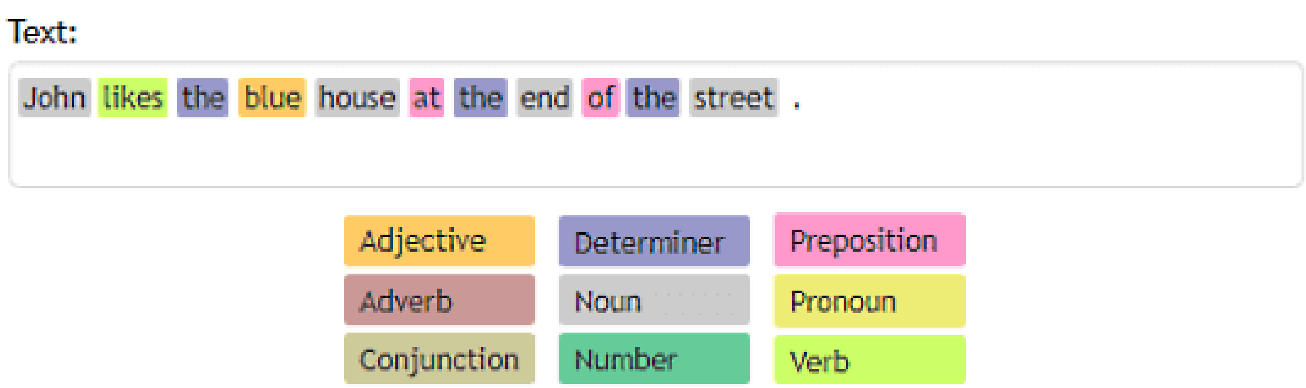
IIC 3800 Tópicos en CC
NLP

https://github.com/marcelomendoza/IIC3800

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')]
```

Conceptos de NLP

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_)
```

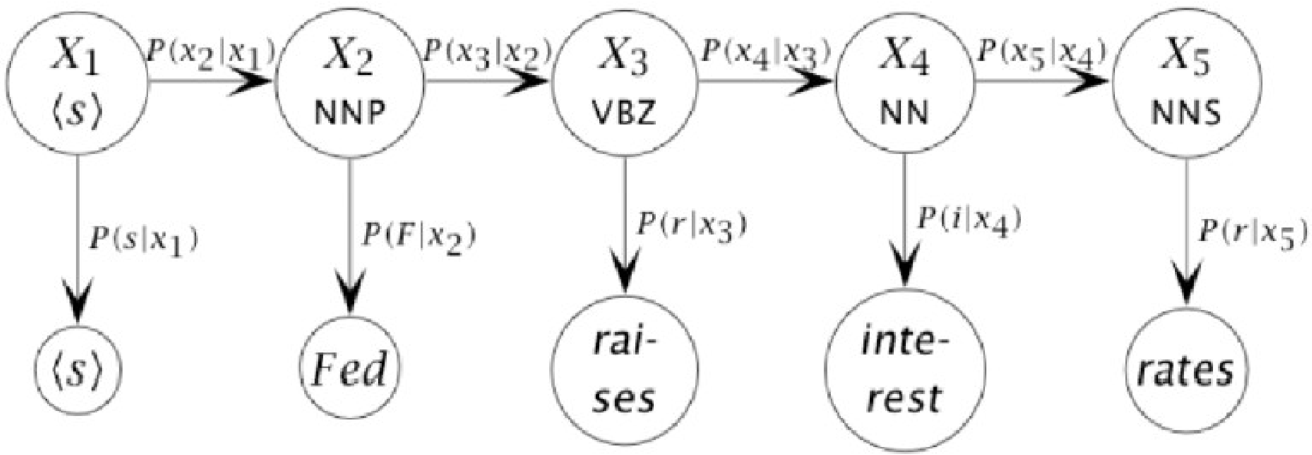
```
Un | Un | DET
desastroso | desastroso | NOUN
espirítu | espirítu | PROPN
posee | poseer | VERB
tu | tu | DET
tierra | tierra | NOUN
: | : | PUNCT
```

POS tagging con HMM (modelo clásico)

- 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: $P(t \mid x_1, \ldots, x_{t+1}) = P(t \mid x_{t+1})$

