Trường Đại học Công nghệ Thông tin - Đại học Quốc gia TP.HCM

DS310.L21: Xử lý ngôn ngữ tự nhiên cho Khoa học dữ liệu

Phân tích cú pháp thành tố

Tháng 4 năm 2021

(Một số slides tham khảo từ Chris Manning, Mike Collins, và Shay Cohen.)

Nhắc lại: Một số mô hình ngôn ngữ

đi phân loại văn bản

- Mô hình Bag-of-words: bỏ qua hoàn toàn thứ tự của "từ".
- Mô hình N-gram: bắt ngữ cảnh ngắn bên trái để dự đoán "từ tiếp theo".

☆ Các mô hình ngôn ngữ vừa nêu hữu dụng trên nhiều bài toán (phân loại văn bản, gán nhãn từ loại, v.v...).
→ Như vậy, một mô hình ngôn ngữ tốt cần "**bắt**" (**capture**) được những được thông tin/đối tượng nào?

Bản chất của các mô hình ngôn ngữ là gì?

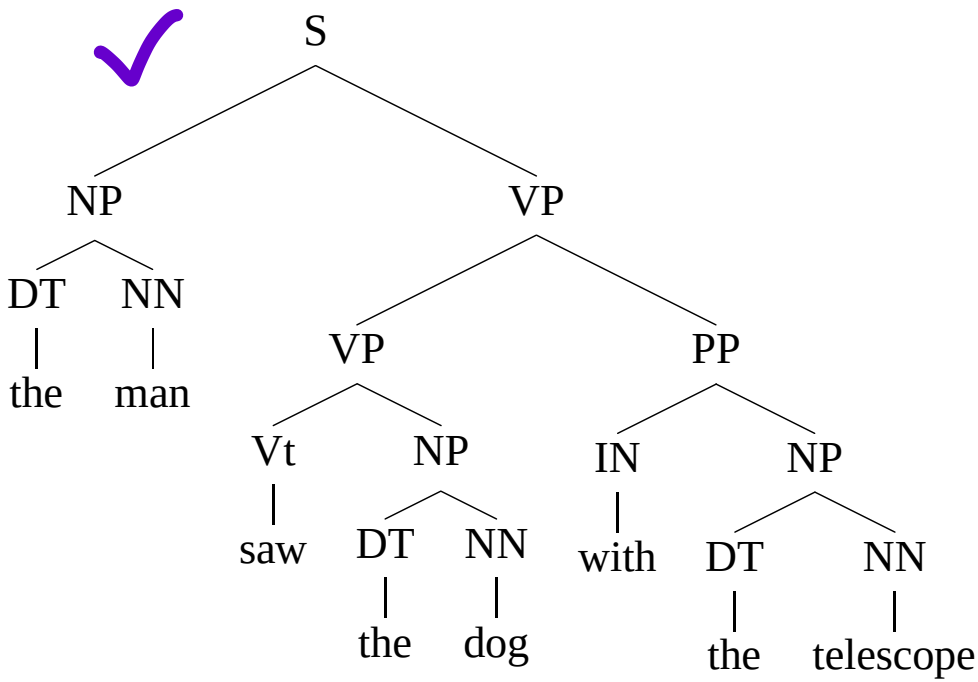
- Mô hình hóa hành vi của ngôn ngữ.
- Các mô hình ngôn ngữ có thể "**bắt**" (**capture**) được những gì?
  - Bag-of-words: Sự xuất hiện của các từ. (Occurrence of words.) tf-idf
  - N-gram: Sự kết hợp của các từ liên tiếp. (Combinations of adjacent word.)
  - **Cấu trúc thành tố: Khả năng thay thế cụm từ. (Phrasal substitutability.)**
  - **Cấu trúc phụ thuộc: Phụ thuộc xa. (Long-range dependencies.)**

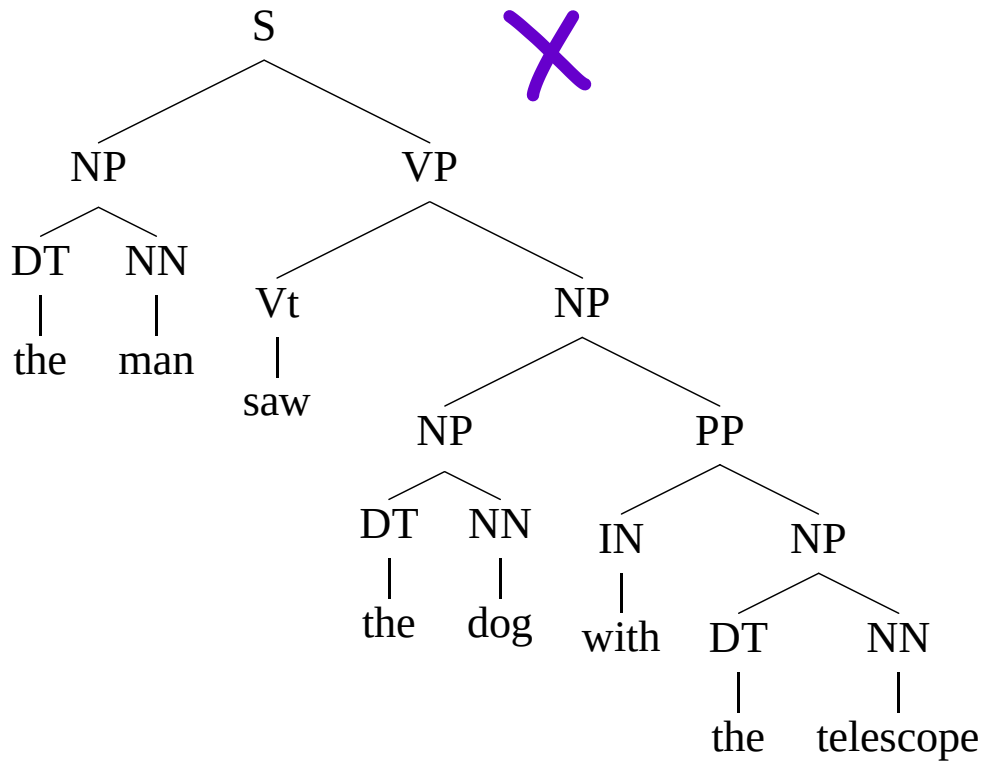
Ví dụ 1: Khả năng thay thế "cụm từ"

