Text Mining

Marcelo Mendoza
http://www.inf.utfsm.cl/~mmendoza
mmendoza@inf.utfsm.cl
A 131, Campus San Joaquín - UTFSM

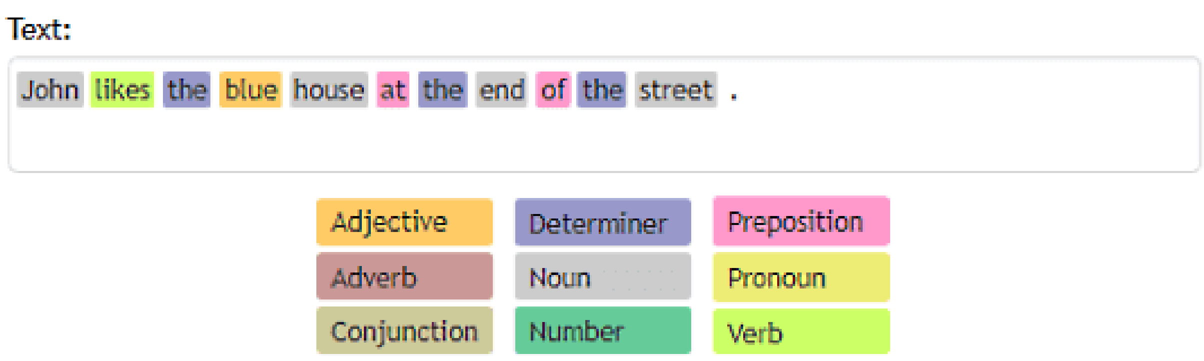
POS tagging

- Etiquetar cada término de acuerdo a la función que este cumple en el texto.
- Puede ayudarnos en tareas como detección de estilo, parsing, detección de colocaciones.
- Tarea importante en NLP.

Conceptos de NLP

POS tagging

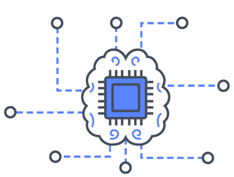
- Etiquetar cada término de acuerdo a la función que este cumple en el texto.
- Puede ayudarnos en tareas como detección de estilo, parsing, detección de colocaciones.
- Tarea importante en NLP.



Conceptos de NLP

POS tagging

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red, young, awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very, slowly, home, yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm, cat, mango, beauty*
	VERB	words for actions and processes	*draw, provide, go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina, IBM, Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh, um, yes, hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that, which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg



Conceptos de NLP

POS tagging en NLTK

```
>>> text = word_tokenize("And now for something completely different")
>>> nltk.pos_tag(text)
[('And', 'CC'), ('now', 'RB'), ('for', 'IN'), ('something', 'NN'),
('completely', 'RB'), ('different', 'JJ')]
```

```
>>> text = word_tokenize("They refuse to permit us to obtain the refuse permit")
>>> nltk.pos_tag(text)
[('They', 'PRP'), ('refuse', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'),
('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]
```

POS tagging en Spacy (Español)

```
!python -m spacy download es_core_news_sm
!python -m spacy download es_core_news_md
```

```
import spacy
import es_core_news_sm
import es_core_news_md

nlp = es_core_news_sm.load()
doc = nlp('''Un desastroso espirítu posee tu tierra:
            donde la tribu unida blandió sus mazas,
            hoy se enciende entre hermanos perpetua guerra,
            se hieren y destrozan las mismas razas.''')

for token in doc: print(token.text, "|", token.lemma_, '|', token.pos_)
```

POS tagging en Spacy (Español)

```
!python -m spacy download es_core_news_sm
!python -m spacy download es_core_news_md
```

