaticales : « *#PREP_DE* » « *#NEG* »
      - Appel de lexiques prédéfinis: « *~services-lex* »

« manque de qualité de service » → Concept **INSATISFACTION**

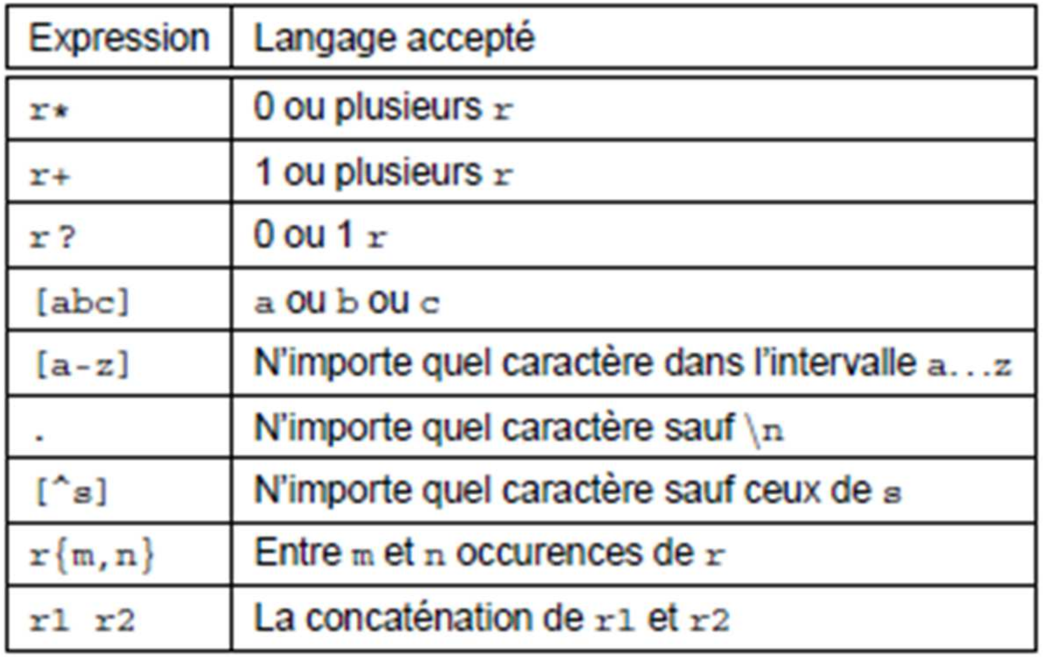
« il n'y a vraiment pas eu de contact », …

(manque|~negation-patt|(il/#NEG/y/avoir/~negation-patt))/(#PREP_DE)?/ (conseil|contact|~services-lex)*

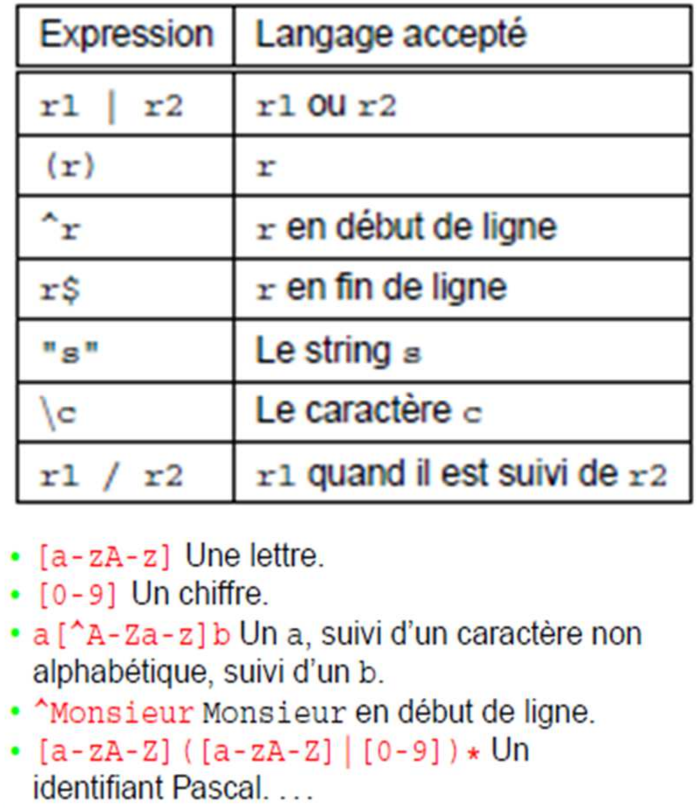
* Exemple : syntaxe de l'outil TEMIS et exemple d'utilisation à EDF pour des analyses des opinions des clients

Rule-based methods using regular expressions

- Syntaxe courante (Unix, perl, etc.)



Expression	Langage accepté
r*	0 ou plusieurs r
r+	1 ou plusieurs r
r ?	0 ou 1 r
[abc]	a ou b ou c
[a-z]	N'importe quel caractère dans l'intervalle a...z
.	N'importe quel caractère sauf \n
[^s]	N'importe quel caractère sauf ceux de s
r{m,n}	Entre m et n occurences de r
r1 r2	La concaténation de r1 et r2



Expression	Langage accepté
r1 | r2	r1 ou r2
(r)	r
^r	r en début de ligne
r$	r en fin de ligne
"s"	Le string s
\c	Le caractère c
r1 / r2	r1 quand il est suivi de r2

- [a-zA-z] Une lettre.
- [0-9] Un chiffre.
- a[^A-Za-z]b Un a, suivi d'un caractère non alphabétique, suivi d'un b.
- ^Monsieur Monsieur en début de ligne.
- [a-zA-Z]([a-zA-Z]|[0-9])* Un identifiant Pascal. ...