- Nhãn từ loại (POS categories) chỉ ra những "từ" có thể thay thế. Ví dụ, việc thay thế tính từ:
  - I saw a **red** cat.
  - I saw a **former** cat.
  - I saw a **billowy** cat.
- Nhãn cụm từ (Phrasal categories) chỉ ra những "cụm từ" cụm từ có thể thay thế. Ví dụ, việc thay thế cụm danh từ:
  - **Dogs** sleep soundly.
  - **My next-door neighbours** sleep soundly.
  - **Green ideas** sleep soundly.

Ví dụ 2: Cấu trúc ngữ pháp thành tố [1]

the man saw the dog with the telescope





[1] Hình ảnh được tham khảo từ bài giảng của Michael Collins.

Ví dụ 3: Phụ thuộc giữa các "từ" cách xa nhau

- Ví dụ 3.a (tiếng Anh): dạng của một "từ" thường dựa trên một "từ khác", cho dù có thể có nhiều từ khác xen vào:

Ví dụ 3: Phụ thuộc giữa các "từ" cách xa nhau

- Ví dụ 3.a (tiếng Anh): dạng của một "từ" thường dựa trên một "từ khác", cho dù có thể có nhiều từ khác xen vào:
  - **Sam sleeps** soundly.
  - **Dogs sleep** soundly.
  - **Sam**, the man with red hair who is my cousin, **sleeps** soundly.

Ví dụ 3: Phụ thuộc giữa các "từ" cách xa nhau

- Ví dụ 3.a (tiếng Anh): dạng của một "từ" thường dựa trên một "từ khác", cho dù có thể có nhiều từ khác xen vào:
  - **Sam sleeps** soundly. ①
  - **Dogs sleep** soundly. ②
  - **Sam**, the man with red hair who is my cousin, **sleeps** soundly. ③
- Ví dụ 3.b (tiếng Việt): sử dụng các "từ" khác nhau dựa vào "danh từ" để thể hiện sự thuận tiện/thích hợp:

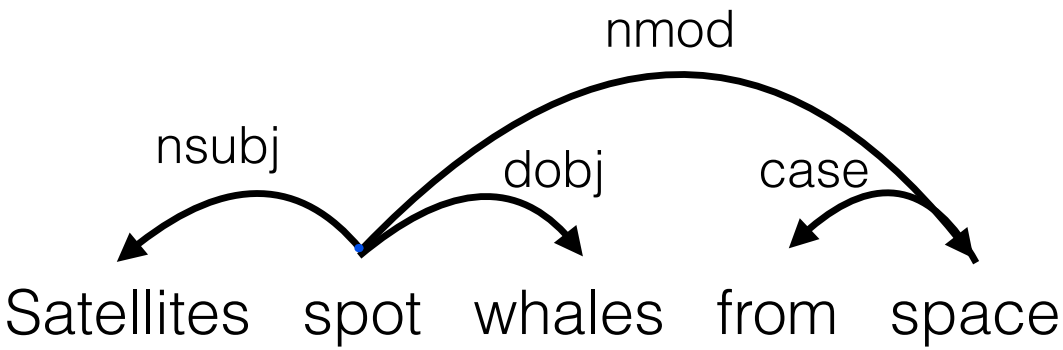
Ví dụ 3: Phụ thuộc giữa các "từ" cách xa nhau

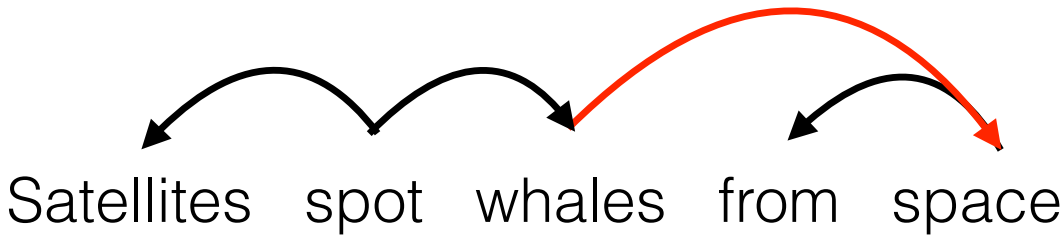
- Ví dụ 3.a (tiếng Anh): dạng của một "từ" thường dựa trên một "từ khác", cho dù có thể có nhiều từ khác xen vào:
  - **Sam sleeps** soundly.
  - **Dogs sleep** soundly.
  - **Sam**, the man with red hair who is my cousin, **sleeps** soundly.
- Ví dụ 3.b (tiếng Việt): sử dụng các "từ" khác nhau dựa vào "danh từ" để thể hiện sự thuận tiện/thích hợp:
  - Ngày mai **họp** có **tiện** cho chị không? (Is tomorrow **good** for you to have **the meeting**?)
  - Bây giờ là thời điểm **thích hợp** để đầu tư vào **thị trường chứng khoán**. (It's a **good** time to invest in **the stock market**.)