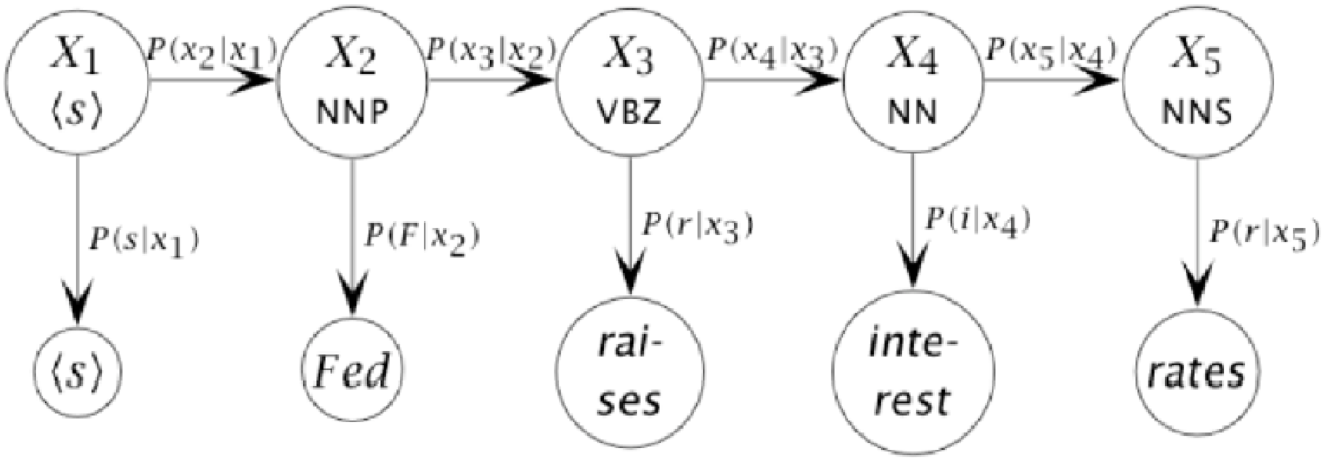
```
import spacy
import es_core_news_sm
import es_core_news_md

nlp = es_core_news_sm.load()
doc = nlp('''Un desastroso espirítu posee tu tierra:
            donde la tribu unida blandió sus mazas,
            hoy se enciende entre hermanos perpetua guerra,
            se hieren y destrozan las mismas razas.''')

for token in doc: print(token.text, "|", token.lemma_, '|', token.pos_)
```

```
Un | Un | DET
desastroso | desastroso | NOUN
espirítu | espirítu | PROPN
posee | poseer | VERB
tu | tu | DET
tierra | tierra | NOUN
: | : | PUNCT
```

POS tagging ¿Cómo funciona?

- Se dispone de un corpus etiquetado.
- La secuencia de tags es interpretada como una cadena de Markov: $P(x_{t+1} \mid x_t, \ldots, x_1) = P(x_{t+1} \mid x_t)$, $x_1, \ldots, x_{t+1}$ representan tags
- Usamos un modelo generativo para términos, con tags como estados ocultos: $P(t \mid x_1, \ldots, x_{t+1}) = P(t \mid x_{t+1})$

POS tagging ¿Cómo funciona?



- En general muestran buena precisión (sobre 90 %).

Conceptos de NLP

HMM POS tagging:

¿Cómo defino la cadena de transiciones?

$Q = q_1 q_2 \dots q_N$	a set of N **states**
$A = a_{11} \dots a_{ij} \dots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \dots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, \dots, v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state q_i
$\pi = \pi_1, \pi_2, \dots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Conceptos de NLP

HMM POS tagging:

¿Cómo defino la cadena de transiciones?

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, \ldots, v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state q_i
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Objetivo del modelo:

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n)$$

Conceptos de NLP

HMM POS tagging: $$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n)$$

Aplicamos la regla de Bayes: $$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} \frac{P(w_1 \ldots w_n | t_1 \ldots t_n) P(t_1 \ldots t_n)}{P(w_1 \ldots w_n)}$$

y simplificamos el denominador: $$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(w_1 \ldots w_n | t_1 \ldots t_n) P(t_1 \ldots t_n)$$

Se asume que la probabilidad de la palabra depende sólo de su tag:

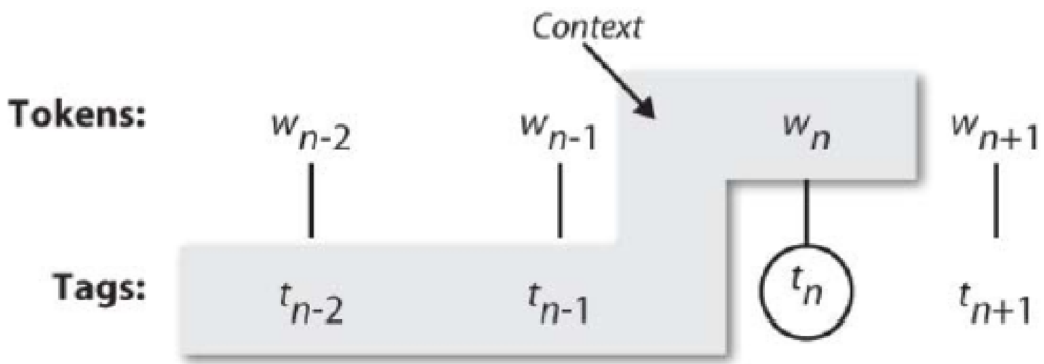
$$P(w_1 \ldots w_n | t_1 \ldots t_n) \approx \prod_{i=1}^{n} P(w_i | t_i)$$