Tiré de http://www.ulb.ac.be/di/ssd/ggeeraer/lg/extexpreg_print.pdf

Rule-based methods using regular expressions - Practice

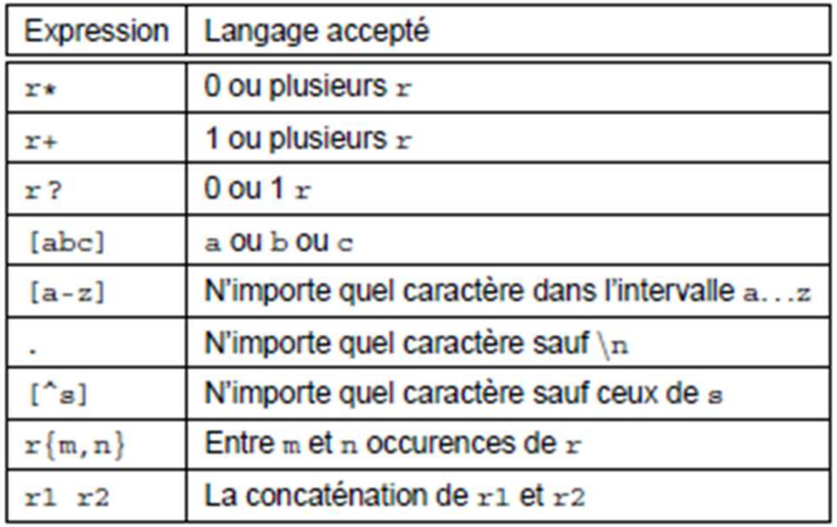
- **Donnez l'expression régulière acceptant l'ensemble des phrases «correctes» selon les critères suivants :**
  - Le premier mot de la phrase a une majuscule ;
  - la phrase se termine par un point ;
  - la phrase est composée d'un ou plusieurs mots (caractères a...z et A...Z), séparés par un espace ;

Test des regexp :
http://www.regexplanet.com/advanced/java/index.html
https://regex101.com/

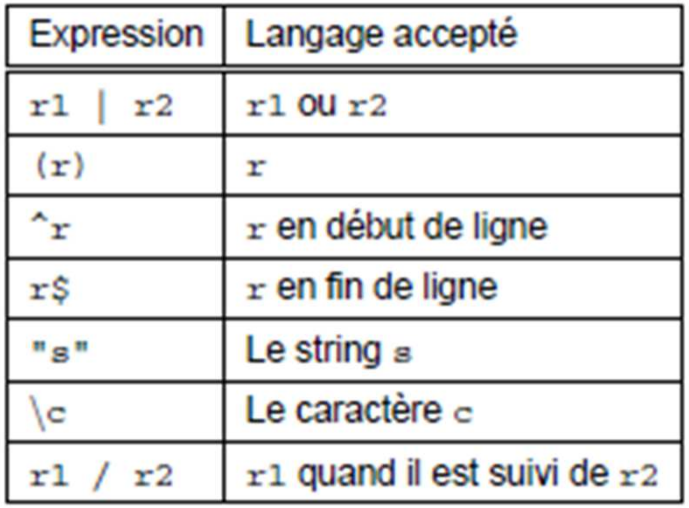
Tiré de http://www.ulb.ac.be/di/ssd/ggeeraer/lg/extexpreg_print.pdf

Rule-based methods using regular expressions - Practice

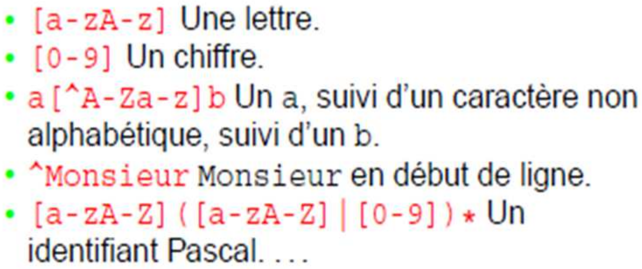
Donnez l’expression régulière acceptant l’ensemble des phrases «correctes» selon les critères suivants :
Le premier mot de la phrase a une majuscule ;
la phrase se termine par un point ;
la phrase est composée d’un ou plusieurs mots (caractères a...z et A...Z), séparés par un espace



Expression	Langage accepté
r*	0 ou plusieurs r
r+	1 ou plusieurs r
r ?	0 ou 1 r
[abc]	a ou b ou c
[a-z]	N’importe quel caractère dans l’intervalle a...z
.	N’importe quel caractère sauf \n
[^s]	N’importe quel caractère sauf ceux de s
r{m,n}	Entre m et n occurences de r
r1 r2	La concaténation de r1 et r2



Expression	Langage accepté
r1 | r2	r1 ou r2
(r)	r
^r	r en début de ligne
r$	r en fin de ligne
"s"	Le string s
\c	Le caractère c
r1 / r2	r1 quand il est suivi de r2



- [a-zA-z] Une lettre.
- [0-9] Un chiffre.
- a[^A-Za-z]b Un a, suivi d’un caractère non alphabétique, suivi d’un b.
- ^Monsieur Monsieur en début de ligne.
- [a-zA-Z]([a-zA-Z]|[0-9])* Un identifiant Pascal. . . .

Tiré de http://www.ulb.ac.be/di/ssd/ggeeraer/lg/extexpreg_print.pdf

Rule-based methods using regular expressions

- **Donnez l’expression régulière acceptant l’ensemble des phrases «correctes» selon les critères suivants :**
  - le premier mot de la phrase a une majuscule ;
  - la phrase se termine par un point ;
  - la phrase est composée d’un ou plusieurs mots (caractères a...z et A...Z), séparés par un espace ;

```
^[A-Z][A-Za-z]*(\ [A-Za-z]+)*\.$
```