Ví dụ 3: Phụ thuộc giữa các "từ" cách xa nhau

→ Trong thực tế, chúng ta mong muốn có một mô hình ngôn ngữ có thể bắt được các sự phụ thuộc như có thể "**bắt**" (**capture**) được các thông tin như vừa nêu. Ví dụ như bài toán " **dịch máy**" (**machine translation**) và " **hiểu ngôn ngữ**" (**language understanding**).

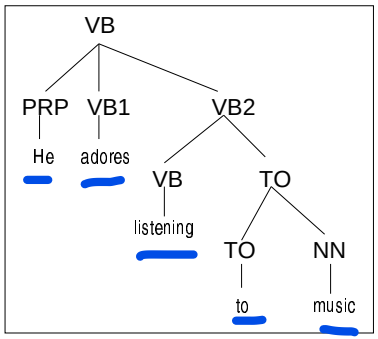
Ví dụ 4: Cấu trúc ngữ pháp phụ thuộc [2]





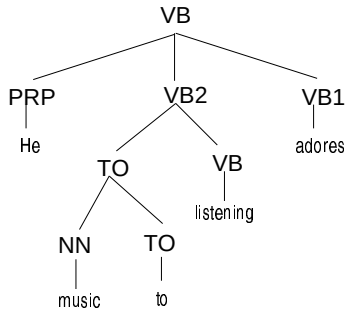
[2] Hình ảnh được tham khảo từ bài giảng của Karl Stratos.

Ví dụ 5: "Dịch máy" sử dụng "cấu trúc thành tố" làm biểu diễn trung gian[3]



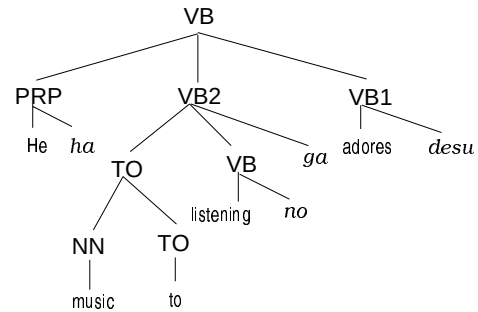
1. Channel Input

Reorder



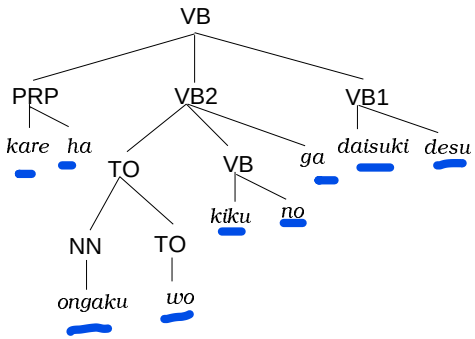
2. Reordered

Insert



3. Inserted

Translate



4. Translated

Reading off Leaves

kare ha ongaku wo kiku no ga daisuki desu

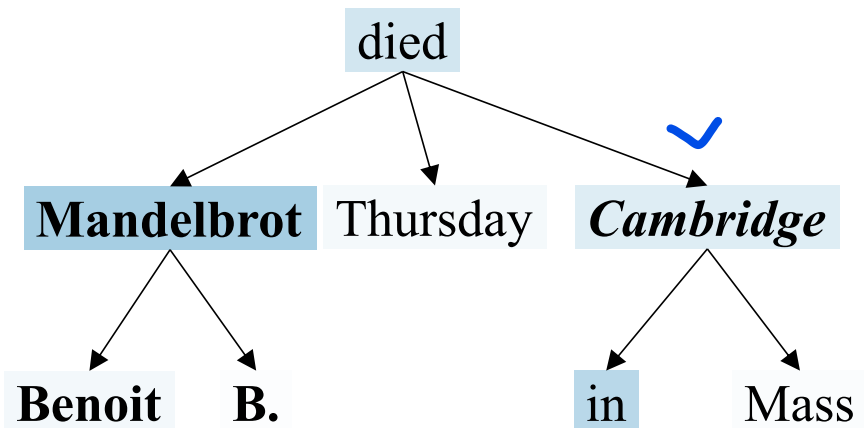
5. Channel Output

[3] [1] Yamada and Knight. "A Syntax-based Statistical Translation Model". 2001.

Ví dụ 6: "Rút trích quan hệ" sử dụng "cấu trúc phụ thuộc" làm biểu diễn trung gian[4]

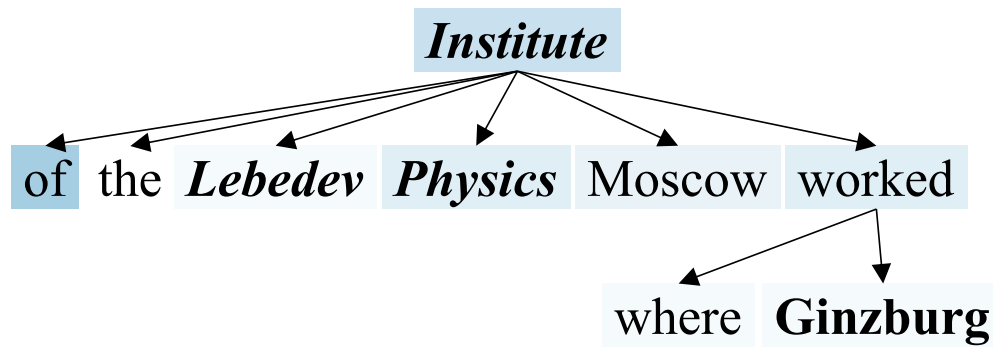
Relation: *per:city_of_death*

Benoit B. Mandelbrot, a maverick mathematician who developed an innovative theory of roughness and applied it to physics, biology, finance and many other fields, died Thursday in ***Cambridge***, Mass.