y se asume que el tag sólo depende del tag previo:

$$P(t_1 \ldots t_n) \approx \prod_{i=1}^{n} P(t_i | t_{i-1})$$

Finalmente: $$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n) \approx \underset{t_1 \ldots t_n}{\operatorname{argmax}} \prod_{i=1}^{n} \overbrace{P(w_i | t_i)}^{\text{emission}} \overbrace{P(t_i | t_{i-1})}^{\text{transition}}$$

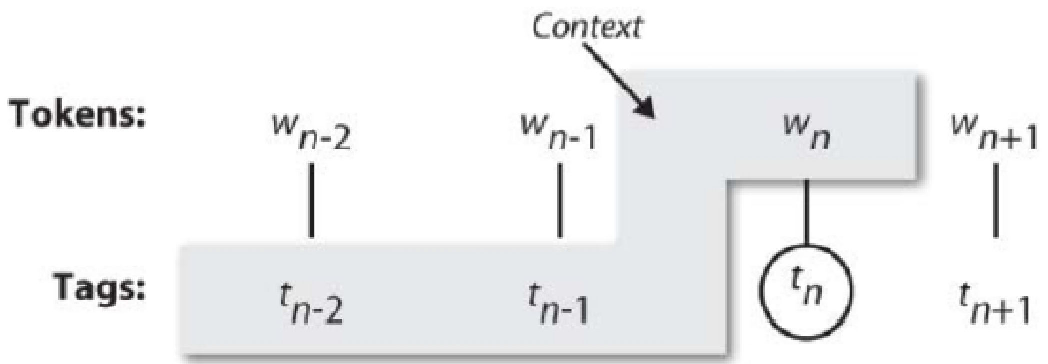
POS tagging ¿Cómo funciona?



- Considera los tags de las dos palabras precedentes.
- En general muestra mejor precisión que HMM.

Conceptos de NLP

POS tagging ¿Cómo funciona?



- Considera los tags de las dos palabras precedentes.
- En general muestra mejor precisión que HMM.

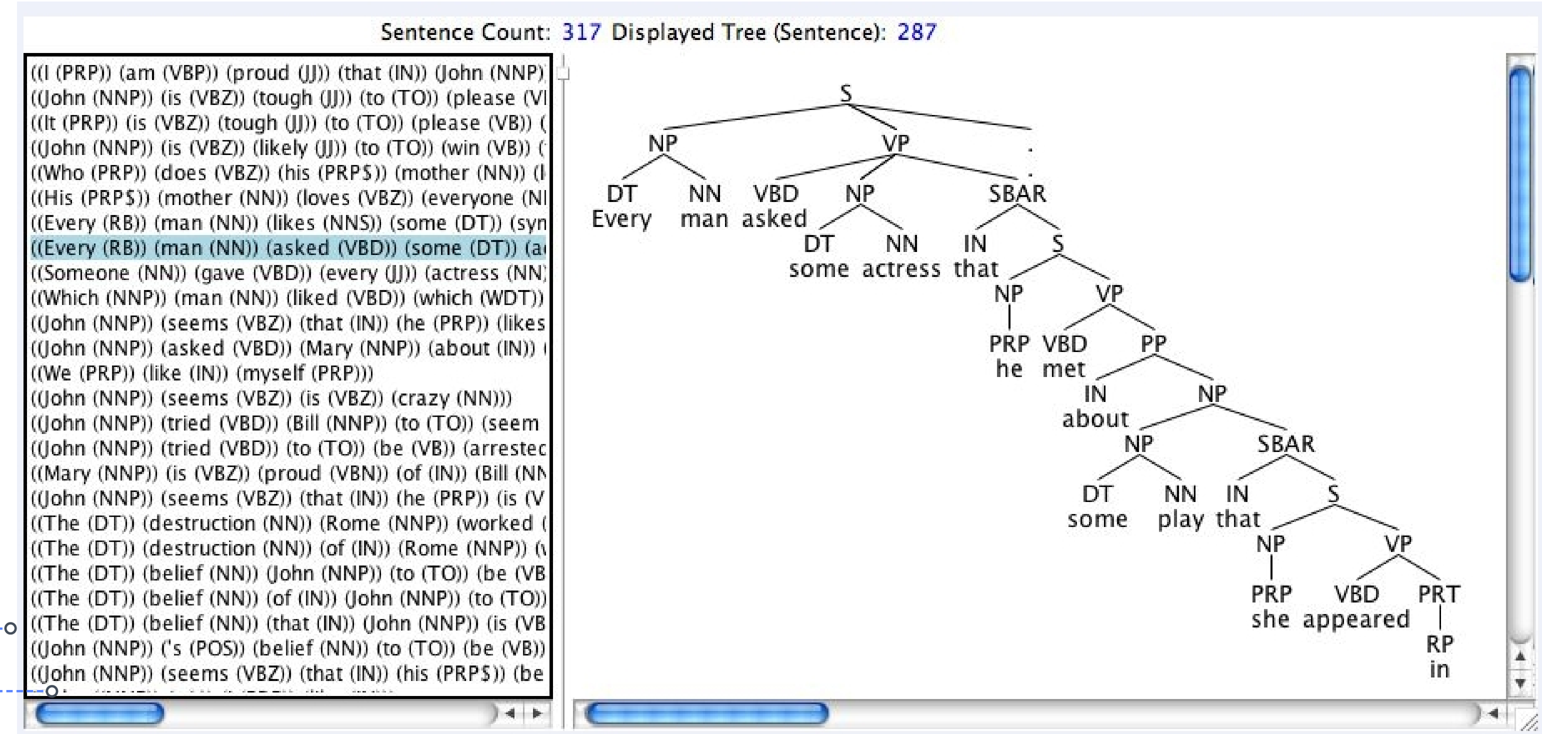
Hinrich Schütze, Yoram Singer: Part-of-Speech Tagging using a Variable Memory Markov Model. ACL1994: 181-187

