

Traitement Automatique des Langues Naturelles

Cours 3 : Des mots à la structure de la phrase

Chloé Braud, Philippe Muller

Master IAFA 2024-2025

Fin du cours sur les mots : Outils de pré-traitements

NLTK :

- tokenizer par défaut basé sur des regex (Penn TreeBank)
<https://www.nltk.org/api/nltk.tokenize.treebank.html#nltk.tokenize.treebank.TreebankWordTokenizer>
- TweetTokenizer : aussi basé sur des règles https://www.nltk.org/_modules/nltk/tokenize/casual.html#TweetTokenizer
- Multi-word expression Tokenizer : après tokenisation, regroupe les expressions multi-mots, toujours basé sur des règles / lexiques
https://www.nltk.org/_modules/nltk/tokenize/mwe.html
- Difficile à adapter à d'autres langues ...

```
>>> import nltk
>>> sentence = """At eight o'clock on Thursday morning
... Arthur didn't feel very good."""
>>> tokens = nltk.word_tokenize(sentence)
>>> tokens
['At', 'eight', "o'clock", 'on', 'Thursday', 'morning',
'Arthur', 'did', "n't", 'feel', 'very', 'good', '.']
```

Fin du cours sur les mots : Outils de pré-traitements

Spacy :

- *spaCy is a free, open-source library for advanced Natural Language Processing (NLP) in Python*
- *spaCy is not research software. It's built on the latest research, but it's designed to get things done. This leads to fairly different design decisions than NLTK or CoreNLP, which were created as platforms for teaching and research. The main difference is that spaCy is integrated and opinionated. spaCy tries to avoid asking the user to choose between multiple algorithms that deliver equivalent functionality.*
- également basé sur des règles pour la tokenisation
<https://spacy.io/usage/spacy-101> mais proprement implémenté pour de multiples langues
- ne gère pas les expressions multi-mots

Fin du cours sur les mots : Outils de pré-traitements

Stanza (by stanford NLP group)

- Stanza is a collection of accurate and efficient tools for the linguistic analysis of many human languages. (...) The toolkit is designed to be parallel among more than 70 languages, using the Universal Dependencies formalism.
- Statistical tokenization based on neural models

[https://github.com/stanfordnlp/stanza/blob/main/
stanza/models/tokenizer.py](https://github.com/stanfordnlp/stanza/blob/main/stanza/models/tokenizer.py)

- (inclue une interface avec CoreNLP, en java)
- (peut être utilisé via SpaCy

<https://spacy.io/universe/project/spacy-stanza>)

- expansion multi-mots mais pas fusion

[https://stanfordnlp.github.io/stanza/mwt.html#
accessing-syntactic-words-for-multi-word-token](https://stanfordnlp.github.io/stanza/mwt.html#accessing-syntactic-words-for-multi-word-token)

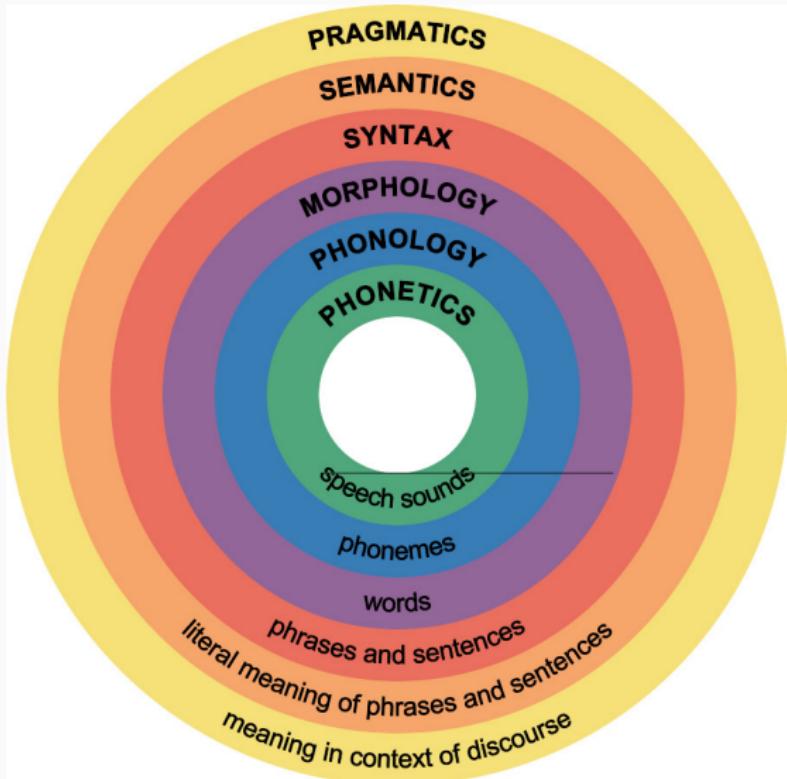
- Performances :

<https://stanfordnlp.github.io/stanza/performance.html>

Un peu de contexte linguistique

Les niveaux d'analyse

- phonologie: les sons
- morphologie: les mots et leur forme
- **syntaxe: l'organisation des mots en phrase**
- sémantique: le sens dans la phrase
- pragmatique: le sens en contexte



Bilan : Les mots

on a vu

- définition de l'unité lexicale
- dépendances lexicales (liens entre les formes, dérivations...)
- première approche / ordre des mots dans la phrase (n-grammes)
- catégorisation (catégorie de mots: verbe, nom etc)
- extraction d'information par patrons lexico-syntaxiques (plutôt lexicaux pour l'instant)

Raffiner les extractions

on avait vu les constructions particulières **nom + adjetif** : focalisation de recherche d'information

12 chose matériel

14 cause efficient

14 chose sensible

14 genre humain

15 corps humain

17 chose semblable

21 réalité objectif

25 lumière naturel

28 chose corporel

38 esprit humain

→ classes de mots particulières

Classes de mots

"Parties du discours" / Catégories morpho-syntaxiques (part of speech, POS)

- noms voiture, maison, banane
 - noms propres Jean, New-York, Linux
 - pronoms il, lui, elle, je, celui-ci
 - verbes sauter, flatter, avoir, faire
 - adjectifs petit, grand
 - adverbes bien, vite, mal, trop, dedans, finalement
 - prépositions avant, de à
 - déterminants le, ce, quelques, une
 - conjonctions et, parce que, ou, donc
 - nombres zéro, un deux, trois
 - interjections euh, hein

(on peut faire des sous-classes)

On dit que les catégories appartiennent à une classe :

- fermée: prépositions, pronoms, déterminants, conjonctions
- ou ouvertes: noms, adjectifs, adverbes, verbes (beaucoup de créations dans ces classes)
- mais ? nombres, interjections

Des jeux de catégories différents selon les corpus / projets d'annotation :
e.g. dans le Penn Treebank (anglais)

Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential <i>there</i>
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP\$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	<i>to</i>
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP\$	Possessive wh-pronoun
36.	WRB	Wh-adverb

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn-treebank_pos.html

Des jeux de catégories différents selon les corpus / projets d'annotation :
 e.g. dans le French Treebank

Catégorie	Sous-catégorie	Morphologie	Description
N	C, P	f,m + s,p	Noms
A	card, ord, poss, qual, indefi, inter, exclam	1,2,3+ f,m + s,p	Adjectifs
Adv	-, inter, exclam, neg	-	Adverbes
P	-	-	Prépositions
D	card, dem, def, indef, exclam, neg, poss, inter, part	1,2,3+ f,m + s,p	Déterminants
CL	S, R, O	1,2,3+ f,m + s,p	Pronoms clitiques
PRO	card, inter, pers, neg, poss, rel, indef	1,2,3+ f,m + s,p	Autres pronoms
PREF			préfixe
C	S, C	-	Conjonctions
I	-	-	Interjections
V	-	W, G, K, P, I, J, F, T, C, S, Y + 1,2,3 + f,m + s,p	Verbes
ET	-	-	Mots étrangers
PONCT	S, W	-	Ponctuation

Tableau 1. Liste des catégories et sous-catégories du FTB.

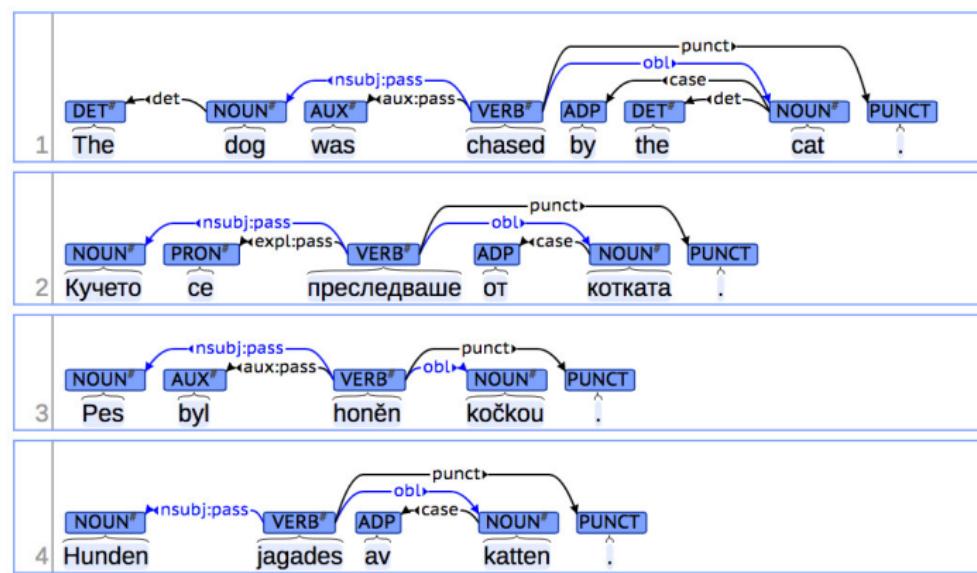
<http://ftb.linguist.univ-paris-diderot.fr/fichiers/public/guide-morphosynt.pdf>

Il existe de nombreux corpus annotés en POS

- Penn Treebank : l'un des plus anciens projets [Marcu et al. 1999] → a donné lieu à des projets dans d'autres langues, i.e. FTB, Penn Chinese Treebank [https://verbs.colorado.edu/chinese/ ...](https://verbs.colorado.edu/chinese/)
- Universal Dependencies <https://universaldependencies.org/>
 - *open community effort with over 300 contributors producing nearly 200 treebanks in over 100 languages*
 - *cross-linguistically consistent treebank annotation for many languages*
 - *the general philosophy is to provide a universal inventory of categories*

Annotations UD

Exemples parallèles en anglais, Bulgare, Tchèque et Suédois :