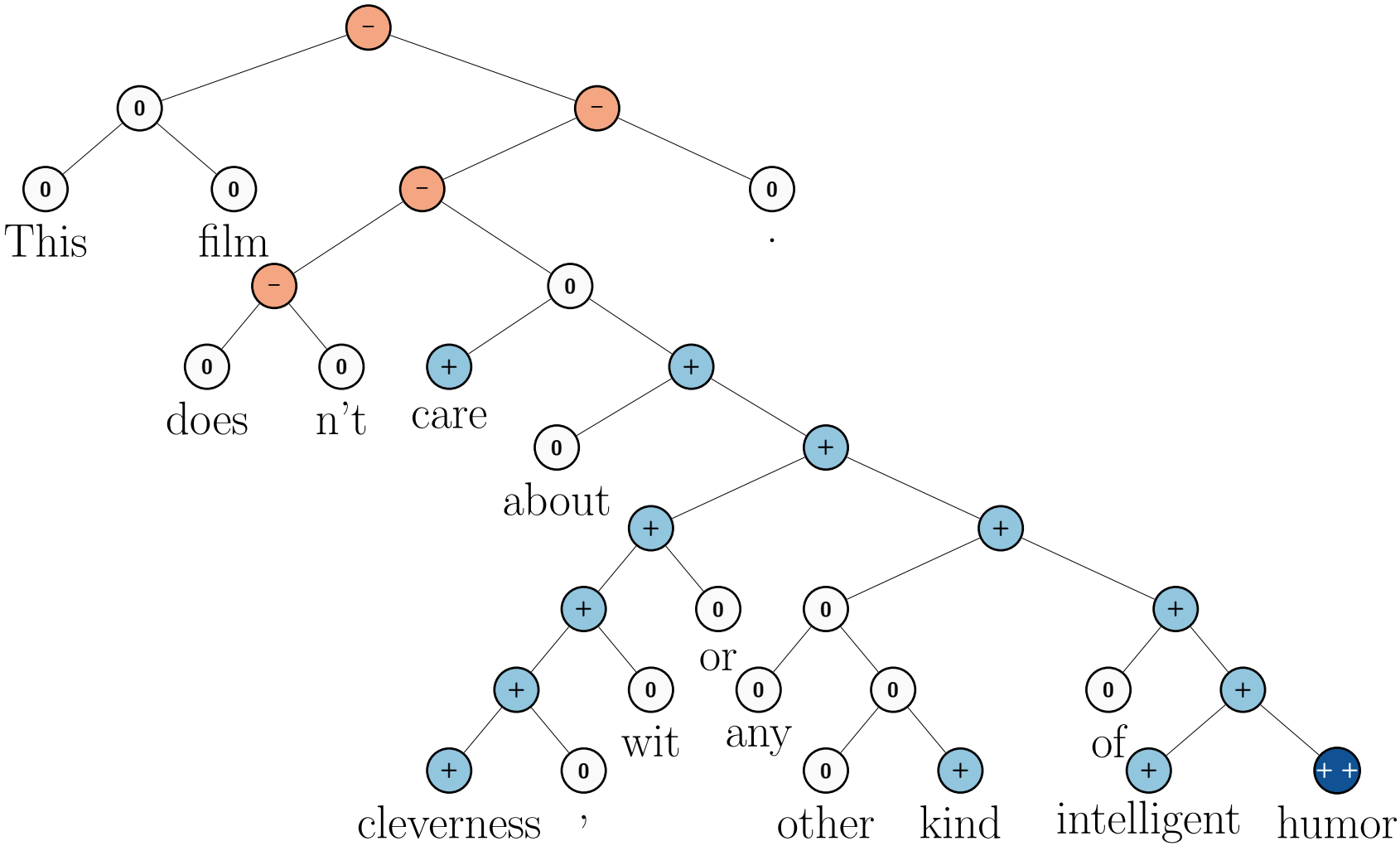
Relation: *per:employee_of*

In a career that spanned seven decades, Ginzburg authored several groundbreaking studies in various fields -- such as quantum theory, astrophysics, radio-astronomy and diffusion of cosmic radiation in the Earth's atmosphere -- that were of "Nobel Prize caliber," said Gennady Mesyats, the director of the ***Lebedev Physics Institute*** in Moscow, where **Ginzburg** worked .