Conceptos de NLP

POS tagging ¿Cuáles datos usan?

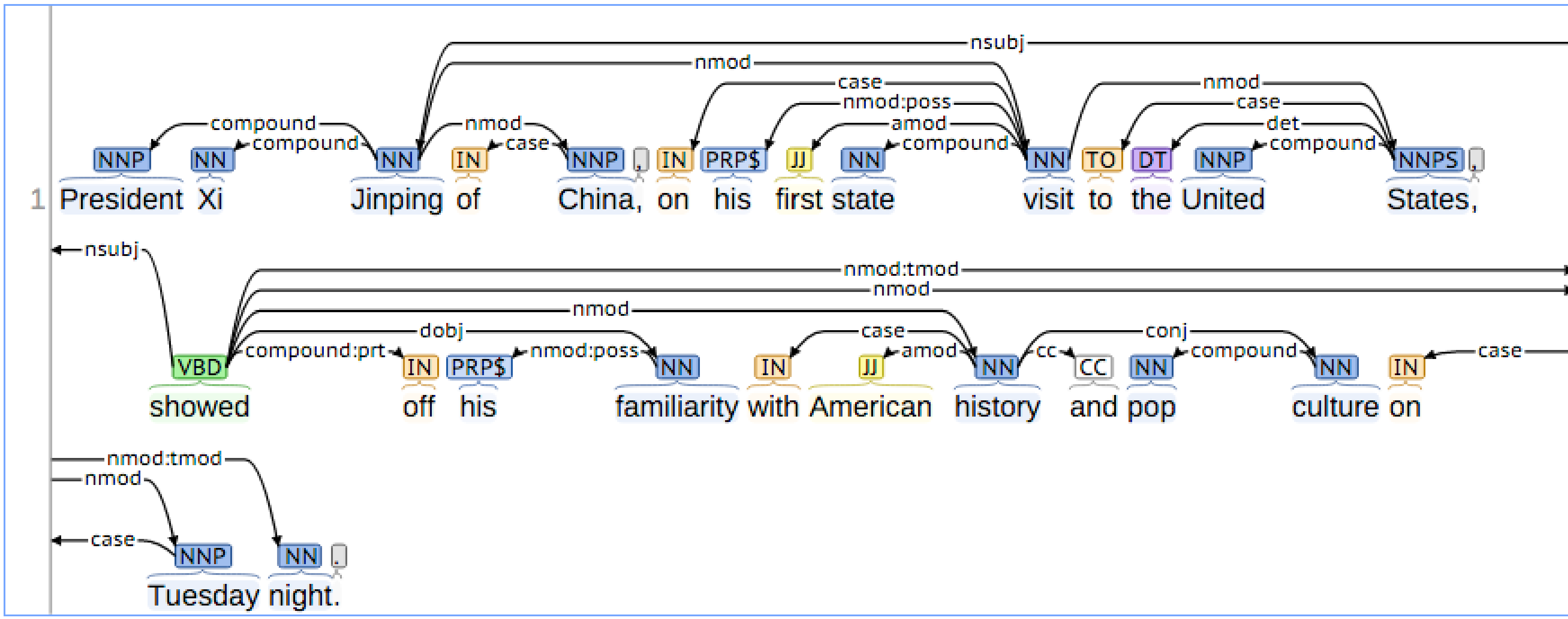
Treebanks: Penn treebank (más famoso), UAM Spanish Treebank, ...