Annotations UD

Un ensemble de POS tag plus restreint :

Open class words	Closed class words	Other
<u>ADJ</u>	<u>ADP</u>	<u>PUNCT</u>
<u>ADV</u>	<u>AUX</u>	<u>SYM</u>
<u>INTJ</u>	<u>CCONJ</u>	<u>X</u>
<u>NOUN</u>	<u>DET</u>	
<u>PROPN</u>	<u>NUM</u>	
<u>VERB</u>	<u>PART</u>	
	<u>PRON</u>	
	<u>SCONJ</u>	

Annotations UD

Mais un gros jeu de traits en plus pour distinguer des propriétés additionnelles des mots:

Lexical features*	Inflectional features*	
	Nominal*	Verbal*
PronType	Gender	VerbForm
NumType	Animacy	Mood
Poss	NounClass	Tense
Reflex	Number	Aspect
Foreign	Case	Voice
Abbr	Definite	Evident
Typo	Degree	Polarity
	Person	
	Polite	
	Clusivity	

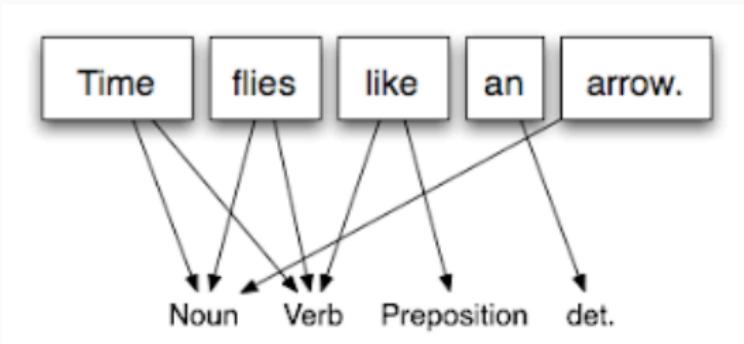
Index: A [abbreviation](#), [abessive](#), [ablative](#), [absolute superlative](#), [absolutive](#), [accusative](#), [active](#), [actor-focus voice](#), [additive](#), [adelative](#), [adessive](#), [adlative](#), [admirative](#), [adverbial participle](#), [affirmative](#), [allative](#), [animate](#), [antipassive](#), [aorist](#), [article](#), [aspect](#), [associative](#), B [bantu noun class](#), [benefactive](#), [beneficiary-focus voice](#), C [cardinal](#), [caritative](#), [case](#), [causative case](#), [causative voice](#), [clusivity](#), [collective noun](#), [collective numeral](#), [collective pronominal](#), [comitative](#), [common gender](#), [comparative case](#), [comparative degree](#), [complex definiteness](#), [conditional](#), [conjunctive](#), [considerative](#), [construct state](#), [converb](#), [count plural](#), [counting form](#), D [dative](#), [definiteness](#), [degree of comparison](#), [deative](#), [demonstrative](#), [desiderative](#), [destinative](#), [direct case](#), [direct voice](#), [directional allative](#), [distributive case](#), [distributive numeral](#), [dual](#), E [elative](#), [elevated referent](#), [emphatic](#), [equative case](#), [equative degree](#), [ergative](#), [essive](#), [evidentiality](#), [exclamative](#), [exclusive](#), F [factive](#), [feminine](#), [finite verb](#), [first person](#), [firsthands](#), [foreign word](#), [formal](#), [fourth person](#), [fraction](#), [frequentative](#), [future](#), G [gender](#), [genitive](#), [gerund](#), [gerundive](#), [greater paucal](#), [greater plural](#), H [habitual](#), [human](#), [humbled speaker](#), I [illative](#), [imperative](#), [imperfect tense](#), [imperfective aspect](#), [inanimate](#), [inclusive](#), [indefinite](#), [indefinite pronominal](#), [indicative](#), [inelative](#), [inessive](#), [in infinitive](#), [informal](#), [injunctive](#), [inlative](#), [instructive](#), [instrumental](#), [interrogative mood](#), [interrogative pronominal](#), [inverse number](#), [inverse voice](#), [irrealis](#), [iterative](#), J [jussive](#), L [lative](#), [location-focus voice](#), [locative](#), M [masculine](#), [masdar](#), [mass noun](#), [middle voice](#), [modality](#), [mood](#), [motivative](#), [multiplicative numeral](#), N [narrative](#), [necessitative](#), [negative polarity](#), [negative pronominal](#), [neuter](#), [nominative](#), [non-finite verb](#), [non-firsthands](#), [non-human](#), [non-past](#), [non-specific indefinite](#), [noun class](#), [number](#), [numeral type](#), O [oblique case](#), [optative](#), [ordinal](#), P [participle](#), [partitive](#), [passive](#), [past](#), [past perfect](#), [patient-focus voice](#), [paucal](#), [perfective aspect](#), [perative](#), [person](#), [personal](#), [pluperfect](#), [plural](#), [plurale tantum](#), [polarity](#), [politeness](#), [positive degree](#), [positive polarity](#), [possessive](#), [potential](#), [present](#), [preterite](#), [privative](#), [progressive](#), [prolative](#), [pronominal type](#), [prospective](#), [purposive case](#), [purposive mood](#), Q [quantifier](#), [quantitative plural](#), [quotative](#), R [range numeral](#), [realis](#), [reciprocal pronominal](#), [reciprocal voice](#), [reduced definiteness](#), [reflexive](#), [register](#), [relative](#), S [second person](#), [set numeral](#), [singular](#), [singulare tantum](#), [specific indefinite](#), [subelative](#), [subessive](#), [subjunctive](#), [sublative](#), [superrelative](#), [superessive](#), [superlative case](#), [superlative degree](#), [supine](#), T [temporal](#), [tense](#), [terminal allative](#), [terminative](#), [third person](#), [total](#), [transgressive](#), [translative](#), [trial](#), [typo](#), U [uter](#), V [verb form](#), [verbal adjective](#), [verbal adverb](#), [verbal noun](#), [vocative](#), [voice](#), Z [zero person](#)

Etiquetage morpho-syntaxique / POS tagging



- Une étape de pré-traitement très commune
 - sur laquelle se fonde l'analyse syntaxique
 - mais aussi utile en soi (e.g. patrons lexico-syntaxiques..)
- Difficulté ? les ambiguïtés !

Ambiguités au niveau morpho-syntaxique



La	petite	brise	la	glace
N/P/Det	A/N	V/N	N/P/Det	V/N
La	petite	fille	brise	la
N/P/Det	A/N	N	V/N	N/P/Det

Pire en anglais (11% des types mais ≈ 40% des occurrences)

- La POS détermine le lemme 'brise' = brise / briser
- La séquence de POS détermine l'analyse syntaxique (voir ci-après)

Ambiguités morpho-syntaxiques

Exemple de “que” dans les corpus de référence du français
(French TreeBank + Sequoia)

Tag	Compte	Exemple
Adverbe	188	publié qu'en anglais
Conj. de subordination	935	ils pensent que ...
Pronom relatif	438	des devises qu'elle utilisera
Pronom interrogatif	20	qu'attendez vous ?

Étiquetage morpho-syntaxique / POS tagging : modèles

De la séquence de tokens à la séquence de catégories morpho-syntaxiques

:

Entrée	La	petite	fille	brise	la	glace
Sortie	Det	A	N	V	Det	N

Modèles à base de traits

- L'apprentissage automatique est fondée sur la conception de **traits** (descripteurs, caractéristiques)
- Des indices élémentaires, qui font le lien entre certains aspects des données observées (d) et la classe (c) qu'on veut prédire
- La prédiction de la classe c dépend uniquement des traits f , construits à partir des données d
- Chaque trait a une valeur réelle/catégorielle/booléenne
- On applique des algorithmes d'apprentissage à partir de ces traits

Exemples

- Catégorisation de texte
 - classe c : ‘informatique’
 - Données d : document
 - Traits f : les mots (booléen/comptage par mot)
- Authentification d'auteurs
 - classe c : ‘Flaubert’
 - Données d : document (livre)
 - Traits f : les n-grammes extraits
- classification de sentiment
 - classe c : ‘sentiment négatif’
 - Données d : tweet
 - Traits f : les mots

Classificateur linéaire

- Classification depuis un ensemble de traits $\{f_i\}$ vers des classes $\{c\}$
- Attribuer un poids λ_i pour chaque trait f_i (et chaque classe c)
- Pour une certaine paire (c, d) les traits votent avec leur poids
- On prend la classe c qui maximise la somme des différents traits
- Différentes manières pour choisir les poids

Classificateur linéaire: exemple

- Classification de texte
- Comme traits, on prend la fréquence des lemmes
- $\max_{c \in C} \sum_i \lambda_{c,i} f_i$

"Le virus a surtout ciblé les ordinateurs sous Windows."

trait f_i	classe c_j	
	informatique	biologie
virus	0.5	0.6
cibler	0.4	0.2
ordinateur	0.8	-0.6
Windows	0.9	-1.2

- $vote(informatique) = 0.5 \times 1 + 0.4 \times 1 + 0.8 \times 1 + 0.9 \times 1 = 2.6$
- $vote(biologie) = 0.6 \times 1 + 0.2 \times 1 + (-0.6) \times 1 + (-1.2) \times 1 = -1.0$

→ En pratique e.g. classifieur MaxEnt

Problèmes de séquence

- Beaucoup de tâches en TAL ont des séquences en tant qu'entrée
- La tâche est alors d'étiqueter chaque élément de la séquence

Donald Tusk est le président du

Conseil européen depuis décembre 2014.

Problèmes de séquence

- Beaucoup de tâches en TAL ont des séquences en tant qu'entrée
- La tâche est alors d'étiqueter chaque élément de la séquence

POS tagging

Donald	Tusk	est	le	président	du
NPP	NPP	V	DET	NC	P+D

Conseil	européen	depuis	décembre	2014	.
NPP	ADJ	P	NC	NUM	PONCT

Problèmes de séquence

- Beaucoup de tâches en TAL ont des séquences en tant qu'entrée
- Le tâche est alors d'étiqueter chaque élément de la séquence

Reconnaissance d'entités nommées

Donald	Tusk	est	le	président	du
NPP	NPP	V	DET	NC	P+D
B-PER	I-PER	O	O	O	O
Conseil	européen	depuis	décembre	2014	.
NPP	ADJ	P	NC	NUM	PONCT
B-ORG	I-ORG	O	B-DAT	I-DAT	O

Reconnaissance d'entités nommées

- Une sous-tâche importante dans le TAL
- Trouver et classifier les noms dans un texte

Selon une étude publiée par l'Insee mercredi 28 octobre, l'écart entre le coût horaire de la main-d'œuvre à Paris et celui à Berlin se resserre.