Sites pour vérifier les expressions régulières: regexplanet.com

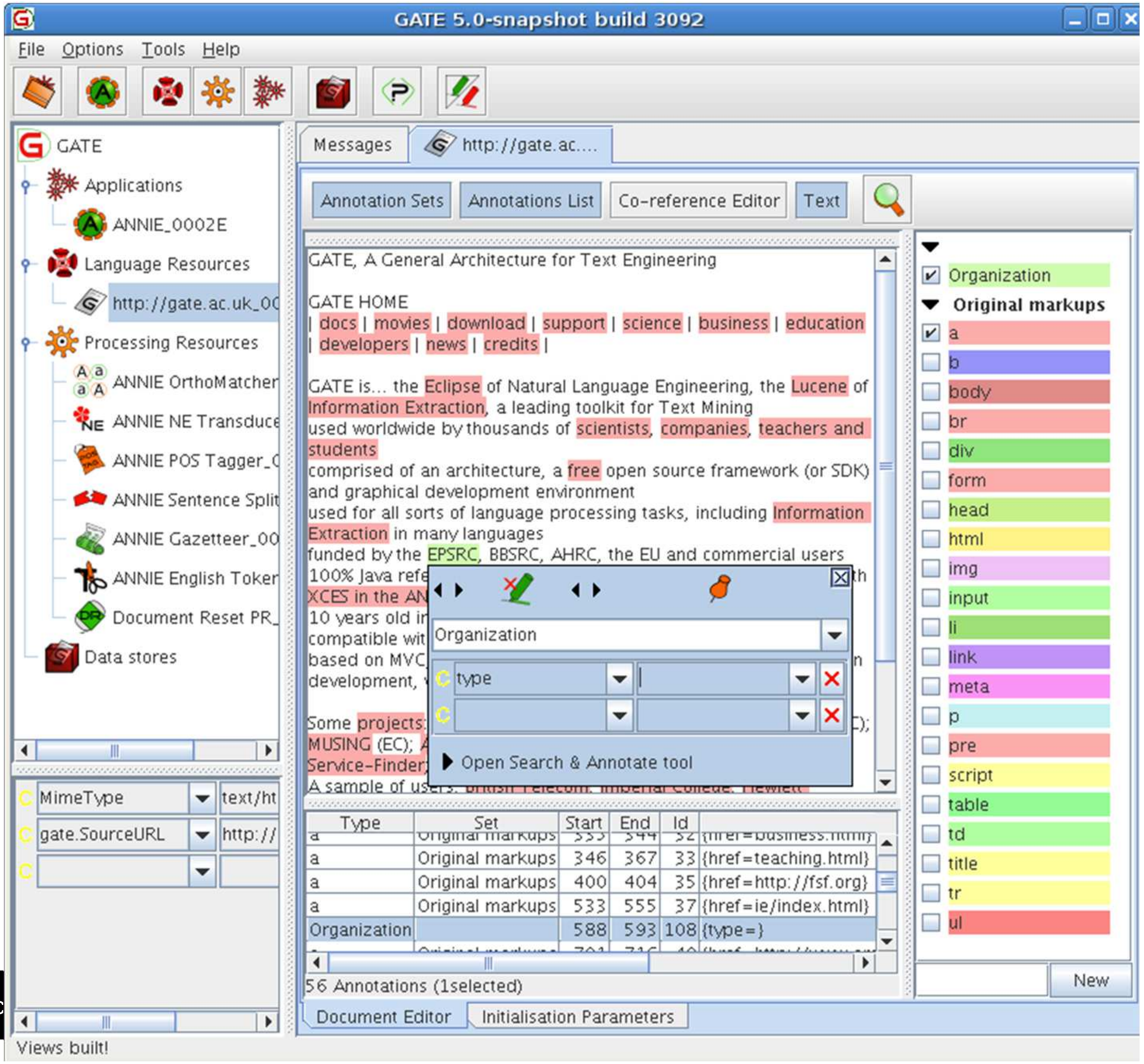
Tiré de http://www.ulb.ac.be/di/ssd/ggeeraer/lg/extexpreg_print.pdf

Tools

- Unitex : http://www-igm.univ-mlv.fr/~unitex/
- Les grammaires de NLTK
- Gate

GATE

- **General Architecture for Text Engineering,**
- **Suite Java pour l'extraction d'info et le NLP,**
- **Utilisé à l'échelle internationale avec des mises à jour continues,**
- **Intégration facile des différents outils et formats: divers taggers etc.**



GATE, A General Architecture for Text Engineering

GATE HOME
| docs | movies | download | support | science | business | education
| developers | news | credits |

GATE is... the Eclipse of Natural Language Engineering, the Lucene of Information Extraction, a leading toolkit for Text Mining
used worldwide by thousands of scientists, companies, teachers and students
comprised of an architecture, a free open source framework (or SDK) and graphical development environment
used for all sorts of language processing tasks, including Information Extraction in many languages
funded by the EPSRC, BBSRC, AHRC, the EU and commercial users
100% Java refe
XCES in the AN
10 years old in
compatible wit
based on MVC
development,

Some projects:
MUSING (EC);
Service-Finder
A sample of users: British Telecom, Imperial College, Hewlett

Type	Set	Start	End	Id	
a	Original markups	333	344	32	{href=business.html}
a	Original markups	346	367	33	{href=teaching.html}
a	Original markups	400	404	35	{href=http://fsf.org}
a	Original markups	533	555	37	{href=ie/index.html}
Organization		588	593	108	{type=}

56 Annotations (1selected)

GATE

Fonctionnalités:

Système d'extraction d'information (ANNIE)

Annotation à base de règles: JAPE

Ontologies

Machine Learning

Dictionnaires externes (Gazetteer)

Permet une conception d'un système hybride: à base de règles + Machine Learning

- Interface pour l'annotation manuelle
- Possibilité d'intégrer GATE à Hadoop :

Hadoop-GATE https://github.com/wpm/Hadoop-GATE

GATE

- **Différents exemples de projets de recherche avec GATE**
  - Environnement web permettant d'effectuer les tâches d'annotation manuelle (crowdsourcing) (Bontcheva et al., 2014)
  - Interface permettant d'interroger des ontologies (Damljanovic, 2010)
  - Classification de textes en sentiments:
    - GATE+SVM (Funk, 2008)
    - À base de règles JAPE

GATE : JAPE Grammars

- **Voir le tutoriel : https://gate.ac.uk/sale/thakker-jape-tutorial/GATE%20JAPE%20manual.pdf**
- **Exemple :**
  - Texte : *AC Milan player David Beckham is going to comment on his future with LA Galaxy, who are eager to keep him in USA.*
  - Règle : If mention of the word “player” followed by a name of a person Then the person = a player.