Treebank viewer:



Conceptos de NLP

POS tagging y dependencias



Conceptos de NLP

POS tagging y dependencias en Spacy

Editable Code spaCy v3.0 · Python 3 · via Binder

```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")

for token in doc:
    print(token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
            token.shape_, token.is_alpha, token.is_stop)
```

RUN

Conceptos de NLP

POS tagging y dependencias en Spacy

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
Apple	apple	PROPN	NNP	nsubj	Xxxxx	True	False
is	be	AUX	VBZ	aux	xx	True	True
looking	look	VERB	VBG	ROOT	xxxx	True	False
at	at	ADP	IN	prep	xx	True	True
buying	buy	VERB	VBG	pcomp	xxxx	True	False
U.K.	u.k.	PROPN	NNP	compound	X.X.	False	False
startup	startup	NOUN	NN	dobj	xxxx	True	False
for	for	ADP	IN	prep	xxx	True	True
$	$	SYM	$	quantmod	$	False	False
1	1	NUM	CD	compound	d	False	False
billion	billion	NUM	CD	pobj	xxxx	True	False

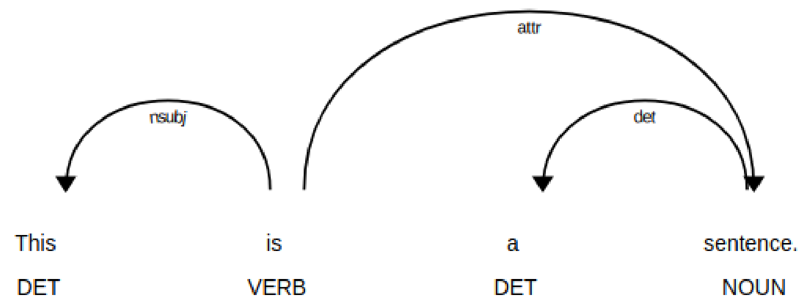
Dependencia sintáctica

Conceptos de NLP

POS tagging y dependencias en Spacy

DEPENDENCY EXAMPLE

```
import spacy
from spacy import displacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("This is a sentence.")
displacy.serve(doc, style="dep")
```



Conceptos de NLP

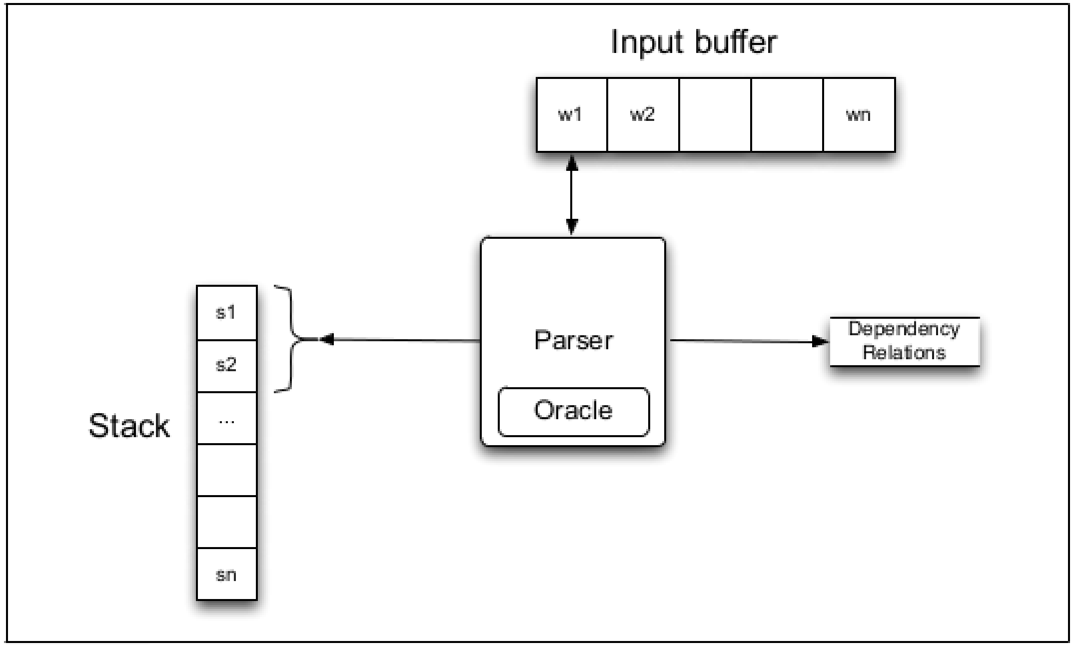
Dependencias ¿Cómo funciona?