Reconnaissance d'entités nommées

- Une sous-tâche importante dans le TAL
- Trouver et classifier les noms dans un texte

Selon une étude publiée par l'**Insee mercredi 28 octobre**, l'écart entre le coût horaire de la main-d'œuvre à **Paris** et celui à **Berlin** se resserre.

Reconnaissance d'entités nommées

- Une sous-tâche importante dans le TAL
- Trouver et **classifier** les noms dans un texte

Selon une étude publiée par l'**Insee** **mercredi 28 octobre**, l'écart entre le coût horaire de la main-d'œuvre à **Paris** et celui à **Berlin** se resserre.

- personne
- lieu
- date
- organization

Reconnaissance d'entités nommées

Codage IOB / BIO

publiée	O
par	O
l'	O
Insee	B-ORG
mercredi	B-DAT
28	I-DAT
octobre	I-DAT

- Avec un codage correct, la reconnaissance d'entités nommés devient une tâche d'étiquetage de séquence
- Inside, Outside, Beginning
- B-XX pour séparer les entités adjacentes du même type

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Marssac-sur-Tarn .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Marssac-sur-Tarn .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Marssac-sur-Tarn .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - **Sous-chaîne de caractères**
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Marssac-sur-Tarn .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Marssac-sur-Tarn .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier un accident s' est produit

à Xx-x-Xx .

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier	un	accident	s'	est	produit
ADV	DET	NC	CLR	V	VPP
à	Marssac-sur-Tarn	.			
P	NPP	PONCT			

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier	un		accident	s'	est	produit
ADV	DET		NC	CLR	V	VPP
O	O		O	O	O	O
à	Marssac-sur-Tarn	.				
P	NPP		PONCT			
O	?		O			

Définition manuelle de traits

- Mots
 - Mot actuel
 - Mots précédent/prochain (le contexte)
 - Sous-chaîne de caractères
 - Forme de mots
- Classifications linguistiques additionnelles
 - POS
- Contexte d'étiquettes
 - Étiquette précédente (ou bien la suivante)

Hier	un	accident	s'	est	produit
ADV	DET	NC	CLR	V	VPP
O	O	O	O	O	O
à	Marssac-sur-Tarn	.			
P	NPP	PONCT			
O	?	O			

Étiquetage de séquence

- L'étiquetage d'un mot en séquence est le même qu'une classification simple
- On a des étiquettes présumées pour les mots précédents
- On utilise les étiquettes présumées des données précédentes, ainsi que les traits des données observées (mots courants, précédents et futurs) pour prédire la meilleure étiquette actuelle

				Traits
produit	à	Marssac-sur-Tarn	.	w_0 Marssac-sur-Tarn
VPP	P	NPP	PONCT	w_{-1} à
O	O	?	?	w_{+1} .
				pos_0 NPP
				pos_{-1} P
				pos_{+1} PONCT
				$hasHyphen$ True
				$label_{-1}$ O

Maximum entropy markov model

- Problème: on fait une séquence de jugements locaux, sans prendre en compte l'optimalité globale
- On espère que les conditions locales donnent un optimum global
- Pour résoudre ce problème, on combine le classifieur d'entropie maximale avec un modèle Markov: maximum entropy hidden Markov (MEMM)
 - Au lieu d'émettre une seule réponse, le classifieur donne les n meilleures réponses
 - On utilise alors un modèle Markov pour calculer la meilleure séquence globale

Classifieur supervisé pour POS-tagging

Décision sur chaque mot en fonction de ses caractéristiques : **quels traits pourrions-nous utiliser ?**

Entrée	La	petite	fille	brise	la	glace
	Det	A	N	V	Det	N

Classifieur supervisé pour POS-tagging

Décision sur chaque mot en fonction de ses caractéristiques : **quels traits pourrions-nous utiliser ?**

Entrée	La	petite	fille	brise	la	glace
	Det	A	N	V	Det	N

- position (début/fin de phrase) : seulement certaines classes (e.g. dét, NP ..)
- majuscules: noms propres (sauf en début de phrase)
- préfixe: ultra- → adjectif
- suffixe: claire-ment → adverbe, -er → verbe

Classifieur supervisé

Le contexte est très informatif (cf bigrammes)

Entrée	La	petite	fille	brise	la	glace
	Det	A	N	V	Det	N

→ les décisions sur les étiquettes sont interdépendantes

$$P(\text{Det}|\text{Start}) > P(\text{N}|\text{Start})$$

$$P(\text{N}|A) > P(V|A)$$

$$P(V|N) > P(N|N)$$

→ modèles markoviens

On peut aussi se contenter des caractéristiques des mots (ou juste des mots) précédents (try it!)

Classificateurs séquentiels

- HMM: Hidden Markov Models
- CRF: Conditional Random Fields
- MEMM: Maximum Entropy Markov Models
- RNN: Recurrent Neural Network

Outils/Performances

Pour le français :

- TreeTagger (Schmidt) (HMM)
- Melt (Denis and Sagot, 2009) (MEMM) 97.80 acc

“Baseline”: tag le plus courant + "nom" pour les mots inconnus $\approx 90\%$

Pour l'anglais :

- Best (PTB): *Meta BiLSTM* (Bohnet et al., 2018) 97.96 acc.
- Best (Social media) : *ACE + fine-tune* (Wang et al., 2020) 93.4 acc.
- Best (UD) : *Multilingual BERT and BPEmb* (Heinzerling and Strube, 2019) 96.77 acc
- pré-NN $\approx 97.2\text{-}97.4$ acc

https://github.com/sebastianruder/NLP-progress/blob/master/english/part-of-speech_tagging.md

[https://aclweb.org/aclwiki/POS_Tagging_\(State_of_the_art\)](https://aclweb.org/aclwiki/POS_Tagging_(State_of_the_art))

Outils/Performances

For many languages : Spacy, Stanza, TranKit (Van Nguyen et al. 2021)

Treebank	System	Tokens	Sents.	Words	UPOS	XPOS	UFeats	Lemmas	UAS	LAS
Overall (90 treebanks)	Trankit	99.23	91.82	99.02	95.65	94.05	93.21	94.27	87.06	83.69
	Stanza	99.26	88.58	98.90	94.21	92.50	91.75	94.15	83.06	78.68
Arabic-PADT	Trankit	99.93	96.59	99.22	96.31	94.08	94.28	94.65	88.39	84.68
	Stanza	99.98	80.43	97.88	94.89	91.75	91.86	93.27	83.27	79.33
	UDPipe	99.98	82.09	94.58	90.36	84.00	84.16	88.46	72.67	68.14
Chinese-GSD	Trankit	97.01	99.7	97.01	94.21	94.02	96.59	97.01	85.19	82.54
	Stanza	92.83	98.80	92.83	89.12	88.93	92.11	92.83	72.88	69.82
	UDPipe	90.27	99.10	90.27	84.13	84.04	89.05	90.26	61.60	57.81
English-EWT	Trankit	98.48	88.35	98.48	95.95	95.71	96.26	96.84	90.14	87.96
	Stanza	99.01	81.13	99.01	95.40	95.12	96.11	97.21	86.22	83.59
	UDPipe	98.90	77.40	98.90	93.26	92.75	94.23	95.45	80.22	77.03
	spaCy	97.44	63.16	97.44	86.99	91.05	-	87.16	55.38	37.03
French-GSD	Trankit	99.7	96.63	99.66	97.85	-	97.16	97.80	94.00	92.34
	Stanza	99.68	94.92	99.48	97.30	-	96.72	97.64	91.38	89.05
	UDPipe	99.68	93.59	98.81	95.85	-	95.55	96.61	87.14	84.26
	spaCy	99.02	89.73	94.81	89.67	-	-	88.55	75.22	66.93
Spanish-Ancora	Trankit	99.94	99.13	99.93	99.02	98.94	98.8	99.17	94.11	92.41
	Stanza	99.98	99.07	99.98	98.78	98.67	98.59	99.19	92.21	90.01
	UDPipe	99.97	98.32	99.95	98.32	98.13	98.13	98.48	88.22	85.10
	spaCy	99.95	97.54	99.43	93.43	-	-	80.02	89.35	83.81

Paper TranKit (EACL démo): <https://aclanthology.org/2021.eacl-demos.10.pdf>

Démo trankit: <http://nlp.uoregon.edu/trankit>

Notebook TD2 :

- Expérimenter avec NLTK / Spacy pour obtenir des séquences de POS, voir ce qu'il se passe dans le cas d'ambiguïté

Dans le fichier discours_methode.txt :

- Extraire les séquences de N + Adj les plus fréquentes
- Extraire tous les noms propres (simplification de la tâche NER)

Les catégories morpho-syntaxiques sont utiles :

- pour l'analyse syntaxique ;
- pour l'extraction d'information ;
- comme information parmi d'autres pour des tâches diverses.

→ Et maintenant, la syntaxe

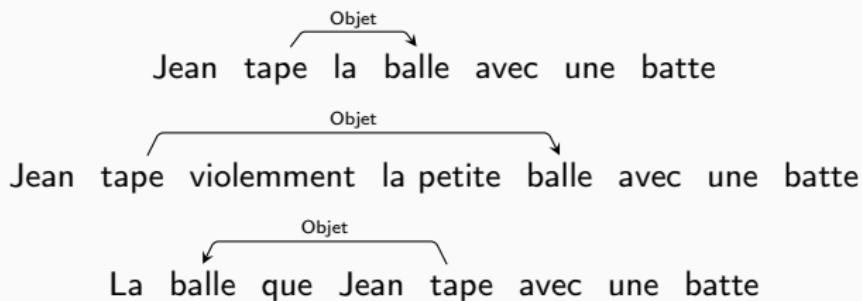
Catégories syntaxiques et Constituants

- Certains **mots** sont complètement substituables → même **POS** :
 - *Je vois un (chien) arriver/Je vois un (problème) arriver*
- Certains **groupes des mots** sont complètement substituables → même type de syntagme (*!phrase* en anglais):
 - *Je vois un (méchant gros chien) arriver/Je vois un (problème ennuyeux) arriver* (Syntagme nominal / NP)
 - *You (make something) / (fly).* (Syntagme verbal / VP)
 - *You make (very interesting and well designed) / (boring) presentations* (Syntagme adjectival / AdjP)

La **syntaxe** : pas seulement des contraintes sur l'ordre des mots
→ structure hiérarchique : grouper des mots en syntagme et des syntagmes en phrase.