Conceptos de NLP

POS tagging con HMM (modelo clásico)



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

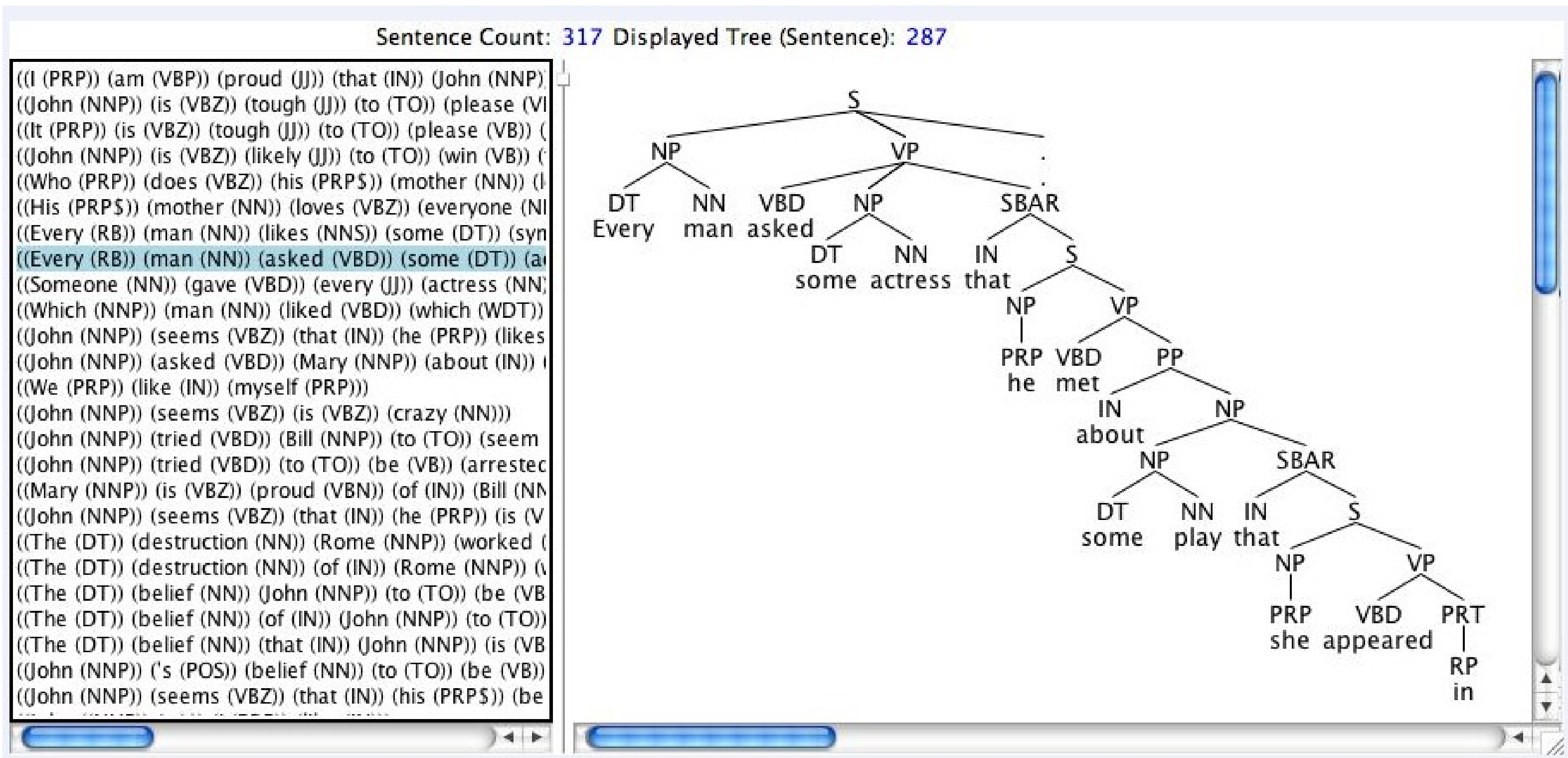
El algoritmo de entrenamiento se llama Viterbi.

Conceptos de NLP

POS tagging ¿Cuáles datos usan?