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

Conceptos de NLP

Dependencias ¿Cómo funciona?

Transition-based parser:



Dependencias ¿Cómo funciona?

Transition-based parser: Deducimos training instances desde un treebank. Podemos determinísticamente registrar las operaciones correctas del parser, creando un training dataset.

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	

Dependencias ¿Cómo funciona?

Training instances (intermedia):

Actúa sobre el primer token del buffer

Stack	Word buffer	Relations
[root, canceled, flights]	[to Houston]	(canceled $\rightarrow$ United)
		(flights $\rightarrow$ morning)
		(flights $\rightarrow$ the)

La transición correcta del parser es *shift*. Luego, creamos training instances de este tipo en el dataset:

$$\langle s_1.w = flights, op = shift \rangle$$
$$\langle s_2.w = canceled, op = shift \rangle$$

Tokens del stack

$$\langle s_1.t = NNS, op = shift \rangle$$
$$\langle s_2.t = VBD, op = shift \rangle$$

POS-tags del stack

$$\langle b_1.w = to, op = shift \rangle$$
$$\langle b_1.t = TO, op = shift \rangle$$

Token y tag del buffer

$$\langle s_1.wt = flightsNNS, op = shift \rangle$$

Template compuesto

Conceptos de NLP

Dependencias ¿Cómo funciona?

Source	Feature templates		
One word	$s_1.w$	$s_1.t$	$s_1.wt$
	$s_2.w$	$s_2.t$	$s_2.wt$
	$b_1.w$	$b_1.w$	$b_0.wt$
Two word	$s_1.w \circ s_2.w$	$s_1.t \circ s_2.t$	$s_1.t \circ b_1.w$
	$s_1.t \circ s_2.wt$	$s_1.w \circ s_2.w \circ s_2.t$	$s_1.w \circ s_1.t \circ s_2.t$
	$s_1.w \circ s_1.t \circ s_2.t$	$s_1.w \circ s_1.t$	

El clasificador predice dep tag a nivel de palabra usando las características indicadas.

Típicamente se usan SVM, LR multinomial o ANN con softmax para esta tarea.

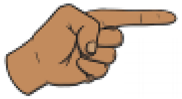
Conceptos de NLP

Dependencias ¿Cómo funciona?

Source	Feature templates		
One word	$s_1.w$	$s_1.t$	$s_1.wt$
	$s_2.w$	$s_2.t$	$s_2.wt$
	$b_1.w$	$b_1.w$	$b_0.wt$
Two word	$s_1.w \circ s_2.w$	$s_1.t \circ s_2.t$	$s_1.t \circ b_1.w$
	$s_1.t \circ s_2.wt$	$s_1.w \circ s_2.w \circ s_2.t$	$s_1.w \circ s_1.t \circ s_2.t$
	$s_1.w \circ s_1.t \circ s_2.t$	$s_1.w \circ s_1.t$	

El clasificador predice dep tag a nivel de palabra usando las características indicadas.

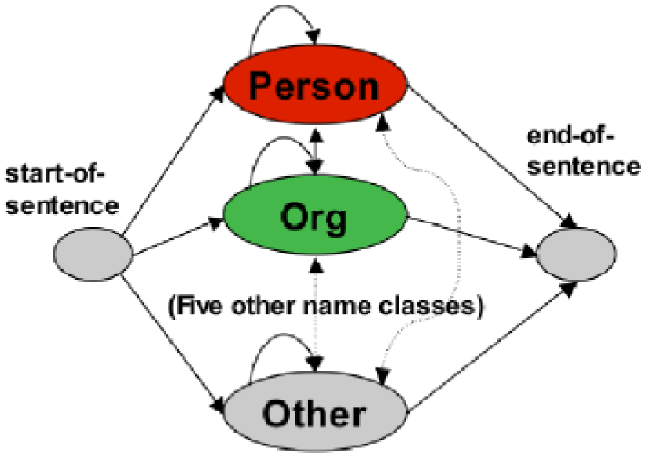
Típicamente se usan SVM, LR multinomial o ANN con softmax para esta tarea.