Extraction d'information et syntaxe

Permet de mieux abstraire le lien (à comparer aux n-grammes)



Objectifs de la syntaxe

- Une langue naturelle a une structure interne.
- Le but de la syntaxe est de capturer cette structure.
- Plus précisément, les objectifs de la syntaxe incluent les suivants:
 - pouvoir décrire les structures des phrases possibles
 - pouvoir analyser les phrases en structures

Avec un vocabulaire fini : une infinité de phrases possibles, pas de 'lexique' des phrases vs les dictionnaires de mots

→ il faut trouver les règles de construction des phrases

Qu'est-ce qu'une phrase grammaticalement correcte ?

1. Ceci n'est pas une phrase.
2. N'est phrase pas ceci une.
3. La phrase précédente est fausse.
4. Cette phrase a mangé mon idée verte sans couleur qui dormait furieusement.

La grammaticalité d'une phrase ne vient pas:

- du contexte (cf 3)
- de sa vérité (cf 1)
- de sa compréhensibilité (cf 4)

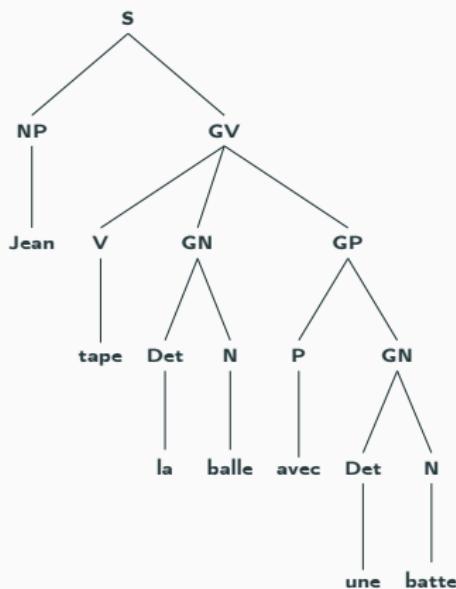
Deux représentations

Constituance vs Dépendance

Deux représentations

Constituance vs Dépendance

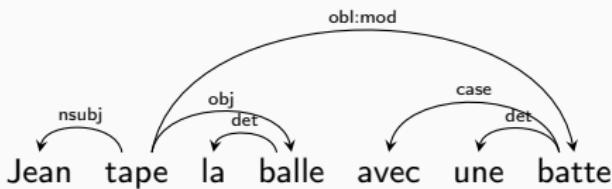
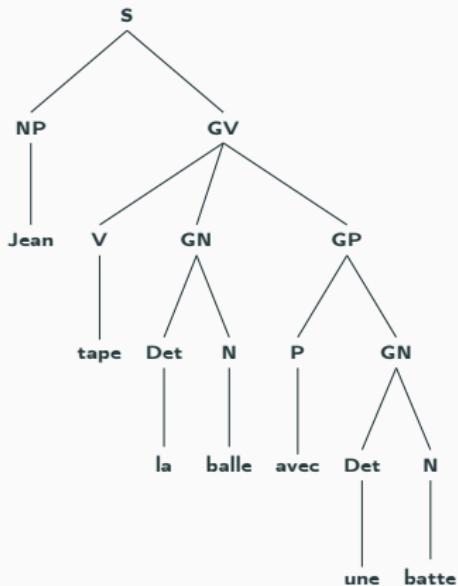
- grammaire de constituants



Deux représentations

Constituance vs Dépendance

- grammaire de constituants
- grammaire de dépendances (UD / Spacy)



Quelques types de constituants i

On identifie un constituant en faisant des tests de substitution :

Syntagme nominal - SN ou GN (Noun Phrase - NP)

- Noms propres, pronoms personnels, noms communs (NC)
 - ex. *il/Nicholas/l'ordinateur*
- NC accompagné d'un spécifieur (déterminant, cardinaux) et év. de *qualificateurs*
 - ex. *deux ordinateurs/l'ordinateur de Nicholas/le suspect présumé mort qui n'a pas encore été trouvé.*
- Expression qui réfère à une entité (un terme au sens logique)

Syntagme verbal - SV ou GV (Verb Phrase - VP)

- toute phrase possède un verbe principal
- ex. *saute de joie/manger à midi/ayant été déçu par sa performance*
- → Différentes formes (e.g. participe, infinitif), temps (e.g. présent, imparfait), modes verbaux (e.g. subjonctif, indicatif)
- Expression prédicative : événement ou état

Syntagme prépositionnel

- *pendant le repas/à droite du bâtiment*
- Direction, position dans le temps ou l'espace

Syntagme adjetival

- modifié par un adverbe : *tellement stupide*
- coordonné : *brillant et beau*
- avec un complément : *difficile à saisir*

Syntagme adverbial, e.g.:

- temporel : *dans une heure/quand j'aurais fini mon livre*
- de verbe : *il est rapidement parti*

Quelques types de constituants iv

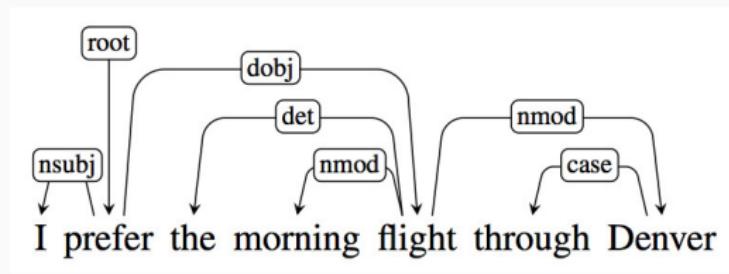
Un syntagme a une tête : V pour SV, N pour SN etc

- la tête dirige un constituant
- chaque tête peut être accompagnée de spécieurs, quantificateurs et compléments
- certains traits de la tête caractérisent tout le constituant
 - par ex: si le verbe est à l'infinitif tout le GV est infinitif, si le N est féminin singulier tout le GN est féminin singulier.

→ Analyse en Constituants Immédiats : construire l'arbre syntaxique à partir des mots + POS

L'analyse en dépendances [Jurafsky, chpt 14]

→ La structure syntaxique est ici décrite en termes de relations grammaticales binaires entre les mots directement



- Les relations sont représentées par des arcs dirigés étiquetés de la tête au dépendant
- On a un ensemble fixé de relations grammaticales de dépendance
- Un noeud racine (root) marque la racine de l'arbre, vue comme la tête de la structure

L'analyse en dépendances : pourquoi ?

- Cette structure encode directement des informations importantes qui sont plus cachées dans la structure en constituents
 - e.g. on 'voit' directement les arguments du verbe, crucial pour savoir qui fait quoi
 - dans l'exemple on a accès directement à des informations sur le 'vol' ('morning' et 'Denver' sont des dépendants directs de 'flight')
- Ces grammaires permettent de gérer les langues dont l'ordre des mots est (relativement) libre plus facilement
- On a déjà vu la notion de tête d'un syntagme : la tête est le mot central qui organise un constituant, les autres mots sont des dépendants directs ou indirects de leur tête
- On spécifie la fonction grammaticale du dépendant par rapport à sa tête, e.g. sujet, objet (direct ou indirect) etc, dépend des cadres théoriques / projets d'annotation

Le plus connu : Universal Dependencies (Nivre et al., 2016)

<https://universaldependencies.org/>

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	Description
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	Description
CONJ	Conjunct
CC	Coordinating conjunction

Figure 14.2 Some of the Universal Dependency relations (de Marneffe et al., 2014).

Des outils pour explorer des corpus UD, e.g. GREW : http:

//universal.grew.fr/?corpus=UD_Afrikaans-AfriBooms@2.10

Des outils pour parser du texte, e.g. UDPipe

<https://universaldependencies.org/tools.html#udpipe>

Syntaxe : notions additionnelles I

Différents types de phrases :

- Simple : *Alan a mangé la pomme.*
- Coordonnée : *Alan a mangé la pomme, mais elle était empoisonnée.*
- Complexe : *Charlie dit qu'Alan a mangé la pomme.*

Différents types de structures verbales :

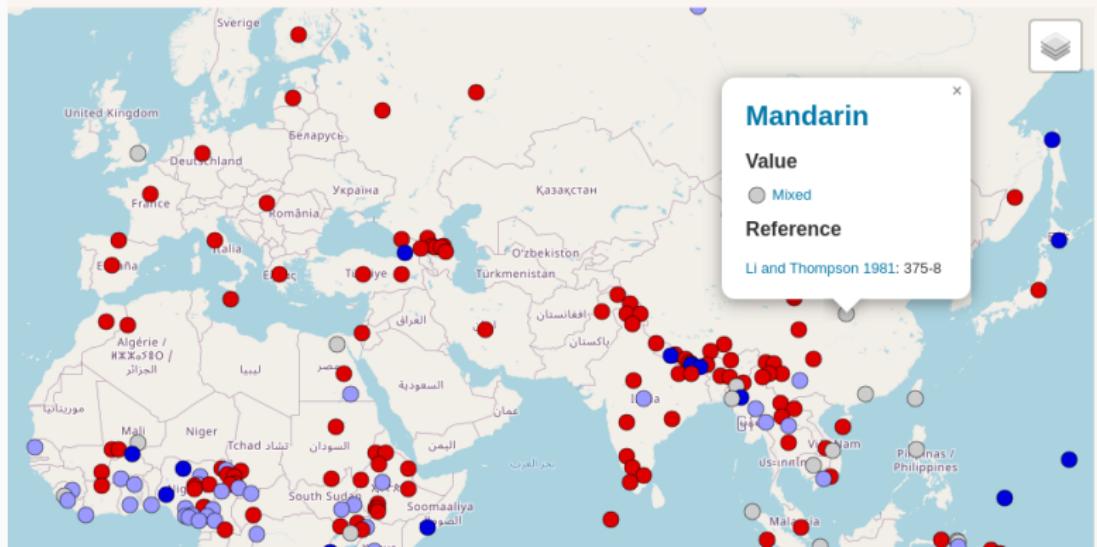
- intransitif : SN + V (verbe sans complément) *dormir, éternuer ...*
- transitif : SN + V + SN (verbe + complément) *donner, voir ...*
- ditransitif : SN + V + SN + SN (2 cplts directs) *give, tell, ...*
- autre : complément phrasique *croire que, demander si ...*
- Différences importantes selon les langues
 - français : *donner quelque chose à quelqu'un*
 - chinois : plusieurs constructions
 - Wǒ sòng tā yī běn shū. / I gave him/her a book.
 - Wǒ sòng yī běn shū gěi tā. / I gave a book to him/her.