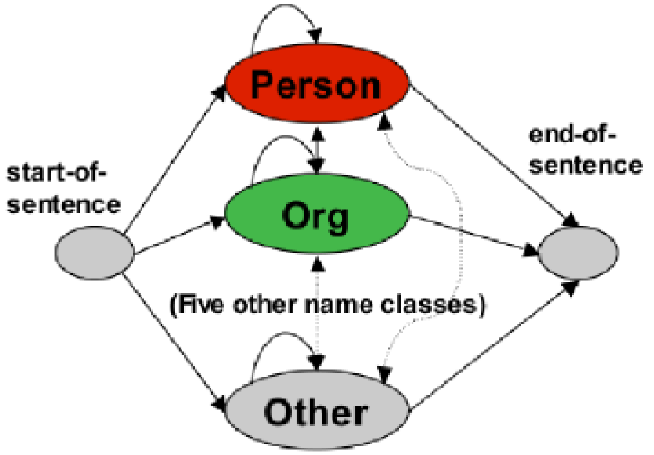
Treebanks: Penn treebank (más famoso), UAM Spanish Treebank, ...

Treebank viewer:



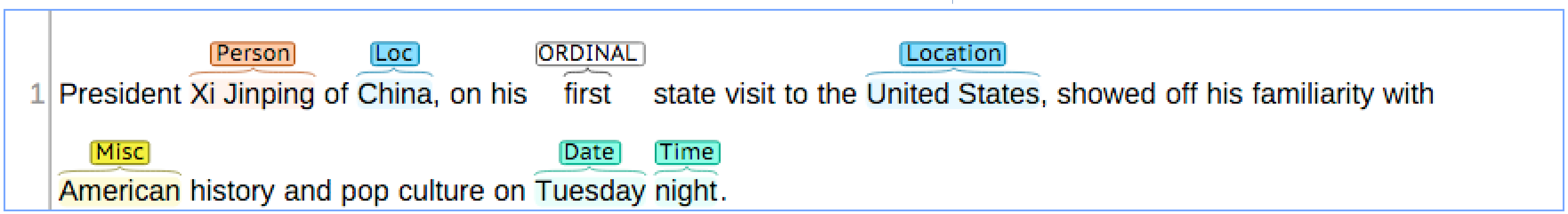
Named Entity Recognition

- 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.