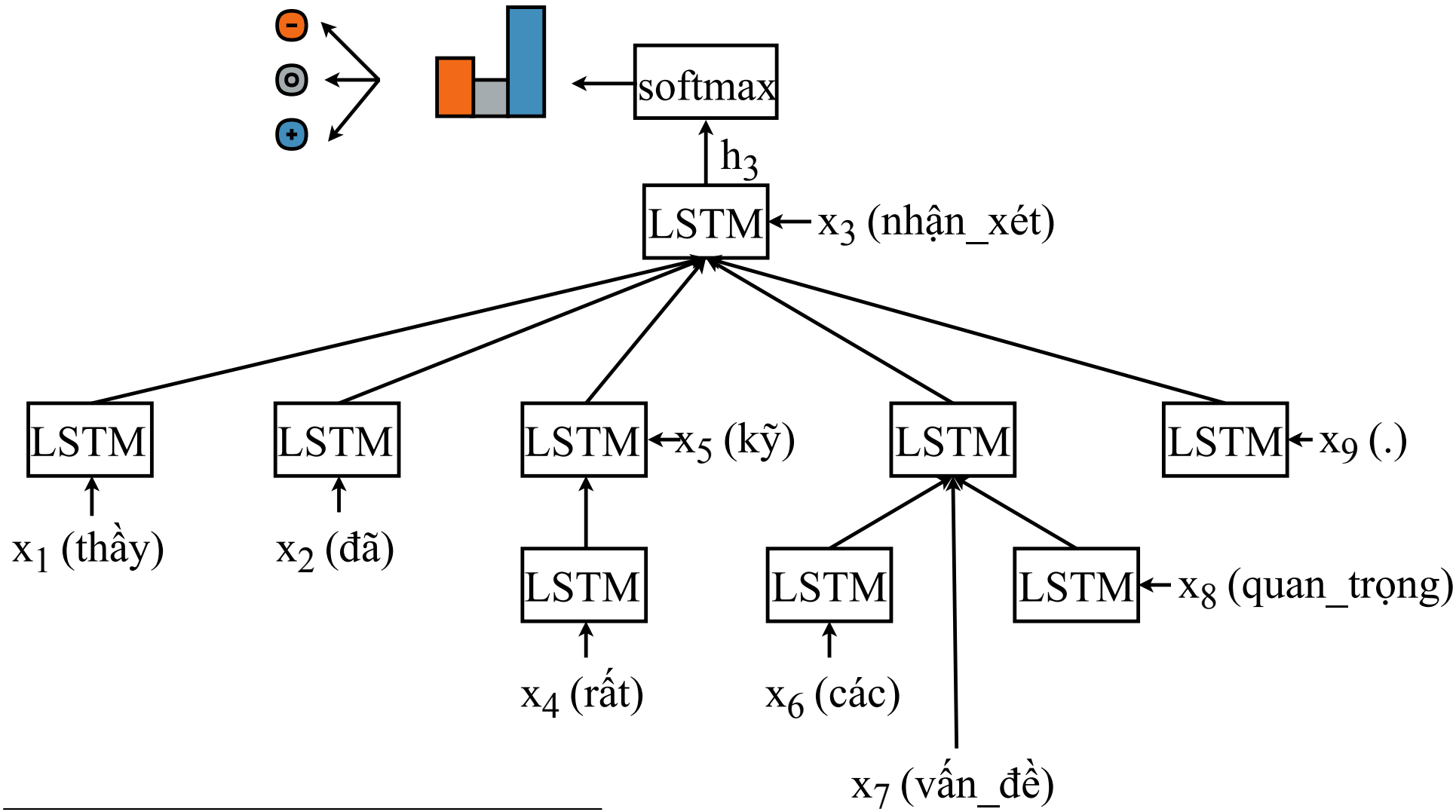
[4] [2] Zhang et al. "Graph Convolution over Pruned Dependency Trees Improves Relation Extraction". 2018.

Ví dụ 7: "Phân tích cảm xúc" sử dụng "cấu trúc thành tố" làm biểu diễn trung gian[5]



[5] [3] Socher et al. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank". 2013.

Ví dụ 8: "Phân tích cảm xúc" sử dụng "cấu trúc phụ thuộc" làm biểu diễn trung gian[6]



[6] [4] Tai et al. "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks". 2015.

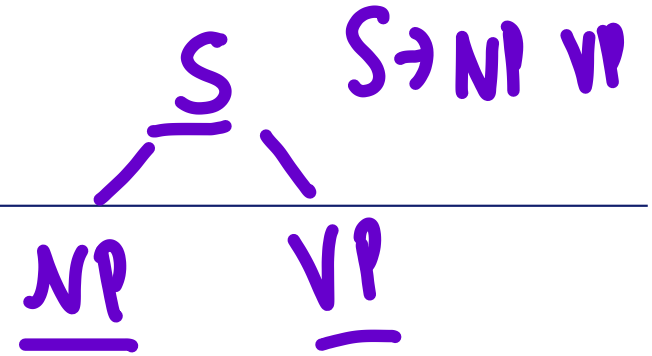
Tại sao chúng ta lại cần cấu trúc câu?

- Chúng ta cần hiểu cấu trúc câu để có thể giải thích ngôn ngữ một cách đúng đắn.
- Con người truyền tải những ý tưởng phức tạp bằng các "ghép" các "từ" với nhau thành các thành phần lớn hơn.
- Chúng ta cần biết được những "từ" nào **liên kết** với những "từ" nào trong một ngôn ngữ.

Tại sao chúng ta lại cần cấu trúc câu?

- Chúng ta cần hiểu cấu trúc câu để có thể giải thích ngôn ngữ một cách đúng đắn.
- Con người truyền tải những ý tưởng phức tạp bằng các "ghép" các "từ" với nhau thành các thành phần lớn hơn.
- Chúng ta cần biết được những "từ" nào **liên kết** với những "từ" nào trong một ngôn ngữ.

Văn phạm phi ngữ cảnh CFG (Context-free grammars[7])



- Một văn phạm phi ngữ cảnh $G = (N, \Sigma, R, S)$ trong đó:
  - N là tập hợp các ký hiệu non-terminal.
  - Σ là tập hợp các ký hiệu terminal.
  - R là tập hợp các luật có dạng $X \to Y_1 Y_2 \ldots Y_n$ với $n \geq 1, X \in N, Y_i \in (N \cup \Sigma)$.
  - $S \in N$ là ký hiệu bắt đầu câu văn.

[7] Văn phạm phi ngữ cảnh được phát triển vào giữa những năm 1950 bởi Noam Chomsky.