Danqi Chen, Christopher D. Manning: A Fast and Accurate Dependency Parser using Neural Networks. EMNLP2014: 740-750

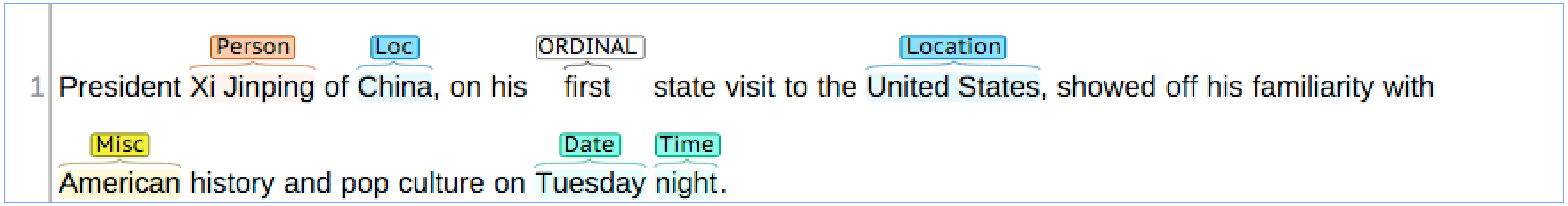
Named Entity Recognition

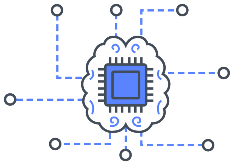
- Tarea: Identificar entidades en texto (personas, organizaciones, etc.)
- Separa el text en chunks, y para cada cual asocia una NE. Opera sobre texto tagged.
- NER types: organization, person, location, date, time, money, percent, facility (human made artifacts), gpe (geo-political ents).
- POS tagging puede ayudar, agregando *entity* como un estado mas.



Conceptos de NLP

Named Entity Recognition

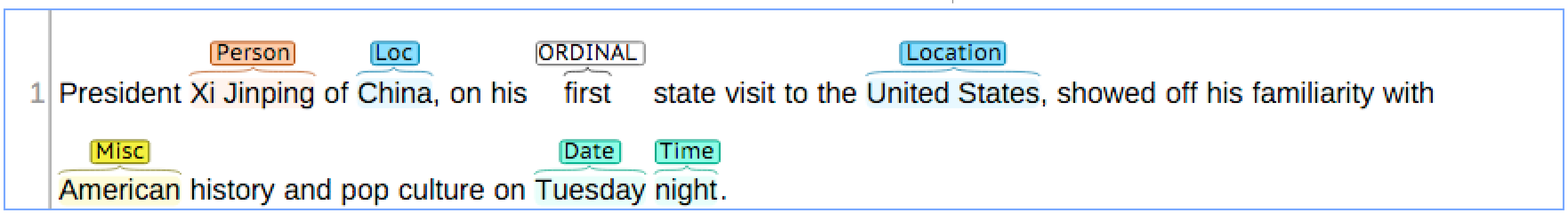




Conceptos de NLP

Named Entity Recognition

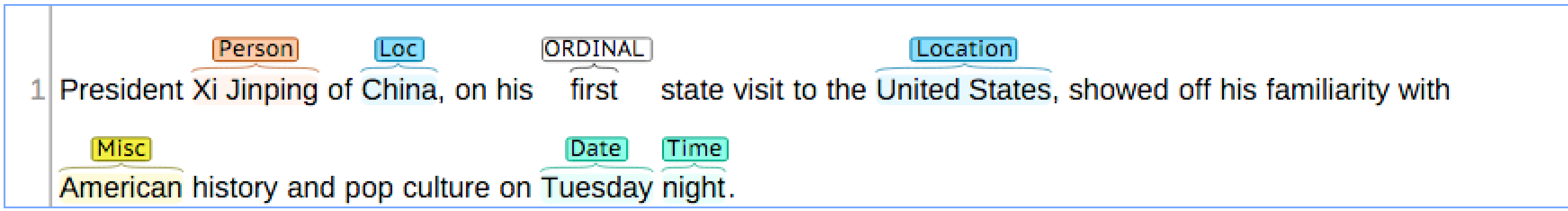
bi-grama



Conceptos de NLP

Named Entity Recognition

bi-grama



Named Entity Recognition puede ser muy desafiante:

Named Entity Recognition en Spacy

```
nlp_md = es_core_news_md.load()
article_text = '''La ONG Fundación del Río explicó este viernes ( ...
doc = nlp_md(article_text)
SVG(data = displacy.render(doc, style="ent"))
```

Conceptos de NLP

Named Entity Recognition en Spacy

```
nlp_md = es_core_news_md.load()
article_text = '''La ONG Fundación del Río explicó este viernes q
doc = nlp_md(article_text)
SVG(data = displacy.render(doc, style="ent"))
```
...

La ONG Fundación del Río ORG explicó este viernes que la decisión de la Organización de la ONU ORG para la Educación ORG , la Ciencia LOC y la Cultura LOC (Unesco ORG) de declarar como geoparque el río Coco LOC , ubicado en el norte de Nicaragua LOC , obliga a las autoridades nicaragüenses a proteger su ecosistema, ya que se encuentra en el área más deforestada de la cuenca. La Unesco LOC está reconociendo la importancia del río Coco LOC , pero también está haciendo un llamado al Gobierno LOC a que actúe en la protección y la conservación de esos ecosistemas, dijo a Efe PER el presidente de la Fundación del Río ORG , Amaru Ruiz PER .

Collocations: n-gramas frecuentes

- El significado conjunto es más que la suma de las partes (compositionality)
  1. Armas de destrucción masiva
  2. Strong tea
  3. Libre de sodio
  4. Intel inside
  5. Fast food
  6. Nuclear war
- Detectar colocaciones mejora la representación del contenido.
- Cada colocación puede ser procesada como un término.
- Se pueden detectar analizando co ocurrencias, etiquetando el par como una colocación si su co ocurrencia es mucho mayor que la esperada (azar, equiprobable).

Conceptos de NLP

Collocations: ¿Cómo se detectan en NLTK?

```
>>> import nltk
>>> from nltk.collocations import *
>>> bigram_measures = nltk.collocations.BigramAssocMeasures()
>>> trigram_measures = nltk.collocations.TrigramAssocMeasures()
>>> finder = BigramCollocationFinder.from_words(
...     nltk.corpus.genesis.words('english-web.txt'))
>>> finder.nbest(bigram_measures.pmi, 10)  # doctest: +NORMALIZE_WHITESPACE
[(u'Allon', u'Bacuth'), (u'Ashteroth', u'Karnaim'), (u'Ben', u'Ammi'),
 (u'En', u'Mishpat'), (u'Jegar', u'Sahadutha'), (u'Salt', u'Sea'),
 (u'Whoever', u'sheds'), (u'appoint', u'overseers'), (u'aromatic', u'resin'),
 (u'cutting', u'instrument')]
```

Conceptos de NLP

Collocations: ¿Cómo se detectan en NLTK?

```
>>> import nltk
>>> from nltk.collocations import *
>>> bigram_measures = nltk.collocations.BigramAssocMeasures()
>>> trigram_measures = nltk.collocations.TrigramAssocMeasures()
>>> finder = BigramCollocationFinder.from_words(
...     nltk.corpus.genesis.words('english-web.txt'))
>>> finder.nbest(bigram_measures.pmi, 10)  # doctest: +NORMALIZE_WHITESPACE
[(u'Allon', u'Bacuth'), (u'Ashteroth', u'Karnaim'), (u'Ben', u'Ammi'),
 (u'En', u'Mishpat'), (u'Jegar', u'Sahadutha'), (u'Salt', u'Sea'),
 (u'Whoever', u'sheds'), (u'appoint', u'overseers'), (u'aromatic', u'resin'),
 (u'cutting', u'instrument')]
```

Podemos usar un umbral de frecuencia absoluta para encontrar *collocations* frecuentes y luego rankear usando PMI:

$$\mathrm{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)}$$

```
>>> finder.apply_freq_filter(3)
>>> finder.nbest(bigram_measures.pmi, 10)  # doctest: +NORMALIZE_WHITESPACE
[(u'Beer', u'Lahai'), (u'Lahai', u'Roi'), (u'gray', u'hairs'),
 (u'Most', u'High'), (u'ewe', u'lambs'), (u'many', u'colors'),
 (u'burnt', u'offering'), (u'Paddan', u'Aram'), (u'east', u'wind'),
 (u'living', u'creature')]
```