Syntaxe : différences entre les langues du monde (*give*)

Values

●	Indirect-object construction	189
●	Double-object construction	84
●	Secondary-object construction	65
●	Mixed	40

GeoJSON ▾



Syntaxe : différences entre les langues du monde

Feature 81A: Order of Subject, Object and Verb



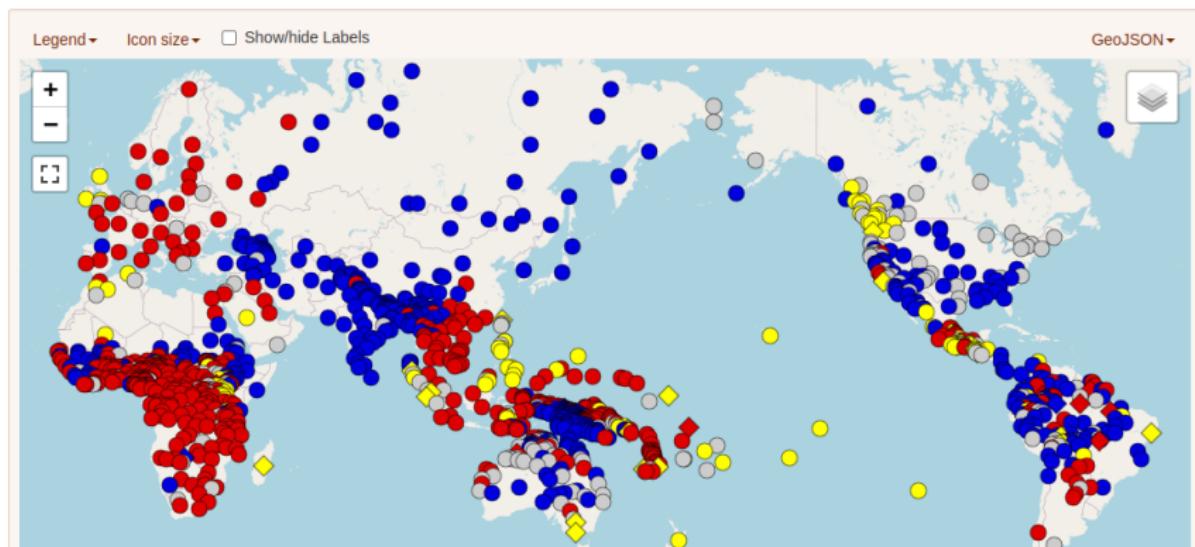
This feature is described in the text of chapter 81 [Order of Subject, Object and Verb](#) by Matthew S. Dryer

cite

You may combine this feature with another one. Start typing the feature name or number in the field below.

Values

● SOV	564
● SVO	488
● VSO	95
● VOS	25
● OVS	11
● OSV	4
● No dominant order	189



Syntaxe : notions additionnelles II

Arguments principaux vs arguments adjoints ou auxiliaires :

principaux centraux pour le sens du verbe, donc non-optionnels.

Montrent en général l'agent, le patient et l'objet de l'action. Autrement dit, sujet et objet direct ou indirect.

adjoints Optionnels; montrent le manière, le temps, le lieu, etc

- Le nombre exact des arguments principaux et adjoints dépend de chaque verbe. Il y a des approches différentes pour formaliser l'étude des arguments des verbes et leur relations : ceci est plus de nature sémantique.
- En linguistique, la structure des arguments d'un verbe est aussi connue sous le nom de **valence**.
- Se généralise au moins au nom:
 X construit $Y \rightarrow$ la construction de X par Y .

Dépendances à longue distance :

Quel est le sujet du 2e verbe ?

- Je vois l'homme qui a tiré sur le président

Dépendances à longue distance :

Quel est le sujet du 2e verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure

Dépendances à longue distance :

Quel est le sujet du 2e verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure

Syntaxe : notions additionnelles III

Dépendances à longue distance :

Quel est le sujet du 2e verbe ? le 2 verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure
- Les Français aiment le fromage, les Néerlandais aussi.

Syntaxe : notions additionnelles III

Dépendances à longue distance :

Quel est le sujet du 2e verbe ? le 2 verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure
- Les Français aiment le fromage, les Néerlandais aussi.

1. relative: NP V NP₁ qui S'

S' ≈ NP₁ VP

Syntaxe : notions additionnelles III

Dépendances à longue distance :

Quel est le sujet du 2e verbe ? le 2 verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure
- Les Français aiment le fromage, les Néerlandais aussi.

1. relative: NP V NP₁ qui S'

$S' \approx NP_1 VP$

2. "montée" du sujet: NP₁ V P NP₂ de S'

$S' \approx NP_1 VP(infin.)$

Syntaxe : notions additionnelles III

Dépendances à longue distance :

Quel est le sujet du 2e verbe ? le 2 verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure
- Les Français aiment le fromage, les Néerlandais aussi.

1. relative: NP V NP₁ qui S' $S' \approx NP_1 VP$
2. "montée" du sujet: NP₁ V P NP₂ de S' $S' \approx NP_1 VP(infin.)$
3. "montée" du complément: $S' \approx NP_2 VP(infin.)$

Syntaxe : notions additionnelles III

Dépendances à longue distance :

Quel est le sujet du 2e verbe ? le 2 verbe ?

- Je vois l'homme qui a tiré sur le président
- Jean promet à Marie d'être à l'heure
- Jean reproche à Marie d'être à l'heure
- Les Français aiment le fromage, les Néerlandais aussi.

- | | |
|-----------------------------------------------------------------|--------------------------------------|
| 1. relative: NP V NP ₁ qui S' | S' ≈ NP ₁ VP |
| 2. "montée" du sujet: NP ₁ V P NP ₂ de S' | S' ≈ NP ₁ VP(infin.) |
| 3. "montée" du complément: | S' ≈ NP ₂ VP(infin.) |
| 4. ellipse: NP ₁ VP ₁ , s' | S' ≈ NP ₂ VP ₁ |

- Comment peut-on donc formellement décrire la structure interne d'une phrase ?

Grammaires formelles i

Formellement, une grammaire est un quadruplet $G = (V_N, V_T, P, S_0)$
où:

1. V_N est l'ensemble des symboles **non-terminaux**. Ils peuvent être décomposés.
e.g. : les catégories verbes ou noms.
2. V_T est un ensemble des symboles **terminaux**. Ils ne peuvent pas être décomposés. On les enregistre dans un **lexique**.
e.g. : les mots salut, moi, pense, etc.
3. P est un ensemble des règles grammaticales (ou productions)
Ces règles prennent souvent la forme : $\alpha \rightarrow \beta$ où α, β sont des séquences des symboles sur l'ensemble $(V_N \cup V_T)$, et α contient au moins un symbole non-terminal.
4. $S_0 \in V_N$ est le symbole du début (start symbol), qui commence les séquences des dérivations.

La hiérarchie des grammaires selon Chomsky

Type de grammaire	Automate équivalent	Forme des productions
Linear Grammars Grammaires linéaires	Finite State Automata Automates à états finis	(1) $A \rightarrow \alpha$ (2) $A \rightarrow \alpha B$ (3) $A \rightarrow B\alpha$ $A, B \in V_N$ et $\alpha \in V_T$
Context-Free Grammars (CFG) Grammaires hors contexte	Pushdown Automata Automates à piles	$A \rightarrow \chi$ $A \in V_N$ et $\chi \in V_T \cup V_N \cup \{\epsilon\}$ * à droite, séquence arbitraire * permet la récursivité.
Context Sensitive Grammars Grammaires sensitives au contexte	Linear-Bounded Automata ?	$\alpha \rightarrow \beta$ $\alpha, \beta \in V_N \cup V_T \wedge \alpha \leq \beta $ Souvent : $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, où $\beta \neq \epsilon$
Unrestrictive Grammars (en) Grammaires sans restrictions	Turing Machines Machines de Turing	$\alpha \rightarrow \beta$ where $\alpha \neq \epsilon$

- Insuffisante pour la langue naturelle

La hiérarchie des grammaires selon Chomsky

Type de grammaire	Automate équivalent	Forme des productions
Linear Grammars Grammaires linéaires	Finite State Automata Automates à états finis	(1) $A \rightarrow \alpha$ (2) $A \rightarrow \alpha B$ (3) $A \rightarrow B\alpha$ $A, B \in V_N$ et $\alpha \in V_T$
Context-Free Grammars (CFG) Grammaires hors contexte	Pushdown Automata Automates à piles	$A \rightarrow \chi$ $A \in V_N$ et $\chi \in V_T \cup V_N \cup \{\epsilon\}$ * à droite, séquence arbitraire * permet la récursivité.
Context Sensitive Grammars Grammaires sensitives au contexte	Linear-Bounded Automata ?	$\alpha \rightarrow \beta$ $\alpha, \beta \in V_N \cup V_T \wedge \alpha \leq \beta $ Souvent : $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, où $\beta \neq \epsilon$
Unrestrictive Grammars (en) Grammaires sans restrictions	Turing Machines Machines de Turing	$\alpha \rightarrow \beta$ where $\alpha \neq \epsilon$

- Insuffisante pour la langue naturelle
- Récursivité nécessaire

La hiérarchie des grammaires selon Chomsky

Type de grammaire	Automate équivalent	Forme des productions
Linear Grammars Grammaires linéaires	Finite State Automata Automates à états finis	(1) $A \rightarrow \alpha$ (2) $A \rightarrow \alpha B$ (3) $A \rightarrow B\alpha$ $A, B \in V_N$ et $\alpha \in V_T$
Context-Free Grammars (CFG) Grammaires hors contexte	Pushdown Automata Automates à piles	$A \rightarrow \chi$ $A \in V_N$ et $\chi \in V_T \cup V_N \cup \{\epsilon\}$ * à droite, séquence arbitraire * permet la récursivité.
Context Sensitive Grammars Grammaires sensitives au contexte	Linear-Bounded Automata ?	$\alpha \rightarrow \beta$ $\alpha, \beta \in V_N \cup V_T \wedge \alpha \leq \beta $ Souvent : $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$ où $\beta \neq \epsilon$
Unrestrictive Grammars (en) Grammaires sans restrictions	Turing Machines Machines de Turing	$\alpha \rightarrow \beta$ where $\alpha \neq \epsilon$

- Insuffisante pour la langue naturelle
- Récursivité nécessaire
- Notion de contexte dans le langage, mais CSG "too powerful"