Named Entity Recognition

bi-grama

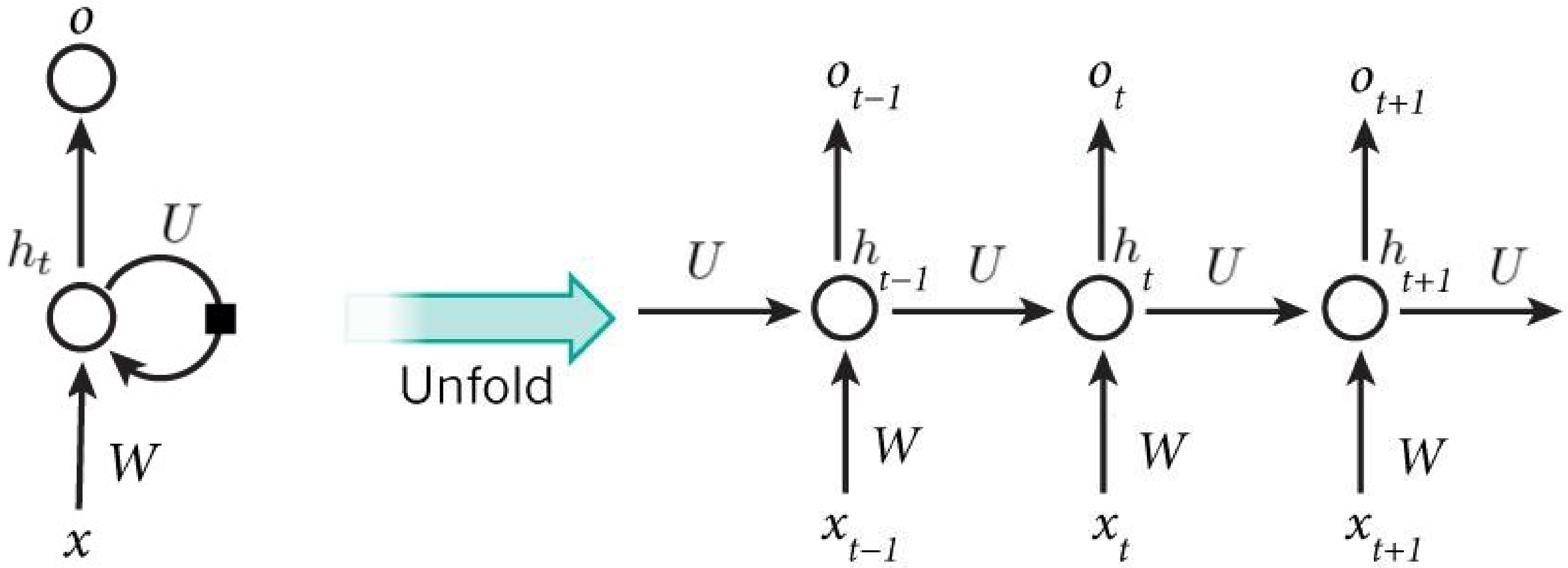


Named Entity Recognition puede ser muy desafiante:

Sequence labeling

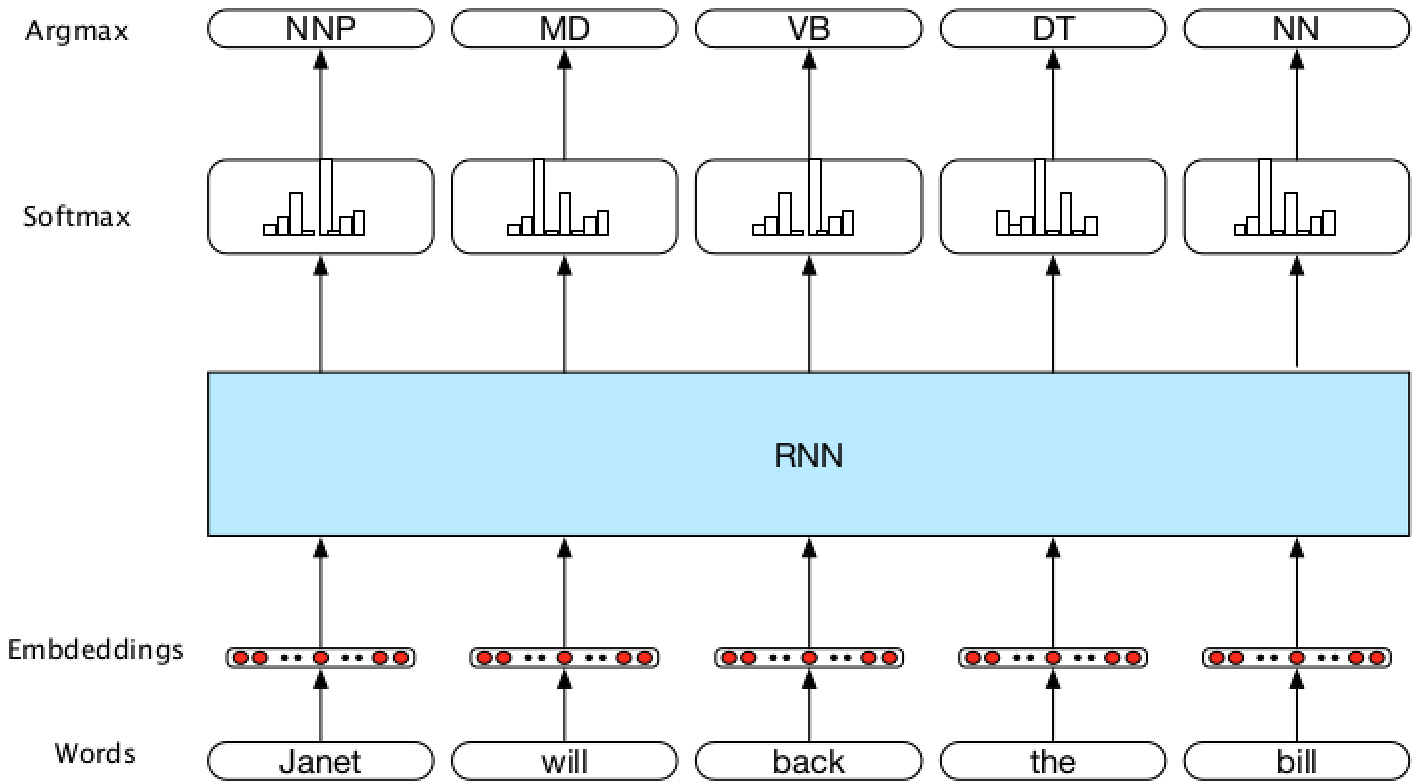
Presentamos los datos como secuencia: $X = (x_1, x_2, \ldots, x_T)$