Ví dụ văn phạm phi ngữ cảnh cho tiếng Anh

N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}

S = S

Σ = {sleeps, saw, man, woman, dog, telescope, the, with, in}

$R =$

S	→	NP	VP
VP	→	Vi	
VP	→	Vt	NP
VP	→	VP	PP
NP	→	DT	NN
NP	→	NP	PP
PP	→	IN	NP

Vi	→	sleeps
Vt	→	saw
NN	→	man
NN	→	woman
NN	→	telescope
NN	→	dog
DT	→	the
IN	→	with
IN	→	in

S: sentence, VP: verb phrase, NP: noun phrase, PP: prepositional phrase, DT: determiner, Vi: intransitive verb, Vt: transitive verb, NN: noun, IN: preposition

Sinh ra các cây cú pháp từ văn phạm CFG bằng phép đệ quy trái

- Cho một văn phạm phi ngữ cảnh G, sinh ra cây cú pháp biểu diễn bởi dãy các chuỗi $s_1, s_2, \ldots, s_n$, trong đó:

Sinh ra các cây cú pháp từ văn phạm CFG bằng phép đệ quy trái

- Cho một văn phạm phi ngữ cảnh G, sinh ra cây cú pháp biểu diễn bởi dãy các chuỗi $s_1, s_2, \ldots, s_n$, trong đó:
  - $s_1 = S$.

Sinh ra các cây cú pháp từ văn phạm CFG bằng phép đệ quy trái

- Cho một văn phạm phi ngữ cảnh G, sinh ra cây cú pháp biểu diễn bởi dãy các chuỗi $s_1, s_2, \ldots, s_n$, trong đó:
  - $s_1 = S$.
  - $s_n \in \Sigma^*$ với Σ^* là tập hợp tất cả các chuỗi có thể được tạo từ tập từ vựng Σ. Mỗi chuỗi s_i $(i = 2, \ldots, n)$ được tạo ra từ chuỗi s_{i-1} bằng cách chọn ký hiệu non-terminal trái cùng trong chuỗi s_{i-1} và thay thế nó bởi β nào đó mà $X \to \beta \in R$.

Sinh ra các cây cú pháp từ văn phạm CFG bằng phép đệ quy trái

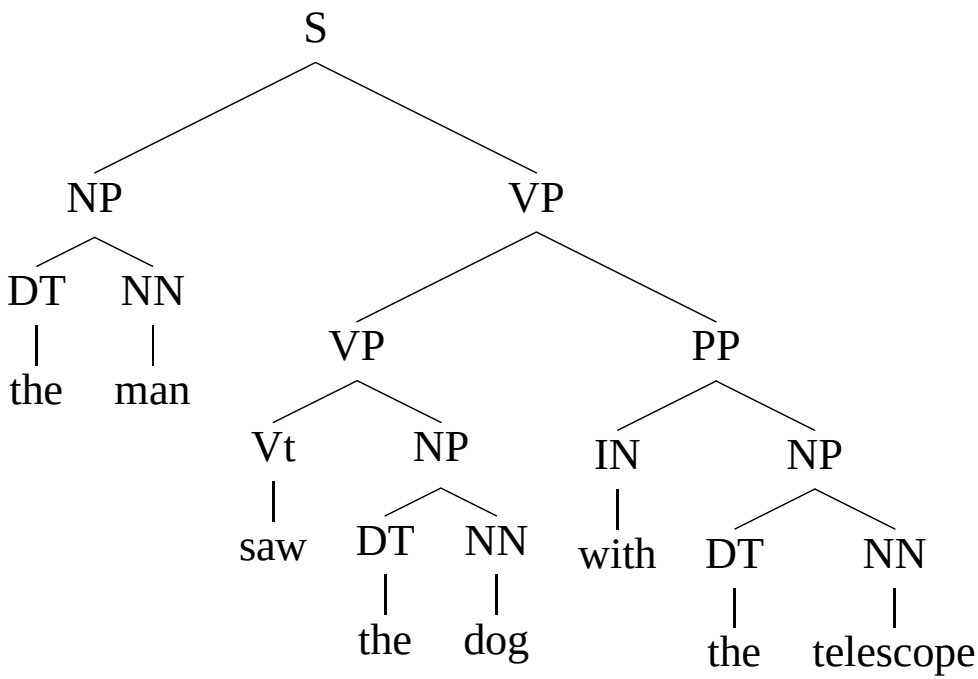
- Cho một văn phạm phi ngữ cảnh G, sinh ra cây cú pháp biểu diễn bởi dãy các chuỗi $s_1, s_2, \ldots, s_n$, trong đó:
  - $s_1 = S$.
  - $s_n \in \Sigma^*$ với Σ^* là tập hợp tất cả các chuỗi có thể được tạo từ tập từ vựng Σ. Mỗi chuỗi s_i $(i = 2, \ldots, n)$ được tạo ra từ chuỗi s_{i-1} bằng cách chọn ký hiệu non-terminal trái cùng trong chuỗi s_{i-1} và thay thế nó bởi β nào đó mà $X \to \beta \in R$.
- s_n là các nút lá của cây cú pháp.