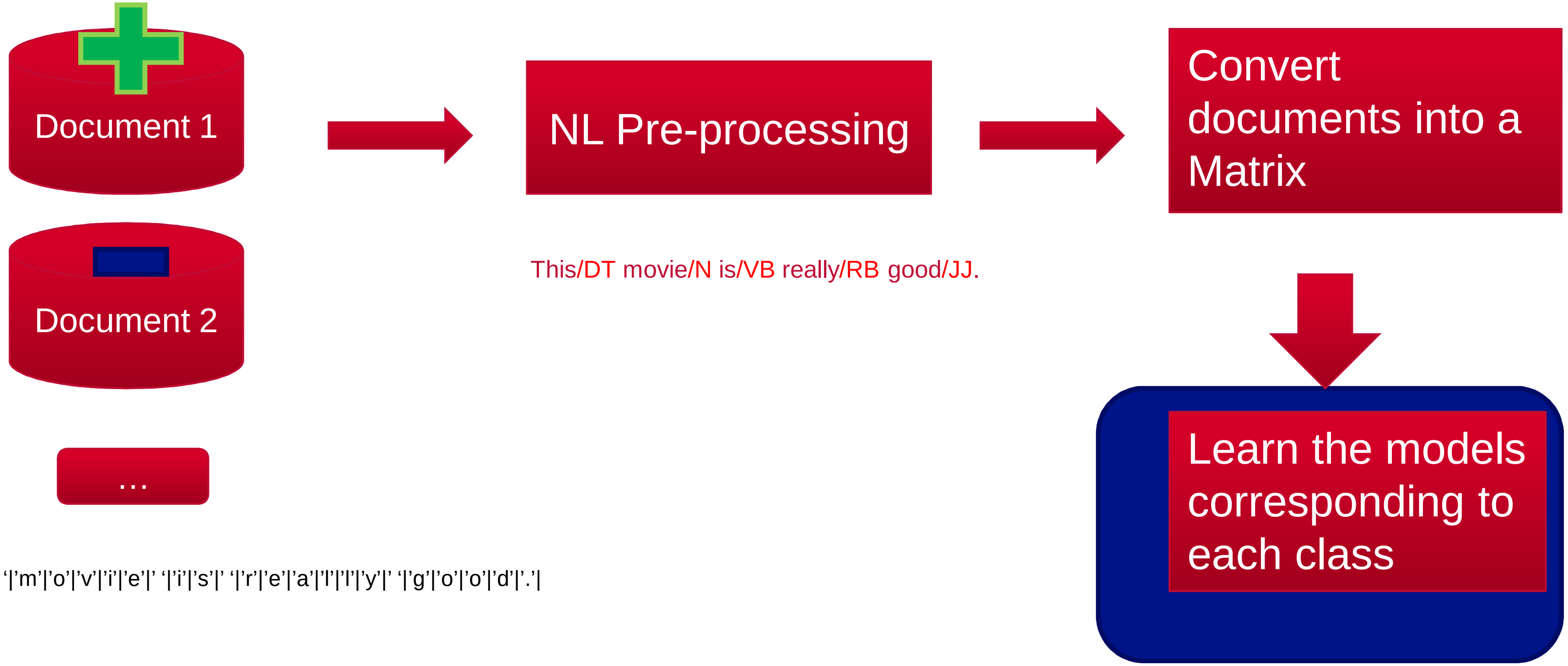
```
Phase:nestedpatternphase
Input: Lookup Token
//note that we are using Lookup and Token both inside our rules.
Options: control = brill
Rule: playerid
(
 {Token.string == "player"}
)
:temp
(
{Lookup.majorType == Person}
|
(
 {Token.kind==word, Token.category==NNP, Token.orth==upperInitial}
 {Token.kind==word, Token.category==NNP, Token.orth==upperInitial}
)
)
```

Supervised machine learning

- **Phase 1 – learning**
  - Training corpus = set of documents annotated Annotation : each document is assigned to a class :
  - Goal : Learn from this corpus the specific features of each class
- **Phase 2 – classification**
  - Using the learned features, the system is able to assign a class to a new document

Phase 1 – learning

- **Learning the classes**



Document 1

Document 2

…

NL Pre-processing

Convert documents into a Matrix

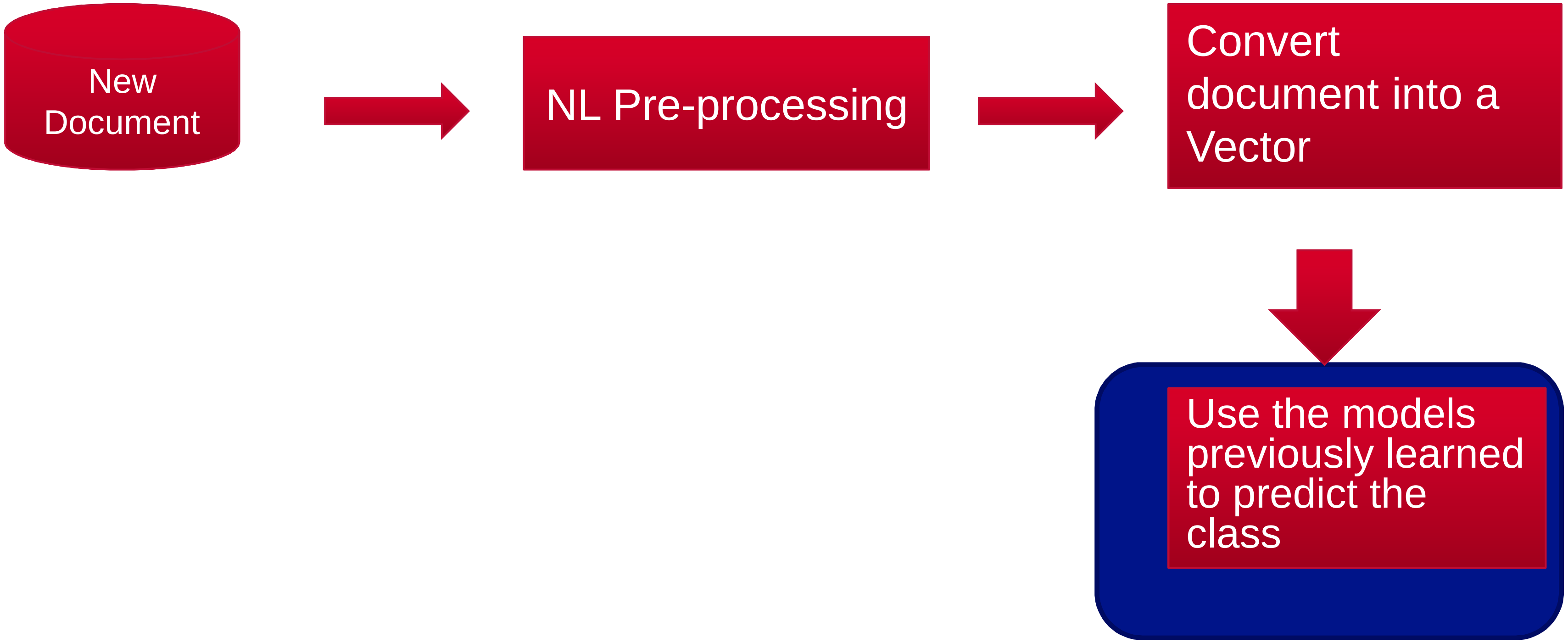
Learn the models corresponding to each class