Recurrente convencional: $h_t = g(W \cdot x_t + U \cdot h_{t-1})$



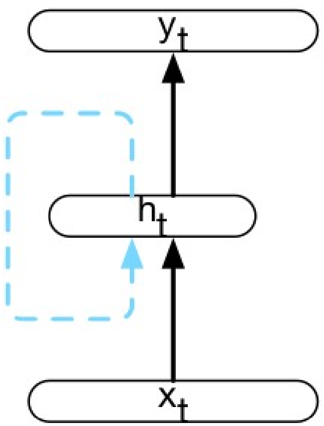
Sequence labeling

Red recurrente (**sequence labeling**):

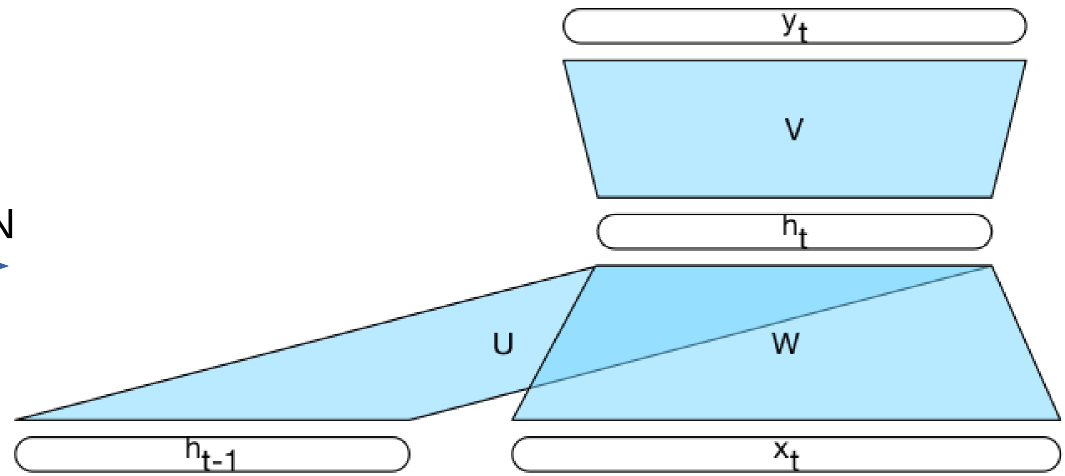


Sequence labeling

Red recurrente:



como FFNN

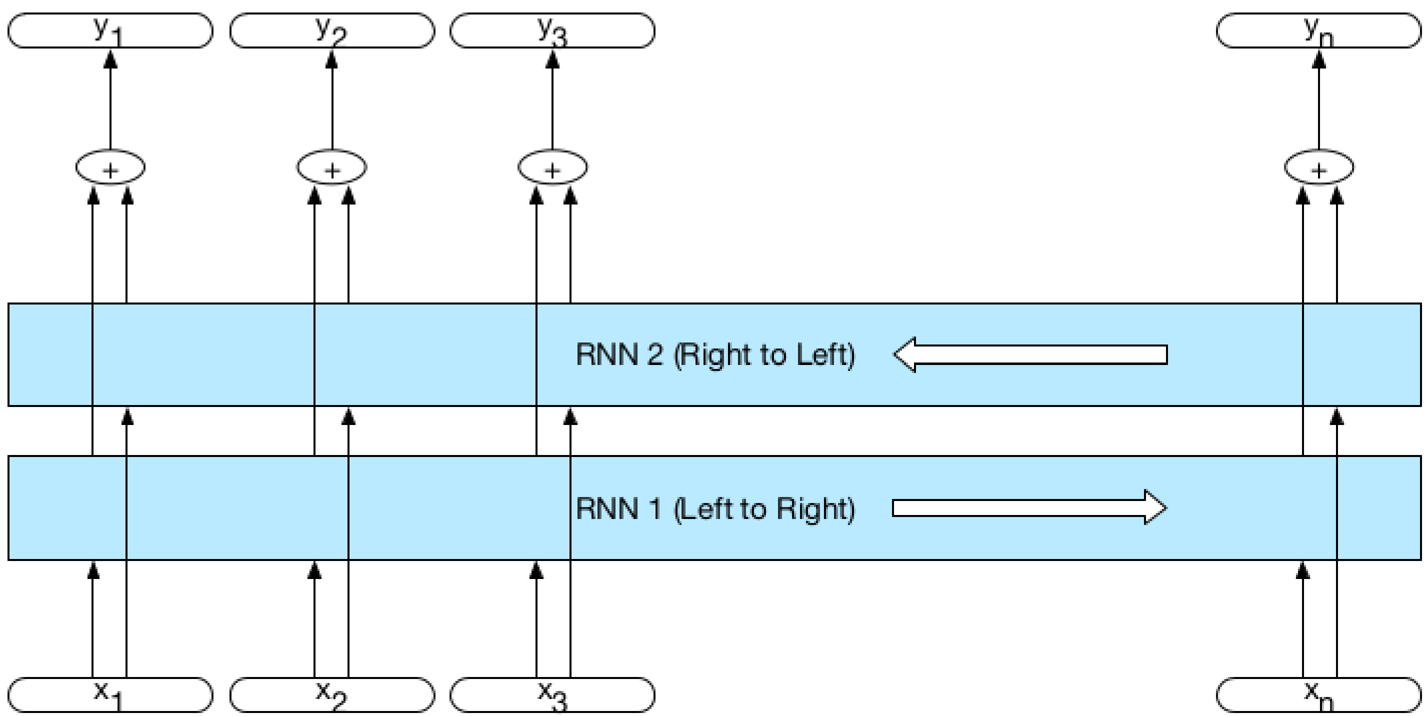


$$
\begin{aligned}
h_t &= g(Uh_{t-1} + Wx_t) \\
y_t &= f(Vh_t)
\end{aligned}
$$

$$y_t = \text{softmax}(Vh_t)$$

Sequence labeling

Bidirectional RNN (biRNN):



Sequence labeling

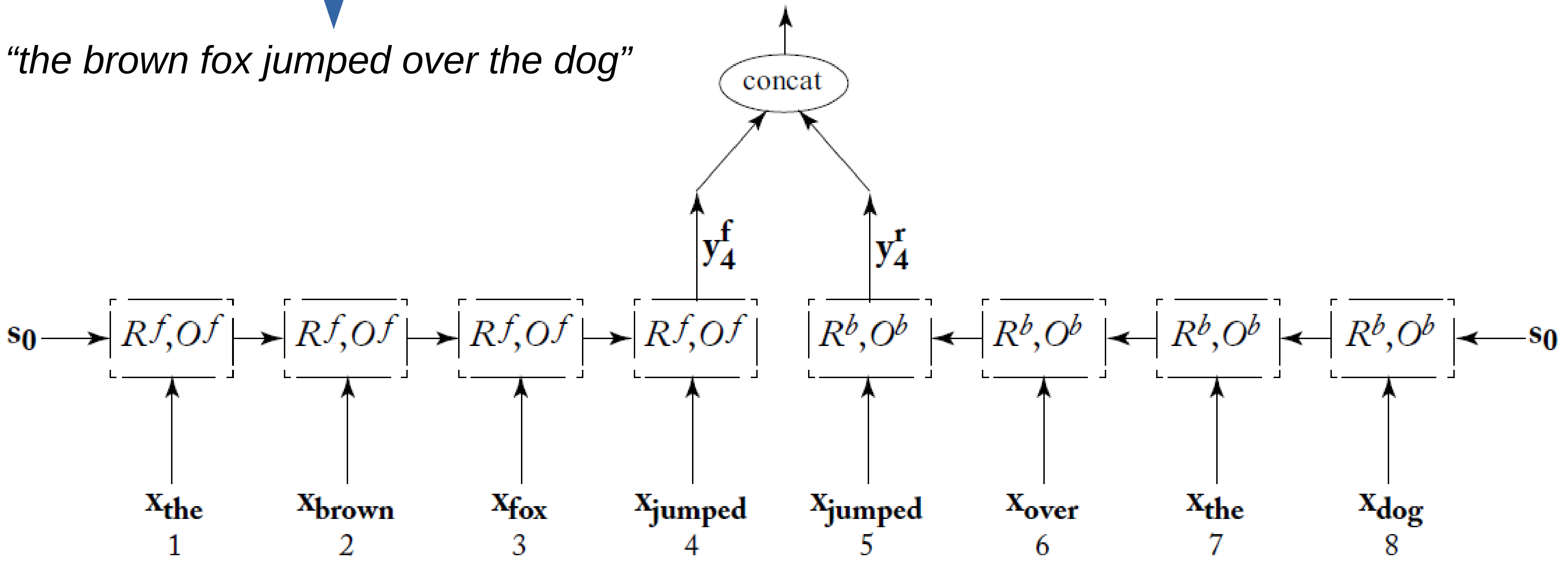
Bidirectional RNN (biRNN):