La hiérarchie des grammaires selon Chomsky

Type de grammaire	Automate équivalent	Forme des productions
Linear Grammars Grammaires linéaires	Finite State Automata Automates à états finis	(1) $A \rightarrow \alpha$ (2) $A \rightarrow \alpha B$ (3) $A \rightarrow B\alpha$ $A, B \in V_N$ et $\alpha \in V_T$
Context-Free Grammars (CFG) Grammaires hors contexte	Pushdown Automata Automates à piles	$A \rightarrow \chi$ $A \in V_N$ et $\chi \in V_T \cup V_N \cup \{\epsilon\}$ * à droite, séquence arbitraire * permet la récursivité.
Context Sensitive Grammars Grammaires sensitives au contexte	Linear-Bounded Automata ?	$\alpha \rightarrow \beta$ $\alpha, \beta \in V_N \cup V_T \wedge \alpha \leq \beta $ Souvent : $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, où $\beta \neq \epsilon$
Unrestrictive Grammars (en) Grammaires sans restrictions	Turing Machines Machines de Turing	$\alpha \rightarrow \beta$ where $\alpha \neq \epsilon$

- Insuffisante pour la langue naturelle
- Récursivité nécessaire → utilisée en NLP pour l'analyse syntaxique
- Notion de contexte dans le langage, mais CSG "too powerful"

Exemple

Règles de productions

S	→	NP VP
NP	→	DET N
NP	→	DET A N
NP	→	NP PP
NP	→	PRO
VP	→	V
VP	→	V NP
VP	→	VP PP
PP	→	P NP

Lexique

DET	→	le
DET	→	la
N	→	chien
N	→	chat
V	→	mange
V	→	dort
PRO	→	la
A	→	petit
A	→	noir
P	→	du

- Cette grammaire peut produire une phrase comme : *Le chat dort*

Exemple

Règles de productions

S	→	NP VP
NP	→	DET N
NP	→	DET A N
NP	→	NP PP
NP	→	PRO
VP	→	V
VP	→	V NP
VP	→	VP PP
PP	→	P NP

Lexique

DET	→	le
DET	→	la
N	→	chien
N	→	chat
V	→	mange
V	→	dort
PRO	→	la
A	→	petit
A	→	noir
P	→	du

- Cette grammaire peut produire une phrase comme : *Le chat dort*
- Donner 1 autre phrase que cette grammaire peut produire avec un verbe transitif

Exemple

Règles de productions

S	\rightarrow	NP VP
NP	\rightarrow	DET N
NP	\rightarrow	DET A N
NP	\rightarrow	NP PP
NP	\rightarrow	PRO
VP	\rightarrow	V
VP	\rightarrow	V NP
VP	\rightarrow	VP PP
PP	\rightarrow	P NP

Lexique

DET	\rightarrow	le
DET	\rightarrow	la
N	\rightarrow	chien
N	\rightarrow	chat
V	\rightarrow	mange
V	\rightarrow	dort
PRO	\rightarrow	la
A	\rightarrow	petit
A	\rightarrow	noir
P	\rightarrow	du

- Cette grammaire peut produire une phrase comme : *Le chat dort*
- Donner 1 autre phrase que cette grammaire peut produire avec un verbe transitif : *Le chat mange le chien*

Exemple

Règles de productions

$$S \rightarrow NP VP$$

$$NP \rightarrow DET N$$

$$NP \rightarrow DET A N$$

$$NP \rightarrow NP PP$$

$$NP \rightarrow PRO$$

$$VP \rightarrow V$$

$$VP \rightarrow V NP$$

$$VP \rightarrow VP PP$$

$$PP \rightarrow P NP$$

Lexique

$$DET \rightarrow le$$

$$DET \rightarrow la$$

$$N \rightarrow chien$$

$$N \rightarrow chat$$

$$V \rightarrow mange$$

$$V \rightarrow dort$$

$$PRO \rightarrow la$$

$$A \rightarrow petit$$

$$A \rightarrow noir$$

$$P \rightarrow du$$

- Cette grammaire peut produire une phrase comme : *Le chat dort*
- Donner 1 autre phrase que cette grammaire peut produire avec un verbe transitif : *Le chat mange le chien*
- Est-ce que cette grammaire peut produire des phrases non grammaticales ?

Exemple

Règles de productions

S	→	NP VP
NP	→	DET N
NP	→	DET A N
NP	→	NP PP
NP	→	PRO
VP	→	V
VP	→	V NP
VP	→	VP PP
PP	→	P NP

Lexique

DET	→	le
DET	→	la
N	→	chien
N	→	chat
V	→	mange
V	→	dort
PRO	→	la
A	→	petit
A	→	noir
P	→	du

- Cette grammaire peut produire une phrase comme : *Le chat dort*
- Donner 1 autre phrase que cette grammaire peut produire avec un verbe transitif : *Le chat mange le chien*
- Est-ce que cette grammaire peut produire des phrases non grammaticales ? e.g. **la chien dort / *le noir chat dort / *Le chien dort le chat*

Exemple

Règles de productions

S	→	NP VP
NP	→	DET N
NP	→	DET A N
NP	→	NP PP
NP	→	PRO
VP	→	V
VP	→	V NP
VP	→	VP PP
PP	→	P NP

Lexique

DET	→	le
DET	→	la
N	→	chien
N	→	chat
V	→	mange
V	→	dort
PRO	→	la
A	→	petit
A	→	noir
P	→	du

- Cette grammaire peut produire une phrase comme : *Le chat dort*
- Donner 1 autre phrase que cette grammaire peut produire avec un verbe transitif : *Le chat mange le chien*
- Est-ce que cette grammaire peut produire des phrases non grammaticales ? e.g. **la chien dort / *le noir chat dort / *Le chien dort le chat*

→ CFG pas assez contraintes

Reconnaissance et analyse

Étant données une grammaire hors contexte et une phrase (chaîne des mots) on peut voir 2 problèmes:

1. Reconnaissance : La phrase est-elle admise par la grammaire ? (est-elle syntaxiquement correcte?)
2. Analyse/Parsing : Si elle est syntaxiquement correcte, quelles sont les (ou la) structures syntaxiques différentes ?

Exemple d'application "moderne" : contrôler les sorties d'un modèle génératif :

Grammar-Constrained Decoding for Structured NLP Tasks without Finetuning (Geng et al., EMNLP 2023)

Reconnaissance et analyse

Étant données une grammaire hors contexte et une phrase (chaîne des mots) on peut voir 2 problèmes:

1. Reconnaissance : La phrase est-elle admise par la grammaire ?
(est-elle syntaxiquement correcte?)
2. Analyse/Parsing : Si elle est syntaxiquement correcte, quelles sont les (ou la) structures syntaxiques différentes ?
 - c'est la chaîne d'applications des règles qui nous donne la structure,
e.g. $S \rightarrow NP\ VP$ c'est le haut de l'arbre en constituant etc
 - L'arbre dérivé (**parse tree**) encode les étapes de cette dérivation
(mais on perd l'ordre), les noeuds internes sont les non terminaux,
les feuilles nous donnent la phrase
 - Le but du parsing : construire l'arbre dérivé à partir d'une séquence
de tokens

Exemple d'application "moderne" : contrôler les sorties d'un modèle génératif :

Grammar-Constrained Decoding for Structured NLP Tasks without Finetuning (Geng et al., EMNLP 2023)

Syntactic Parsing / Analyse syntaxique automatique

En pratique 2 approches possibles :

- Top-Down : on commence au symbole S et on essaye de deviner les dérivations permettant d'obtenir la phrase
- **Bottom-up : on part des tokens et on essaye d'appliquer des règles pour remonter à S**

Encore quelques approches symboliques :

- e.g. Tree Adjoining Grammars (grammaires de 'bouts d'arbres' + systèmes de traits)
- parfois systèmes hybrides incluant des règles pour des cas spécifiques

Surtout des approches statistiques :

- on utilise un corpus annoté / 'arboré' contenant l'annotation d'arbres syntaxiques
 - Format courant d'annotation 'bracketing' e.g. (S (NP (Det Le) (N Chat)) (VP (V dort)))
- Algorithmes type transition-based e.g. Shift-Reduce [Aho Ullman, 1977]

Shift-reduce

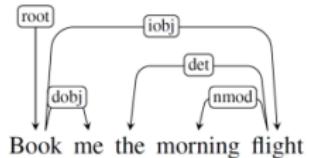
Construit l'arbre de manière incrémentale, de gauche à droite (selon les langues ...).

Transition	Stack	Buffer	Added subtree
Shift	[]	[The, ... , .]	
Shift	[The]	[public, ... , .]	
Reduce-Left-NP	[The, public]	[is, ... , .]	
Shift	[NP]	[is, ... , .]	
Shift	[NP, is]	[still, ... , .]	
Shift	[NP, is, still]	[cautious , .]	
Reduce-Unary-ADVP	[NP, is, ADVP]	[cautious , .]	ADVP→ still
Reduce-Right-VP*	[NP, VP*]	[cautious , .]	VP*→is ADVP
Shift	[NP, VP*, cautious]	[.]	
Reduce-Unary-ADJP	[NP, VP*, ADJP]	[.]	ADJP→ cautious
Reduce-Right-VP	[NP, VP]	[.]	VP→VP* ADJP
Shift	[NP, VP, .]	[]	
Reduce-Right-S*	[NP, S*]	[]	S*→VP .
Reduce-Left-S	[S]	[]	S→NP S*
Finish	[]	[]	

<https://arxiv.org/pdf/1804.07961.pdf>

Shift-reduce

Pareil en dépendances



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)
10	[root]	[]	Done	

Figure 14.7 Trace of a transition-based parse.