This/DT movie/N is/VB really/RB good/JJ.

|’T’|’h’|’i’|’s’|’ ‘|’m’|’o’|’v’|’i’|’e’|’ ‘|’i’|’s’|’ ‘|’r’|’e’|’a’|’l’|’l’|’y’|’ ‘|’g’|’o’|’o’|’d’|’.’|

Phase 2 – classification

- **Predict the class of a new document**

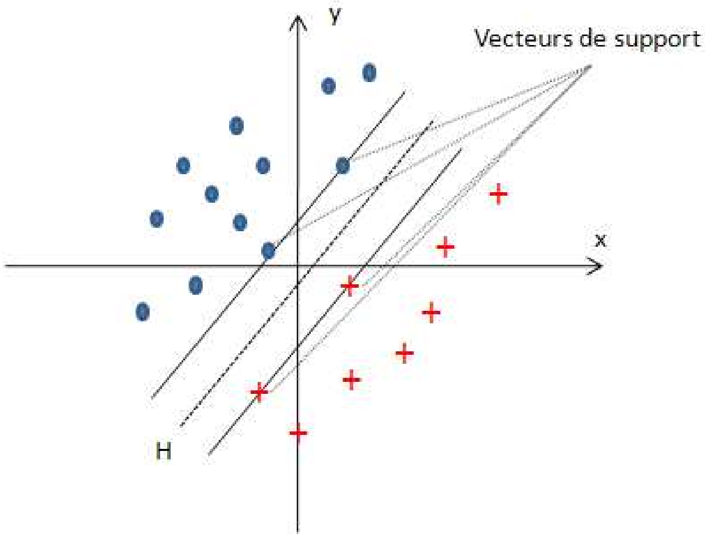

New Document
NL Pre-processing
Convert document into a Vector
Use the models previously learned to predict the class

Generative vs. Discriminative Models

- **Generative (joint) models P(c, d)**
  - Model the distribution of individual classes and place probabilities over both observed data and hidden variables (such as labels)
  - E.g. hidden Markov models, Naïve Bayes,
- **Discriminative (conditional) models P(c|d)**
  - Learn boundaries between classes. Take data as given and put probability over the hidden structure given the data.
  - E.g. logistic regression, maximum entropy models, conditional random fields, support-vector machines, ...

Reminder – Support Vector Machines

- **SVM – Support Vector Machines [Vapnik, 1995]**
  - Main idea



    - Split the training data into 2 sets while maximizing the distance to the separating hyperplan
    - Support vectors : the closest points to the hyperplan
    - Margin : minimal distance between the hyperplan and the training samples
    - => learning = maximize the margin
    - Decision : position of the new point relative to the hyperplan

Reminder – Support Vector Machines

SVM – Support Vector Machines [Vapnik, 1995]

- Usually
  - Use a transformation (a kernel) to move to a space with more dimensions to ensure that the problem can be linearly solved
  - Examples:
    - Linear : $k(x,y) = \exp(\frac{-\|x-y\|^2}{2\sigma^2})$
    - Gaussian $k(x,y) = x.y$
    - Polynomial $K(x,y) = (1 + x.y)^d$

Naive Bayes Classifier

Classification Principle

- Choose the class c maximizing $P(c \mid o)$
  - — Given an observation $o = document$

$$\boxed{\hat{c} = \arg\max_{c} P(c \mid o)}$$

  - — Bayes rule + the fact that $P(o)$ is independent from the class =>

$$\boxed{\hat{c} = \arg\max_{c} P(c \mid o) = \arg\max_{c} \frac{P(o \mid c)P(c)}{P(o)} = \arg\max_{c} P(o \mid c)P(c)}$$

Naive Bayes Classifier

$$\hat{c} = \arg\max_{c} P(c \mid o) = \arg\max_{c} \frac{P(o \mid c)P(c)}{P(o)} = \arg\max_{c} P(o \mid c)P(c)$$

- Naive : assumptions of strong independance between the features
  - $o = doc$ and $(m_1, \ldots.., m_N)$ the words of document o
  - $P(o|c) = P(m_1, \ldots.., m_N|c)$ $=\prod_{i=1}^{N} P(m_i|$c) -> use the log

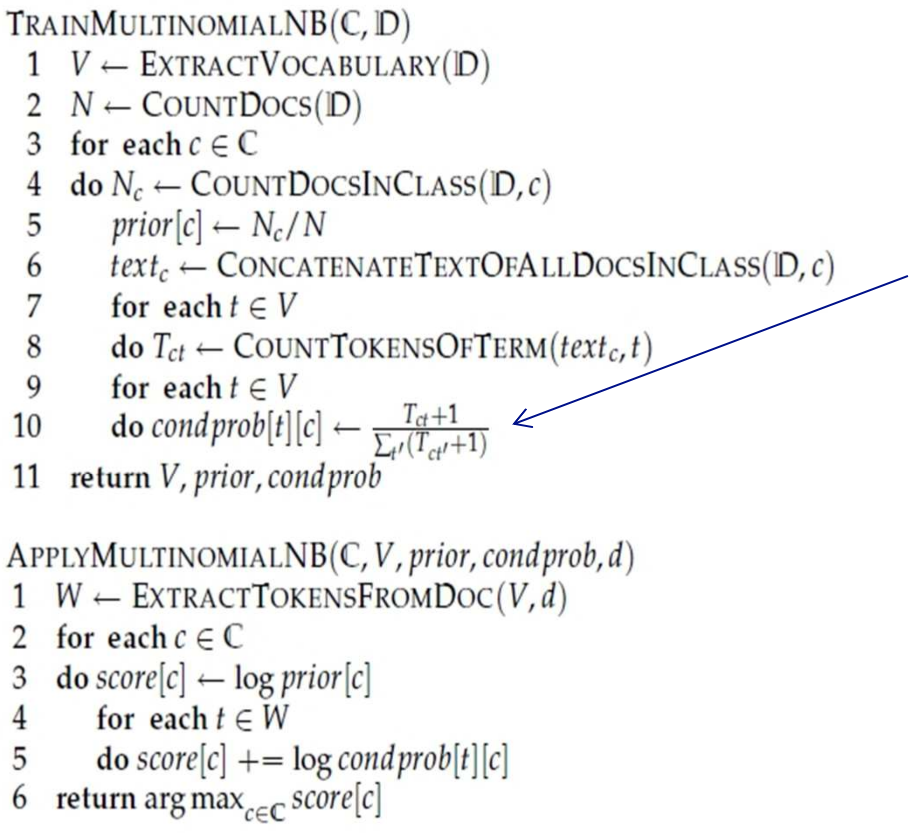
$$\hat{c} = \arg\max_{c \in \mathbb{R}} [log(P(c)) + \sum_{i=1}^{N} log(P(m_i/c))]$$