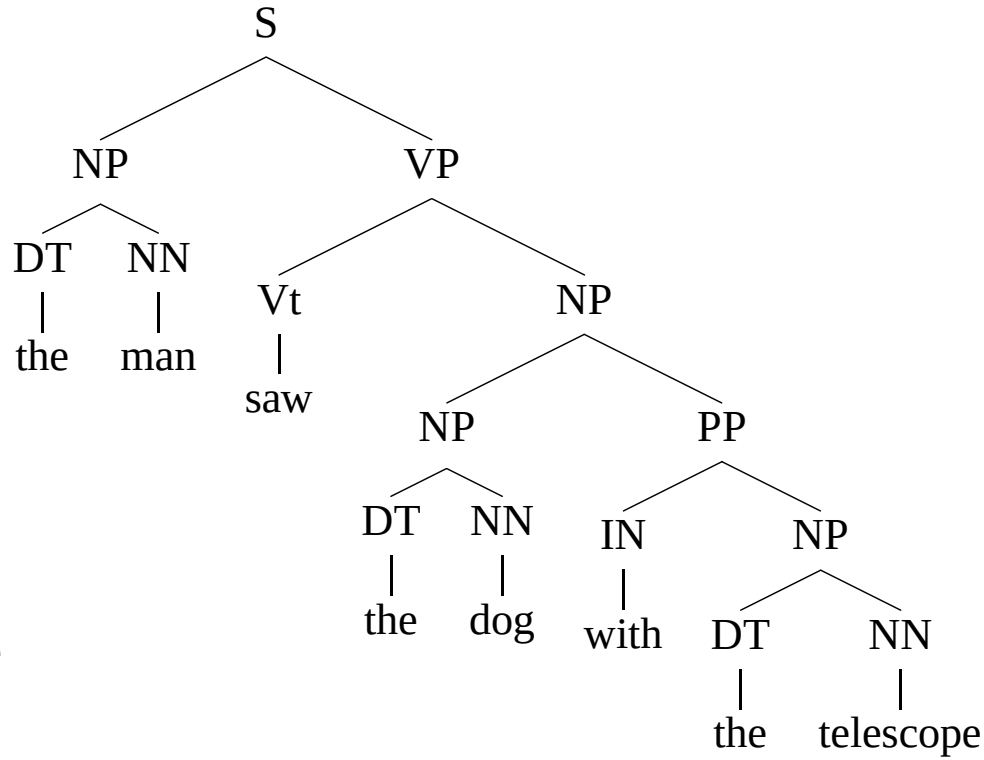
Sinh ra các cây cú pháp từ văn phạm CFG bằng phép đệ quy trái

- Cho một văn phạm phi ngữ cảnh G, sinh ra cây cú pháp biểu diễn bởi dãy các chuỗi $s_1, s_2, \ldots, s_n$, trong đó:
  - $s_1 = S$.
  - $s_n \in \Sigma^*$ với Σ^* là tập hợp tất cả các chuỗi có thể được tạo từ tập từ vựng Σ. Mỗi chuỗi s_i $(i = 2, \ldots, n)$ được tạo ra từ chuỗi s_{i-1} bằng cách chọn ký hiệu non-terminal trái cùng trong chuỗi s_{i-1} và thay thế nó bởi β nào đó mà $X \to \beta \in R$.
- s_n là các nút lá của cây cú pháp.
- Một chuỗi $s \in \Sigma^*$ nằm trong một ngôn ngữ được định nghĩa bởi CFG nếu tồn tại một cây cú pháp được sinh ra mà các nút lá là s.

Nhập nhằng [8]

- Một số câu văn có thể có nhiều hơn một cây cú pháp.





[8] Hình ảnh được tham khảo từ bài giảng của Michael Collins.

Phân tích cú pháp cổ điển

- Trong thực tế các câu văn có thể có số lượng rất lớn cách để phân tích.
- Rất khó để xây dựng một văn phạm với một độ phủ tất cả trường hợp của ngôn ngữ tự nhiên.
- Làm thế nào để chọn ra một cách phân tích cú pháp đúng cho một câu văn đầu vào?

Phân tích cú pháp dựa trên thống kê

- Học từ dữ liệu: ngân hàng cây cú pháp (treebanks).
- Thêm yếu tố xác suất vào các luật cú pháp: probabilistic CFGs (PCFGs).
- Ngân hàng cây cú pháp: là một tập hợp các câu văn với cây phân tích cú pháp của chúng, thường được các chuyên gia ngôn ngữ học thực hiện.
- Việc xây dựng ngân hàng cây cú pháp sẽ giúp chúng ta có các thông tin về tần số của các từ và cú pháp thường sử dụng, và giúp chúng ta có thể đánh giá hệ thống phân tích cú pháp.

Phân tích cú pháp dựa trên thống kê

- Một ví dụ về câu văn đã được phân tích cú pháp trong bộ dữ liệu VietTreebank[9]:

```
(S
    (NP (P Tôi))
    (VP
        (R rất)
        (V tự_hào)
        (VP (V là) (NP (Nc người) (N bạn) (PP (E của) (NP
(Ny Việt_Nam))))))
    (PU .))
```

[9] https://vlsp.org.vn/

Văn phạm phi ngữ cảnh có xác suất (PCFGs)

N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}
S = S
Σ = {sleeps, saw, man, woman, dog, telescope, the, with, in}

$R, q =$

S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4

- Một văn phạm phi ngữ cảnh có xác suất (PCFG) bao gồm:
  - Văn phạm phi ngữ cảnh $G = (N, \Sigma, R, S)$.
  - Với mỗi luật $\alpha \rightarrow \beta$, có một tham số $q(\alpha \rightarrow \beta) \geq 0$. Với mọi $X \in N$, ta có: $\sum_{\alpha \rightarrow \beta : \alpha = X} q(\alpha \rightarrow \beta) = 1$

Văn phạm phi ngữ cảnh có xác suất (PCFGs)

N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}
S = S
Σ = {sleeps, saw, man, woman, dog, telescope, the, with, in}

$R, q =$

S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4

- Với mọi cây cú pháp chứa các luật $\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2, \ldots, \alpha_l \rightarrow \beta_l$, xác suất của cây cú pháp là:

$$\prod_{i=1}^{l} q(\alpha_i \rightarrow \beta_i) \quad (1)$$

Trích xuất văn phạm PCFG từ một "treebank"

- Đầu vào: Dữ liệu huấn luyện chứa các cây cú pháp $t_1, t_2, \ldots, t_n$.
- Đầu ra:
  - N là tập hợp các ký hiệu non-terminal thấy được.
  - Σ là tập hợp các ký hiệu terminal thấy được.
  - $S \in N$ là ký hiệu bắt đầu câu văn.
  - R là tập hợp các luật $\alpha \rightarrow \beta$ thấy được và xác suất của luật được tính bằng cách:

$$q_{ML} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{count}(\beta)} \tag{2}$$