- Approches récentes : linéariser l’arbre en constituants et appliquer des modèles seq2seq

Etat de l'art

Anglais

http://nlpprogress.com/english/constituency_parsing.html :

- Constituants : Span Attention + XLNet (Tian et al., 2020) 96.40
- Dépendances : Label Attention Layer + HPSG + XLNet (Mrini et al., 2019) 96.26

UD shared task, résultats par corpus

<https://universaldependencies.org/conll18/results-las.html>

:

- Français (Sequoia) 89.9
- Français oral 75.8
- Féroïen : 49.4

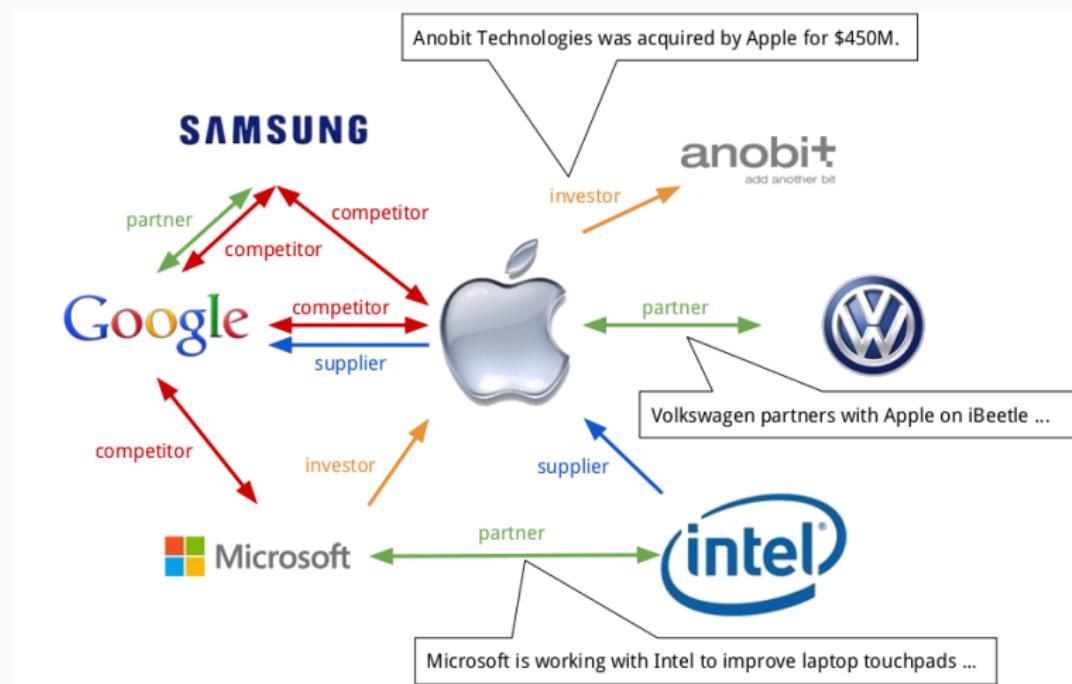
Retour au notebook TD2 : exemples de parsing avec des grammaires

Application : Extraction de relations

- on a déjà vu un peu d'extraction d'information (entités) aux cours 1-2, surtout à base de patrons lexicaux (mots)
- on va maintenant voir l'intérêt d'avoir en plus des informations syntaxiques
- et on va étendre le problème : on veut maintenant extraire des entités, et des relations entre entités

Méthodes d'extraction

Pourquoi



Méthodes d'extraction

- Patterns créés manuellement
- Approche supervisée
- Méthodes de bootstrap
- Méthodes de supervision distantes
- Méthodes non-supervisées

Patterns créés manuellement

```
; ; For <company> appoints <person> <position>

(defpattern appoint
  "np-sem(C-company)? rn? sa? vg(C-appoint) np-sem(C-person) ',,'?
  to-be? np(C-position) to-succeed?:
  company-at=1.attributes, sa=3.span, lv=4.span, person-at=5.attributes,
  position-at=8.attributes |
  ...
  (defun when-appoint (phrase-type)
    (let ((person-at (binding 'person-at))
          (company-entity (entity-bound 'company-at))
          (person-entity (essential-entity-bound 'person-at 'C-person))
          (position-entity (entity-bound 'position-at))
          (predecessor-entity (entity-bound 'predecessor-at))
          new-event)
      (not-an-antecedent position-entity)
      ;; if no company is specified for position, use agent
    ...
  )
```

Patterns pour hyponymes

On reprend Hearst (1992)

- Exemple :
 - *L'agar agar est une substance préparée à partir d'un mélange d'algues rouges, telles que Gelidium, pour le laboratoire ou usage industriel.*
 - Qu'est-ce que le Gelidium ?
 - Comment on le sait ?

Patterns pour hyponymes

On reprend Hearst (1992)

- Exemple :
 - *L'agar agar est une substance préparée à partir d'un mélange d'algues rouges, telles que Gelidium, pour le laboratoire ou usage industriel.*
 - Qu'est-ce que le Gelidium ?
 - Comment on le sait ?

Patterns lexico-syntaxiques de Hearst

- Ys such as X ((, X)* (, and/or) X)
- such Ys as X ...
- X ... or other Ys
- X ... and other Ys
- Ys including X ...
- Ys, especially X ...

Exemples: ‘en particulier’

- Le monde des paparazzi semble avoir pris pour victime plusieurs célébrités, en particulier Miley Cyrus.
- Ces enfants stars, en particulier Miley Cyrus, je pense que vous devez mettre la faute sur les médias
- Les célébrités se sont bien amusées en sirotant des cocktails, en particulier Miley Cyrus que tout le monde semble trouver si exceptionnelle.

avec entités nommées

- Intuition: relation spécifique souvent entre entités spécifiques
 - located-in(organization, location)
 - founded(person, organization)
 - cures(drug, disease)
- Prérequis: étiquettagage d'entités nommées pour aider à l'extraction de relations

Patterns: inconvénients

- Création manuelle pour chaque relation : laborieux
 - spécifique aux langues
 - spécifique au domaine
 - difficile à maintenir
- On ne veut pas le faire pour toutes relations possibles
- Et on veut une meilleure performance
 - e.g. Hearst : 66% exactitude (accuracy) pour extraction de hyponymes

Approche supervisée

- Choix d'un ensemble de relations qu'on veut extraire
- Choix d'un ensemble d'entités nommées pertinentes
- Etiquettagement de données (corpus)
 - Choix d'un corpus représentatif
 - Etiquettagement d'entités nommées dans le corpus (de manière automatique)
 - Classification manuelle des relations entre entités
 - Séparer les données en train, development, test
- Entraînement d'un classifieur sur corpus d'entraînement

Comment faire la classification ?

- Trouver tous les paires d'entités nommées (souvent dans la même phrase)
- Décider si 2 entités sont reliées
- Si oui, classifier la relation
- Pourquoi étape additionnelle ?
 - Classification plus rapide : beaucoup de paires sont éliminés
 - Utilisation de différents ensembles de traits ciblés pour chaque tâche

Définition de traits

“Uber a l’ intention de commencer par faire rouler des voitures sur un court trajet entre deux locaux de la société à Pittsburgh, a déclaré sa porte-parole Sarah Abboud .”

- Founder ?
- Subsidiary ?
- Employee ?
- Inventor ?

Définition de traits

Traits de mots

“Uber a l'intention de commencer par faire rouler des voitures sur un court trajet entre deux locaux de la société à Pittsburgh, a déclaré sa porte-parole Sarah Abboud.”

- Mots principaux de M1 et M2, ainsi que la combinaison
Uber Abboud Uber-Abboud
- ‘sac de mots’ et bigrammes en M1 et M2
{ Uber, Sarah, Abboud, Sarah Abboud }
- Mots ou bigrammes en positions particulières à gauche et à droite de M1/M2
M2:-1 porte-parole
M1:+1 a
- ‘sac de mots’ ou bigrammes entre 2 entités
{ a, l', intention, de, commencer, . . . , Pittsburgh, déclaré, voiture }

Définition de traits

Traits basés sur entités nommées

“Uber a l'intention de commencer par faire rouler des voitures sur un court trajet entre deux locaux de la société à Pittsburgh, a déclaré sa porte-parole Sarah Abboud.”

- Types d'entités nommées
 - M1: ORG
 - M2: PERSON
- Concaténation des deux types
ORG-PERSON
- Niveau d'entité de M1/M2 (NAME, NOMINAL, PRONOUN)
 - M1: NAME
 - M2: NAME

Traits à base de l'arbre syntaxique

“Uber a l'intention de commencer par faire rouler des voitures sur un court trajet entre deux locaux de la société à Pittsburgh, a déclaré sa porte-parole Sarah Abboud.”

- Séquence de morceaux syntaxiques de l'un à l'autre
NP VP VPinf PP PP PP VP NP
- Chemin de constituents à travers l'arbre syntaxique de l'un à l'autre
NP \uparrow S \uparrow S \downarrow NP
- Chemin de dépendances
Uber – a – déclaré – Abboud

Traits de gazetteer / mots de déclenchement

“Uber a l’intention de commencer par faire rouler des voitures sur un court trajet entre deux locaux de la société à Pittsburgh, a déclaré sa porte-parole Sarah Abboud.”

- liste de déclenchement, e.g. *termes de parenté* (parent, femme, grandparent, soeur, ...)
- gazetteer : liste de noms géographiques/géopolitiques, e.g. *pays, villes*
+CITY

Définition de traits

“American Airlines, a unit of AMR, immediately matched the move, spokesman Tim Wagner said.”

Entity-based features

Entity ₁ type	ORG
Entity ₁ head	<i>airlines</i>
Entity ₂ type	PERS
Entity ₂ head	<i>Wagner</i>
Concatenated types	ORGPERS

Word-based features

Between-entity bag of words	{ <i>a, unit, of, AMR, Inc., immediately, matched, the, move, spokesman</i> }
Word(s) before Entity ₁	NONE
Word(s) after Entity ₂	<i>said</i>

Syntactic features

Constituent path	$NP \uparrow NP \uparrow S \uparrow S \downarrow NP$
Base syntactic chunk path	$NP \rightarrow NP \rightarrow PP \rightarrow NP \rightarrow VP \rightarrow NP \rightarrow NP$
Typed-dependency path	<i>Airlines</i> \leftarrow_{subj} <i>matched</i> \leftarrow_{comp} <i>said</i> \rightarrow_{subj} <i>Wagner</i>

Choix de classifieur

- Choix d'un classifieur selon préférence :
 - Régression logistique
 - Naïve Bayes
 - Support Vector Machines
 - ...
 - ou réseaux de neurones
- Entraîner sur ensemble d'entraînement, finetuner sur ensemble de développement, tester sur ensemble de test

Evaluation d'extraction de relations supervisé

- Calculer précision/rappel/ F_1 pour chaque relation
- $P = \frac{\# \text{ relations extraits correctement}}{\# \text{ totale de relations extraits}}$
- $R = \frac{\# \text{ relations extraits correctement}}{\# \text{ totale de relations gold}}$
- $F_1 = \frac{2PR}{P+R}$

Résumé

- (+) Bonne performance, avec grand corpus d'entraînement, et test similaire à train
- (-) Annoter est laborieux et coûteux
- (-) Difficile de généraliser à d'autres genres