$$\mathrm{biRNN}(\boldsymbol{x}_{1:n}, i) = \boldsymbol{y}_i = [\mathrm{RNN}^f(\boldsymbol{x}_{1:i}); \mathrm{RNN}^b(\boldsymbol{x}_{n:i})].$$

forward

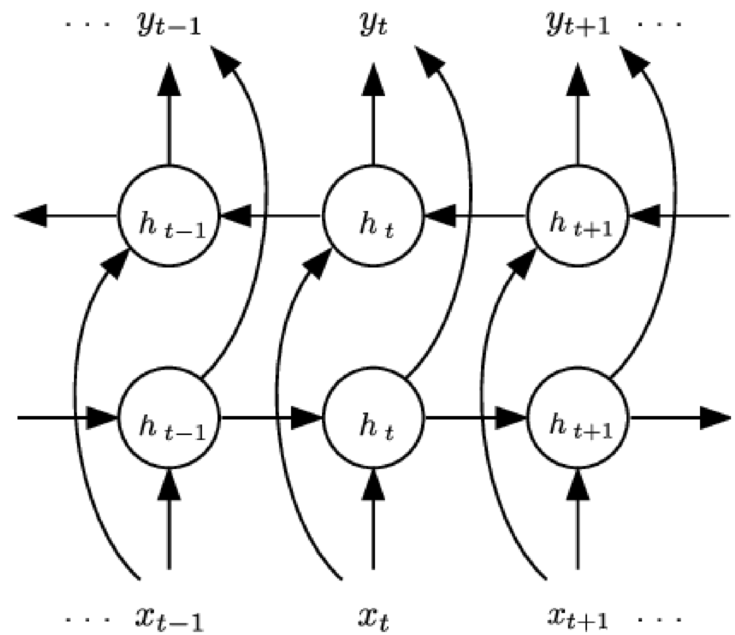
backward

"the brown fox jumped over the dog"



Sequence labeling

Bidirectional RNN (biRNN):



Dos capas:

$$h_t^{(f)} = g(W^{(f)} \cdot x_t + U^{(f)} \cdot h_{t-1}^{(f)})$$
$$h_t^{(b)} = g(W^{(b)} \cdot x_t + U^{(b)} \cdot h_{t+1}^{(b)})$$
$$y_t = g(V^{(f)} \cdot h_t^{(f)} + V^{(b)} \cdot h_t^{(b)})$$

Sequence labeling

Bidirectional RNN para POS tagging con modelo preentrenado de subpalabras

Char embeddings: $x_i = \phi(s, i) = [E_{[w_i]}; \mathrm{RNN}^f(c_{1:\ell}); \mathrm{RNN}^b(c_{\ell:1})].$

FastText

char n-grams

POS-tagging: $p(t_i = j | w_1, \ldots, w_n) = \mathrm{softmax}(\mathrm{MLP}(\mathrm{biRNN}(x_{1:n}, i)))_{[j]}$

Sequence labeling

Bidirectional RNN para POS tagging con modelo preentrenado de subpalabras

char 1-grams