So sánh CFG và PCFG

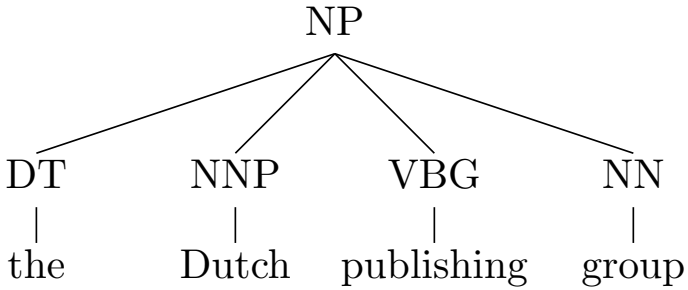
- Văn phạm CFG cho chúng ta biết liệu rằng một câu văn nào đó nằm trong ngôn ngữ mà nó định nghĩa.
- Văn phạm PCFG cho chúng ta cơ chế để tính toán điểm số (cụ thể là giá trị xác suất) cho các cây cú pháp khác nhau của một câu văn.

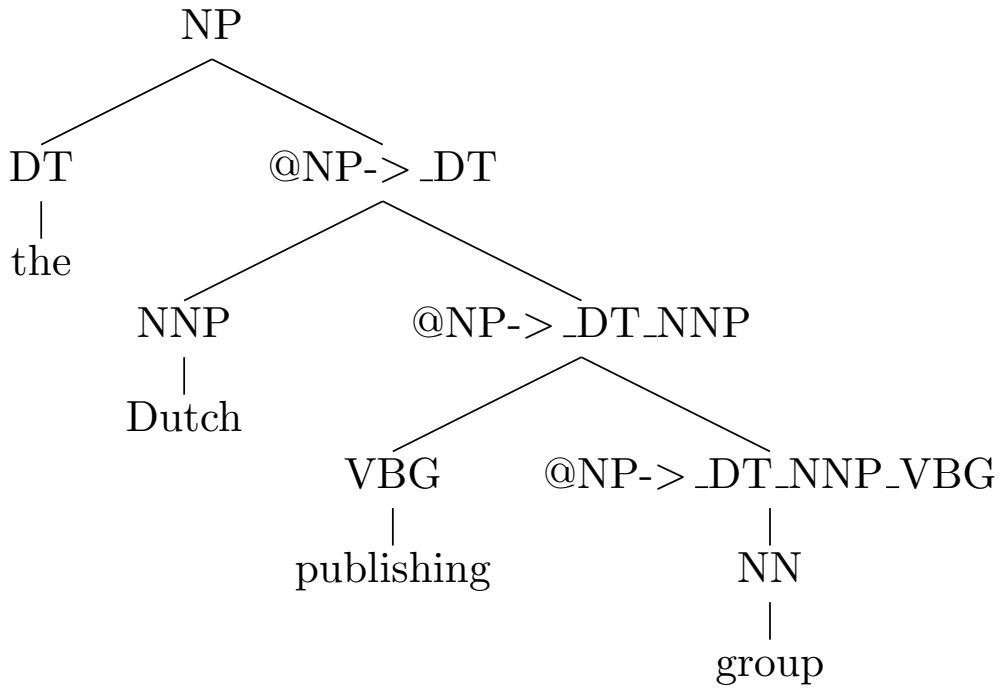
Phân tích cú pháp với một văn PCFG cho trước

- Cho biết một câu văn s và một văn phạm PCFG, làm thế nào để được giá trị xác suất cao nhất của cây cú pháp cho câu s?
- Thuật toán CKY: áp dụng vào PCFG với dạng chuẩn của Chomsky (Chomsky normal form - CNF)? Tại sao dạng chuẩn lại được sinh ra?
- Chomsky Normal Form (CNF): tất cả các luật phải tuân theo một trong hai chuẩn sau:
  - $X \rightarrow Y_1 Y_2$ trong đó $X \in N$, $Y_1 \in N$, $Y_2 \in N$
  - $X \rightarrow Y$ trong đó $X \in N$, $Y \in \Sigma$

Chuyển PCFGs sang CNF

- Các luật n-ary ($n > 2$): Ví dụ: NP $\rightarrow$ DT NNP VBG NN





- Các luật unary: Ví dụ: VP $\rightarrow$ Vi, Vi $\rightarrow$ sleeps
  - Loại bỏ tất cả các luật unary, rồi thêm vào luật VP $\rightarrow$ sleeps

Input: a sentence $s = x_1 \dots x_n$, a PCFG $G = (N, \Sigma, S, R, q)$.
Initialization:
For all $i \in \{1 \dots n\}$, for all $X \in N$,

$$\pi(i, i, X) = \begin{cases} q(X \rightarrow x_i) & \text{if } X \rightarrow x_i \in R \\ 0 & \text{otherwise} \end{cases}$$

Algorithm:

- For $l = 1 \dots (n-1)$
  - For $i = 1 \dots (n-l)$
    - Set $j = i + l$
    - For all $X \in N$, calculate

$$\pi(i, j, X) = \max_{\substack{X \rightarrow YZ \in R, \\ s \in \{i \dots (j-1)\}}} (q(X \rightarrow YZ) \times \pi(i, s, Y) \times \pi(s+1, j, Z))$$

and

$$bp(i, j, X) = \arg \max_{\substack{X \rightarrow YZ \in R, \\ s \in \{i \dots (j-1)\}}} (q(X \rightarrow YZ) \times \pi(i, s, Y) \times \pi(s+1, j, Z))$$

Output: Return $\pi(1, n, S) = \max_{t \in \mathcal{T}(s)} p(t)$, and backpointers bp which allow recovery of $\arg\max_{t \in \mathcal{T}(s)} p(t)$.

CKY for computing best parse

Compute score for span

Represent span