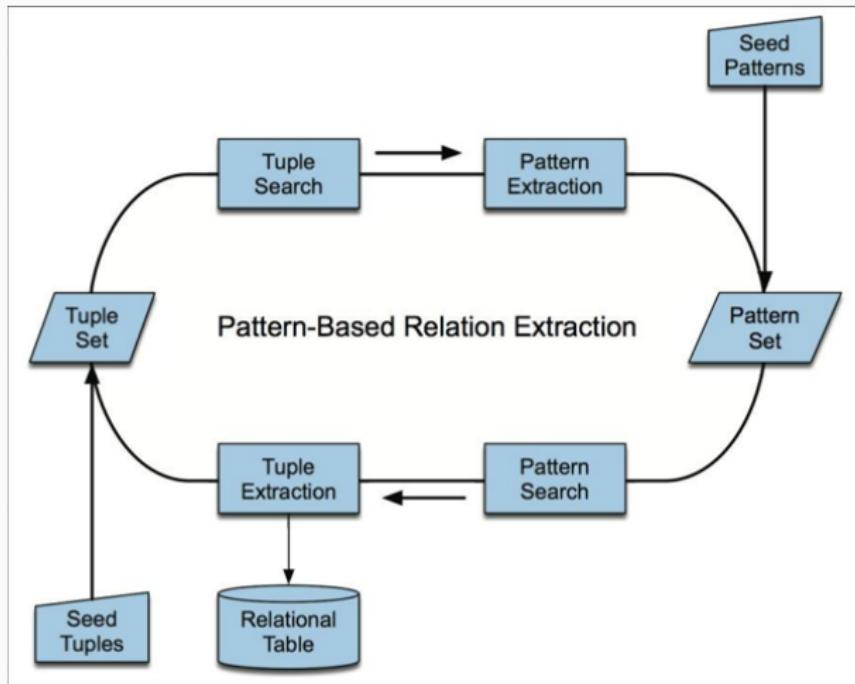
Méthodes de bootstrap

- Et si on n'a pas de données d'entraînement ?
- Peut-être on a
 - quelques exemples clés (seed tuples)
 - quelques patterns avec haute précision
 - des grands volumes de texte non-annoté
- → méthode de bootstrap : apprendre automatiquement à remplir la relation

Bootstrap: exemple

- Relation cible : lieu de sépulture
- Seed tuple : [*Johnny Halliday, Saint Barthélémy*]
- Trouver toutes instances pour *Johnny Halliday* et *Saint Barthélémy*
 - la tombe de Johnny Hallyday à Saint-Barthélemy
 - Johnny Hallyday a été enterré à Saint-Barthélemy
 - Johnny Halliday repose au cimetière à Saint- Barthélémy
- Utiliser les patterns pour trouver des nouveaux tuples
- Itérer

Bootstrap



- Extraction automatique de paires (*auteur, livre*) du web
- On commence avec 5 paires:

Author	Book
Isaac Asimov	The Robots of Dawn
David Brin	Startide Rising
James Gleick	Chaos: Making a New Science
Charles Dickens	Great Expectations
William Shakespeare	The Comedy of Errors

- On apprend des patterns :

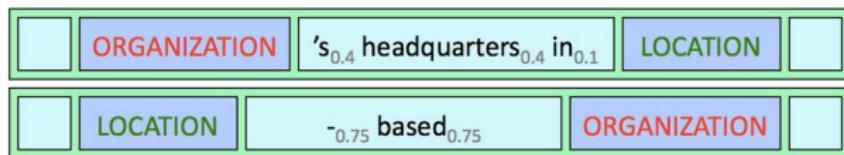
URL Prefix	Text Pattern
<code>www.sff.net/locus/c.*</code> <code>dns.city-net.com/~lmann/awards/hugos/1984.html</code> <code>dolphin.upenn.edu/~dcummins/texts/sf-award.htm</code>	<code>title by author (</code> <code><i>title</i> by author (</code> <code>author title (</code>

- On utilise les patterns pour récolter plus d'instances et patterns

Snowball (Agichtein & Gravano 2000)

- Algorithme itératif similaire
- Mais on exige que X et Y soient des entités nommées
- Et on calcule une valeur de confiance pour chaque pattern

Organization	Location of Headquarters
Microsoft	Redmond
Exxon	Irving
IBM	Armonk
Boeing	Seattle
Intel	Santa Clara



Inconvénients

- Sensible à l'ensemble de tuples initial
- Problème de déviation ('semantic drift') à chaque itération
- Précision parfois faible

- Hypothèse : Si deux entités appartiennent à une certaine relation, toute phrase contenant ces deux entités est susceptible d'exprimer cette relation
- Idée clé: utiliser une base de données de relations pour obtenir beaucoup d'exemples d'entraînement
 - au lieu de créer quelques tuples “graines” (seed) à la main (bootstrap)
 - au lieu d'utiliser un corpus annoté manuellement (supervisé)

Supervision distante

1. Pour chaque relation

Born-In

2. Pour chaque paire dans grande base de données

<Albert Einstein, Ulm> ; <Marie Curie, Varsovie>

3. Trouver phrases dans grand corpus avec les deux entités

Marie Curie naît à Varsovie

Albert Einstein, né (1879) à Ulm

4. Extraire les traits fréquents (arbre syntaxique, mots, ...)

PER naît à LOC

PER, né (XXXX) à LOC

5. Entraînement d'un classifieur supervisé avec des milliers de patterns

- Avantages d'une approche supervisée
 - on exploite des connaissances riches et fiables créées à la main
 - les relations ont des noms canoniques
 - on peut utiliser des traits avancés (e.g. traits syntaxiques)
- Avantages d'une approche non-supervisée
 - on peut exploiter des quantités illimitées de texte
 - permet un très grand nombre de traits faibles
 - non sensible au corpus de d'entraînement: indépendant du genre

Supervision distante : à retenir

- approche de supervision distante utilise une base de données avec instances de relation connues comme source de supervision.
- on classifie des paires d'entités, et non des mentions de paires d'entités.
- les traits d'une paire d'entités décrivent les patterns dans lequel les deux entités ont été vus à travers de nombreux phrases dans un grand corpus
- on peut utiliser 100 fois, même 1000 fois plus de données que dans le paradigme supervisé.
- idée généralisable à tout moyen qui permet automatiquement d'avoir des annotations, même plus ou moins bruitées
(→ on verra plus tard avec des modèles génératifs)

Approche non-supervisée

- ‘open information extraction’ : extraction de relations depuis le web, sans données d’entraînement, sans liste de relations
 1. On utilise des données syntaxiques pour entraîner un classifieur pour identifier des ‘tuples fiables’
 2. On extrait toutes relations entre NP, on les garde si fiable
 3. On évalue la qualité des relations (ordonnance) à base de redondance

Problème de synonymie

- Extraction de tuples synonymiques
 - (pont aérien, soulage, crise de la faim)
 - (crise de la faim, est atténué par, pont aérien)
 - (pont aérien, aide à résoudre, crise de la faim)
 - (pont aérien, allège, crise de la faim)
- On a appris quatre faits, ou un seul ?
- Comment identifier/combiner relations synonymiques ?

On retrouve les problèmes de

- similarité sémantique
- normalisation (cf Ontologies)

Approche neuronale ?

Domaine très actif !

Mais on verra plus tard

Démonstration/Exercice 1/3

Extraction d'information avec structure syntaxique

<https://spike.apps.allenai.org/>

Prise en main avec le tutoriel: requêtes simples Sélectionnez le corpus "Wikipedia for IE" (IE=information extraction)

- Lisez et appliquer les exemples du tutoriel "basic queries"
- Suivez la même démarche pour extraire des liens entre groupes musicaux et leur membres. Sauvez le résultat en csv depuis l'interface.
- On peut facilement voir si les résultats retournés sont corrects ou non (analysez vos résultats dans le fichier csv) mais il est beaucoup plus dur de voir si on manque quelque chose. Pour cela on peut tester avec quelques exemples particuliers: ajoutez "Pink Floyd" à votre dernière requête pour voir si les informations liées à ce groupe sont bien capturées. Que se passe-t-il ici ?

Exemple de pattern : <E>band:e=ORG band member <E>member:e=PERSON

Démonstration/Exercice 2/3

Prise en main avec le tutoriel: requêtes séquentielles

Les requêtes séquentielles permettent d'écrire des patrons où l'on restreint les résultats avec un ordre possible des mots de la requête

- Lisez et appliquer les exemples du tutoriel "Sequences"
- Que pensez vous des résultats d'extraction de date ? Pouvez vous faire mieux, en utilisant des choses vues dans le tutoriel précédent ?
- Adaptez vos patrons de requête simple en requête de séquence toujours pour extraire des membres de groupes de musique. Sauvez le résultat et évaluez la qualité.
- Enrichissez le résultat en utilisant un patron avec la date où quelqu'un rejoint un groupe. Vous pouvez combiner plusieurs patrons en créant un "Query set" (sauvable/rechargeable).

Exemples de pattern :

```
<E>band:e=ORG member <E>member:e=PERSON  
<E>member:e=PERSON ... joined <E>band:e=ORG ... <E>date:e=DATE
```

Démonstration/Exercice 3/3

Prise en main avec le tutoriel: requêtes syntaxiques

Les requêtes syntaxiques fonctionnent sur le principe d'un patron exemple, qui va servir à une requête plus générique.

- Lisez et appliquer les exemples du tutoriel "Structured queries"
- Adaptez maintenant le tutoriel pour construire une base de données de groupe musicaux et leurs membres, la plus complète possible en utilisant les requêtes structurées.
- vous pouvez mettre au point des patrons intuitivement, suivre la méthodologie consistant à choisir des "graines" (seed), pour en déduire des patrons. Il est plus simple de faire une requête simple pour ça (en mettant par exemple un nom de groupe et un nom de membre de ce groupe).
- de même après une première extraction, on peut chercher des manques évidents, et trouver d'autres graines avec les informations manquantes.

Correction

après une requête avec un exemple de base: Robert Plant Led Zeppelin
on voit des formes du genre "former singer X of Y" d'où la requête structurée :

```
<E>band: [e]LedZeppelin $[w=frontman|singer|guitarist|bassist|drummer]singer  
<E>member: [e]RobertPlant
```

ici il faut vraiment restreindre aux entités nommées pour le groupe sinon on a pop singer, etc la tentation est grande d'ajouter "member" mais trop vague

Ensuite on regarde ce qu'il manque: U2 y'a que Bono ? on refait avec U2 et autre membre genre Adam Clayton

on voit "U2 bandmate Adam Clayton" et donc on peut essayer

```
<E>band:U2 $bandmate <E>member:AdamClayton
```

et aussi ajouter un possessif <E>band: [e]LedZeppelin \$'s
\$[w=frontman|singer|guitarist|bassist|drummer]singer <E>member: [e]RobertPlant