Naive Bayes Classifier

$$\boxed{\hat{c} = \arg\max_{c \in \mathbb{R}} [log(P(c)) + \sum_{i=1}^{N} log(P(m_i/c))]}$$

- Training on the labelled database
  - Estimating $P(c)$ and $P(m_i|c)$
    - $P(c) \; = \dfrac{documents\ in\ class\ C}{total\ number\ of\ documents}$
    - $P(m_i|c) = frequency\ of\ the\ word\ m_i\ in\ class\ c$

TP Sentiment analysis avec NB



```
TrainMultinomialNB($\mathbb{C}, \mathbb{D}$)
$V \leftarrow$ ExtractVocabulary($\mathbb{D}$)
$N \leftarrow$ CountDocs($\mathbb{D}$)
for each $c \in \mathbb{C}$
do $N_c \leftarrow$ CountDocsInClass($\mathbb{D}, c$)
    $prior[c] \leftarrow N_c / N$
    $text_c \leftarrow$ ConcatenateTextOfAllDocsInClass($\mathbb{D}, c$)
    for each $t \in V$
    do $T_{ct} \leftarrow$ CountTokensOfTerm($text_c, t$)
    for each $t \in V$
    do $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
return $V, prior, condprob$

ApplyMultinomialNB($\mathbb{C}, V, prior, condprob, d$)
$W \leftarrow$ ExtractTokensFromDoc($V, d$)
for each $c \in \mathbb{C}$
do $score[c] \leftarrow \log prior[c]$
    for each $t \in W$
    do $score[c] \mathrel{+}= \log condprob[t][c]$
return $\arg\max_{c \in \mathbb{C}} score[c]$
```

▶ **Figure 13.2** Naive Bayes algorithm (multinomial model): Training and testing.

$P(m_i|c)$ = *frequency of the word t in class c*
+ Laplace smoothing

Question

- **Is Naïve Bayes a generative or a discriminative model?**
  - Na¨ıve Bayes is a generative model!
  - P(c|d) = P(d|c)P(c) P(d) P(c|d)P(d) = P(d|c)P(c) = P(d, c)
  - While we use a conditional probability P(c|d) for classification, we model the joint probability of c and d
  - This means it is trivial to invert the process and generate new text given a class label.

Logistic regression

- **If we only want to classify text, we do not need the full power of a generative model, but a discriminative model is sufficient.**
- **We only want to learn P(c|d).**
- **A general framework for this is logistic regression.**
  - logistic because it uses a logistic function regression combines a feature vector (d) with weights (β) to compute an answer

Logistic regression

- **Binary case:**

$$P(\text{true}|d) = \frac{1}{1 + exp(\beta_0 + \sum_i \beta_i X_i)}$$

$$P(\text{false}|d) = \frac{exp(\beta_0 + \sum_i \beta_i X_i)}{1 + exp(\beta_0 + \sum_i \beta_i X_i)}$$

- **Multinomial case:** $P(c|d) = \frac{exp(\beta_{c,0} + \sum_i \beta_{c,i} X_i)}{\sum_{c'} exp(\beta_{c',0} + \sum_i \beta_{c',i} X_i)}$

$$P(c|d) = \frac{exp(z_c)}{\sum_{c'} exp(z_{c'})}$$

Softmax function

  - where X are the features contained in d (for example tf-idf of word2vec).

Logistic regression

- Given this model formulation,
  - we want to learn parameters (the weights β) that maximise the conditional likelihood of the data according to the model P(c/d).
- Due to the softmax function
  - we not only construct a classifier, **but learn probability distributions over classes.**
- There are many ways to chose weights :
  - Perceptron : Find misclassified examples and move weights in the direction of their correct class
  - Margin-Based Methods such as Support Vector Machines : can be used for learning weights
  - **Logistic Regression : Directly maximize the conditional log-likelihood via gradient descent**

Logistic regression

- **Directly maximize the conditional log-likelihood**

$$\log P(c|d,\beta) = \log \prod_{c,d\in(C,D)} P(c_n|d_n,\beta)$$

$$= \sum_{c,d\in(C,D)} \log P(c_n|d_n,\beta)$$

$$\log P(c_n|d_n,\beta) = \sum_{c,d\in(C,D)} \log \frac{exp(\sum_i \beta_{c,i} X_i)}{\sum_{c'} exp(\sum_i \beta_{c',i} X_i)}$$

- **via gradient descent**
  - Derivative with respect to β is concave

Evaluation scores

- **In the task of correct assignment to class c**
  - R = Recall : (number of system’s correct assignments to class c) / (number of documents labelled c)
    - A system that tends to infrequently assign class c (high system *silence* for class c) will have a low recall

Evaluation scores

- **In the task of correct assignment to class c**
  - P = Precision : (number of system’s correct assignments to class c) / (number of system’s assignments to class c)
    - A system that tends to allocate class c too frequently (system *noise* is high for class c) will have precision

Evaluation scores

- **In the task of correct assignment to class c**
  - F-score : harmonic mean between recall and precision = 2×(P ×R) / (P +R)

Hybrid methods

- **Hand-crafted features based on linguistic features to support classification**
  - Example 1 : In the term-document matrix
    - The terms are replaced by concepts in the term / document matrix « j'aimerais » => attentes du client L. Kuznick, A-L. Guènet, A. Peradotto, and C. Clavel. L'apport des concepts métiers pour la classification des questions ouvertes d'enquête. In Actes de TALN, Montréal, 2010.
  - Example 2 : linguistic and syntactic patterns are used as inputs of supervised machine learning
    - Barrière, V., Clavel, C., Essid, E., Opinion Dynamics Modeling for Movie Review Transcripts Classification with Hidden Conditional Random Fields, Interspeech 2017

Semi-supervised learning

- **When using big unlabelled data but labelled data are missing**
  - Example
    - – Train word2vec on the unlabelled data
    - – Supervised learning on the labelled part
      - Johnson, R., & Zhang, T. (2015). Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*

References

- **https://nlp.stanford.edu/IR-book**
- **Foundations of Statistical Natural Language Processing Christopher D. Manning and Hinrich Schütze**
- Deep Natural Language Processing course offered in Hilary Term 2017 at the University of Oxford.
- In French :
  - *Une petite introduction au traitement automatique des langues naturelles* par François Yvon http://perso.limsi.fr/Individu/anne/coursM2R/intro.pdf
  - *Introduction au TALN et à l'ingénierie linguistique* par Isabelle Tellier http://www.lattice.cnrs.fr/sites/itellier/poly_info_ling/info-ling.pdf

Deep learning for natural language processing

Chloé Clavel,
chloe.clavel@telecom-paristech.fr,

Telecom ParisTech, France

Outline of the course

Introduction

- Objectives of the course
- Problem statement

Classical machine learning vs.deep learning

Multilayer Neural Networks

- Use for NLP
- ML NN inputs/outputs
- ML NN Layers and back propagation

Other NN architectures

- Convolutional Neural Networks
- Recursive deep models
- Recurrent neural networks

Objectives of the course

At the end of this lecture,

- you will be able to explain the "philosophy" of deep learning vs. classical machine learning approaches
- you will master the ML NN architectures for NLP tasks
- you will be able to cite other neural network architectures for NLP tasks and explain their underlying principles

Problem statement

- Training dataset consisting of samples $\{xi, yi\}i = 1, N$
- xi - inputs, e.g. words (indices or vectors !), context windows, sentences, documents, etc.
- yi - labels we try to predict, e.g. other words, class : sentiment, named entities, buy/sell decision,

NLP tasks

Assigning labels to words :

- Part-Of-Speech tagging (POS),
- chunking (CHUNK),
- Named Entity Recognition (NER)
- Semantic Role Labeling (SRL)

Assigning labels to sentence/document :

- Topic classification
- opinon classification (positive vs. negative)

Classical machine learning vs. deep learning

Could speech and language processing be seen as a linear problem ?

NLP requirements

Input-output functions should solve the selectivity-invariance dilemma

- insensitive to irrelevant variations of the inputs
- very sensitive to particular minute variations of the inputs
- (for example : the pitch variation due to the speaker when you want to develop an emotion recognition system)

First option :Classical machine learning

In the simplest cases :

- linear classifiers on top of **hand-engineered features**
- A two-class linear classifier computes a weighted sum of the feature vector component
- if the weighted sum is above a threshold → choose the class

First option : Classical machine learning

With this option, the challenge is on the design of hand-engineered features

Using semantics, lexicons, etc. (see Lectures 1 and 2) in order to build feature extractor that solves the selectivity-invariance dilemma : build representations that are

- selective to the aspects of the text that are important for discrimination
- invariant to irrelevant aspects

Requires engineering skill and linguistic expertise

Second option : Deep learning

Statement

- do not use linguistic expertise and build general purpose learning procedures to automatically learn representations

Philosophy

- input : try to pre-process the features as little as possible and
- use a multilayer neural network (NN) architecture trained in an *end-to-end* fashion.
- ex : use characters as input

Second option : Deep learning

Deep learning architecture

Multilayer stack of simple modules

- subject to learning
- that computes non-linear input-output mappings
- that transform their inputs to increase both the selectivity and the invariance of the representation

Second option : Deep learning

Deep learning architecture

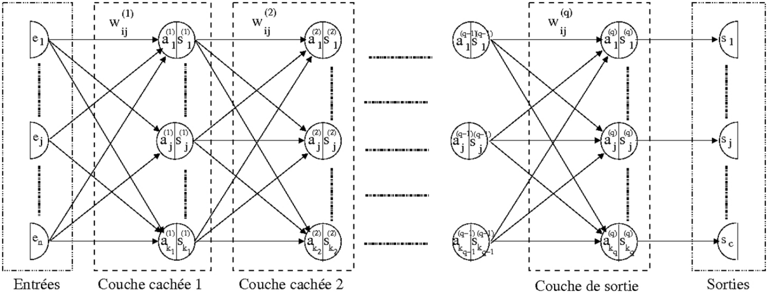
For example, with a depth of 5 to 20 non-linear layers, a system can implement extremely intricate functions of its inputs that are simultaneously sensitive to minutes details and insensitive to large irrelevant variations

Multilayer Neural Networks - ML NN

1. Use for NLP
2. Inputs
3. Outputs
4. Layers and backpropagation

ML Neural networks principles

- A multilayer neural network can distort the input space to make the classes of data linearly separable
- If the weights are set correctly, a neural network with enough neurons and a non-linear activation function can approximate a very wide range of mathematical functions



ML NN use for NLP

- For binary classification problems
- For multiclass classification problems
- More complex structured prediction problems

Advantages : The non-linearity of the network, as well as the ability to easily integrate pre-trained word embeddings, often lead to superior classification accuracy.

ML NN use for NLP

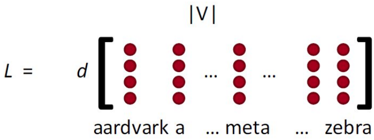
Examples :

- Syntactic parsing : Chen, D., & Manning, C. (2014). A Fast and Accurate Dependency Parser using Neural Networks. EMNLP 2014
- Dialog state tracking : Henderson, M., Thomson, B., & Young, S. (2013). Deep Neural Network Approach for the Dialog State Tracking Challenge. Sigdial 2013

ML NN Inputs

Reminder from Lecture 2b about word embeddings

INPUT : words are represented as indices taken from a finite dictionary $\mathcal{D}$
OUTPUT : Lookup table feature vector



Conceptually you get a word's vector by left multiplying a one-hot vector e by L

ML NN Inputs

Option 1

- Use pre-trained word vectors (the best to do : if you have a small training dataset). Example :

  → In Valentin Barriere, Chloé Clavel, Slim Essid : « Attitude Classification in Adjacency Pairs of a Human-Agent Interaction with Hidden Conditional Random Fields », ICASSP 2018

  → we had about 500 utterances and we use representations learnt from a Google News corpus of 100 billions words `https://code.google.com/archive/p/word2vec/`

ML NN Inputs

Option 2

- Train your word vectors on your database in an unsupervised manner (the best to do : if you have a big dataset with peculiarities). Example :

  → In Maslowski, I., Lagarde, D., Clavel, C., In-the-wild chatbot corpus from opinion analysis to interaction problem detection, ICNLSSP 2017

  → we train word2vec on 1,813,934 dialogues.

ML NN Inputs

Option 3

Re-train vectors for your task (the best to do : if you have a big **labelled** dataset)
How to train multilayer neural network (NN) architecture, in an end-to-end fashion ?
STEP 1 : The architecture takes the input sentences and learns several layers of feature extraction that process the inputs.
STEP 2 : The features computed by the deep layers of the network are automatically trained by backpropagation to be relevant to the task.

Window approach

Starting from an example : input : "Museums in Paris are amazing"
output : "O O B_LOC O O"
The output for "Paris" depends on its context of occurrence ("Paris Hilton" will be a person)
→ build a context window : e.g. we represent each word using a 4-dimensional word vector and we use a 5-word window (the previous 2 and the following 2) as input (as in the above example), then the input $x \in \mathbb{R}^{20}$.

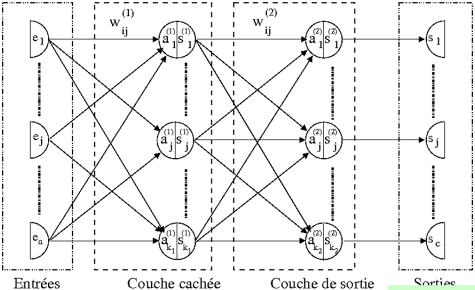
ML NN outputs

Case where the dimension of the outputs $d_{out} = 1$ which means that the network's output is a scalar.
Such networks can be used

- for regression (or scoring) by considering the value of the output
- for binary classification by consulting the sign of the output.

ML NN outputs

Networks with $d_{out} = c > 1$ can be used for k-class classification, by associating each dimension with a class, and looking for the dimension with maximal value.



n

$k_1$unités de la couche 1

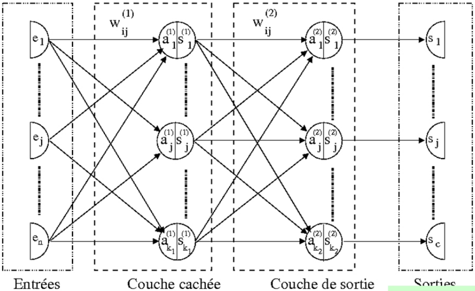
$k_2 = c$

l=1:c, nombre de classes

ML NN outputs

Similarly, if the output vector entries are positive and sum to one, the output can be interpreted as a distribution over class assignments (such output normalization is typically achieved by applying a softmax transformation on the output layer).

ML NN Layers



n

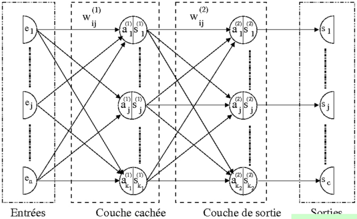
$k_1$unités de la couche 1

k_2 = c

l=1:c, nombre de classes

Hidden layer : between the inputs and the outputs there are layers with hidden outputs

ML NN Layers



n

$k_1$unités de la couche 1

k_2 = c

l=1:c, nombre de classes

FORWARD : we need to compute all the outputs of the $m-1$ layer to compute the outputs of the m layer.
PRACTICE in the case of two layers : try to compute the final outputs

Training and backpropagation algorithm

1. define the loss
2. compute partial derivatives
3. apply gradient descent algorithm from output layers to input layers

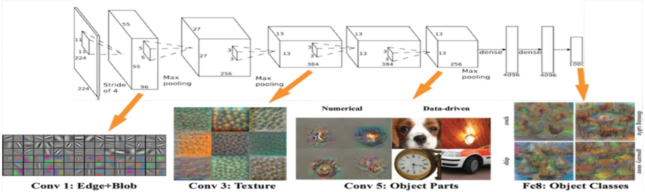
See lecture Neural Networks

Other NN architectures

- Convolutional neural networks
- Recursive deep models
- Recurrent neural networks and variants

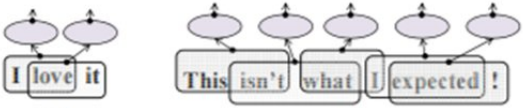
Convolutional Neural Networks

- Variation of multilayer perceptrons designed to require minimal preprocessing and using *convolutional* layers
- the network learns the filters



Convolutional Neural Networks

Example of use for the text : Johnson, R., & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks.

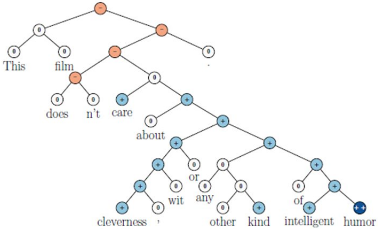


(a) (b)

Figure 3: Convolution layer for variable-sized text.

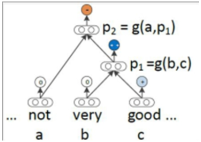
Recursive deep models

Tree representation of movie sentences using Stranford parser
Each node of the tree is labelled in (-, +,0) to provide the structure that is required for the training of a recursive model (Sentiment TreeBank Database)



Recursive deep models

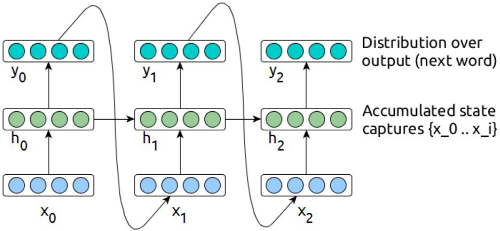
Training step : learning g function that compute the upper outputs in the binary tree



REF : R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, EMNLP 2013.

Recurrent Neural Networks

Use for language models


y_0
y_1
y_2
Distribution over output (next word)
h_0
h_1
h_2
Accumulated state captures {x_0 .. x_i}
x_0
x_1
x_2

- Reads inputs xi to accumulate state hi and predict outputs yi
- Variants : LSTM networks (Long Short Term Memory Networks), RNN using gating mechanisms such as GRU (Gated Recurrent Units)

Support and materials

- LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015) : 436.
- Lectures from Stanford `http://cs224d.stanford.edu/lectures/CS224d-Lecture4.pdf`
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug), 2493-2537.
- Goldberg, Yoav. "A primer on neural network models for natural language processing." Journal of Artificial Intelligence Research 57 (2016) : 345-420.
- Lectures from Oxford : `https://github.com/oxford-cs-deepnlp-2017/lectures`