

Can I can the can?

Can **MD** 0.995509 **NN** 0.00417068 **VBP** 0.000320821

I **NP** 1

can **MD** 0.995509 **NN** 0.00417068 **VBP** 0.000320821

the **DT** 1

can **MD** 0.995509 **NN** 0.00417068 **VBP** 0.000320821

? **Fit** 1

Can/**MD** I/**NP** can/**VBP** the/**DT** can/**NN** ?/**Fit**

Ang. *part-of-speech tagging* (w skrócie *POS tagging*) oznacza oznaczanie słów, ale ponieważ analiza morfologiczna dostarcza oznaczeń (nie muszą to być tylko części mowy), w tej fazie następuje tylko ich ujednoznacznienie.

- Algorytmy analizy składniowej są kosztowne czasowo ($\mathcal{O}(n^3)$).
- W języku występuje naturalna niejednoznaczność (np. *struga*).
- Jedno zdanie może mieć tysiące różnych rozbiorów składniowych.
- Usuwanie niejednoznaczności jest znacznie tańsze niż analiza składniowa i pozwala wyeliminować znaczną część (niepoprawnych) rozbiorów składniowych.
- W niektórych zastosowaniach może zastąpić analizę składniową.

- Budowane **ręcznie** i z wbudowanym mechanizmem **uczenia się**. Pierwsze bardzo pracochłonne, drugie dużo powszechniejsze i pozwalające na łatwe dostosowanie do innych języków.
- Oparte o stosunkowo łatwe w interpretacji **reguły** i **statystyczne**. Do pierwszej kategorii zalicza się wszystkie systemy budowane ręcznie, ale też te oparte o transformacje, które uczone są na podstawie zbioru tekstów. Do drugiej – programy wykorzystujące niejawne modele Markowa (HMMs) itp.

Spójrzmy jeszcze raz na przykładowe zdanie:

Can I can the can?

Can MD NN VBP

I NP

can MD NN VBP

the DT

can MD NN VBP

? **Fit**

Jaka część mowy może wystąpić po rodzajniku *the*? Jakie słowa mogą występować na początku zdania przed zaimkiem osobowym w mianowniku? Co może wystąpić po nich, a przed *the*?

ENGTWOL jest przykładem programu do rozstrzygnięcia niejednoznaczności, który wykorzystuje ręcznie opracowane reguły. Reguły tworzą gramatykę ograniczeń – służą do usuwania niepoprawnych znaczników. Przykład reguły (dla języka angielskiego):

Mając na wejściu słowo „that”: **Jeżeli** następane słowo jest przymiotnikiem, przysłówkiem lub kwantyfikatorem, a po nim następuje granica zdania, zaś poprzednie słowo nie jest czasownikiem dopuszczającym przymiotniki jako dopełnienie, **to** usuń znaczniki inne niż przysłówkowe, **w przeciwnym wypadku** usuń znaczniki przysłówkowe.

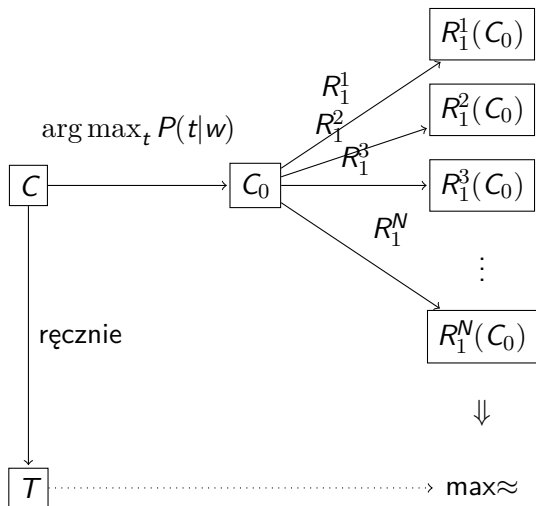
Algorytm Erica Brilla – szablon reguł

Algorytm Erica Brilla wykorzystuje szablon reguł, np.:

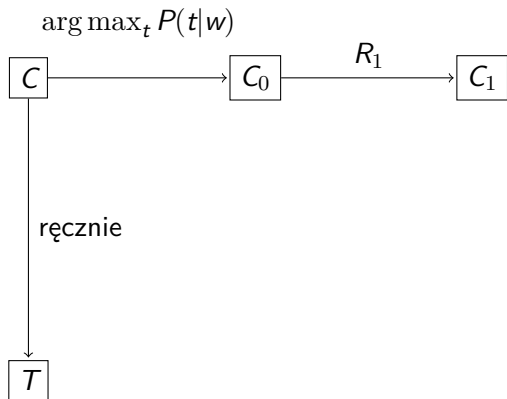
Szablon	Znaczenie: zamień A na B jeśli
A B PREVTAG C	poprzedni znacznik to C
A B PREV1OR2OR3TAG C	wśród poprz. 3 znaczników jest C
A B PREV1OR2TAG C	wśród poprz. 2 znaczników jest C
A B NEXT1OR2TAG C	wśród nast. 2 znaczników jest C
A B NEXTTAG C	następny znacznik to C
A B SURROUNDTAG C D	otaczające znaczniki to C i D
A B NEXTBIGRAM C D	następna para znaczników to C i D
A B PREVBIGRAM C D	poprzednia para znaczników to C i D

Szablony mogą też odwoływać się bezpośrednio do słów.

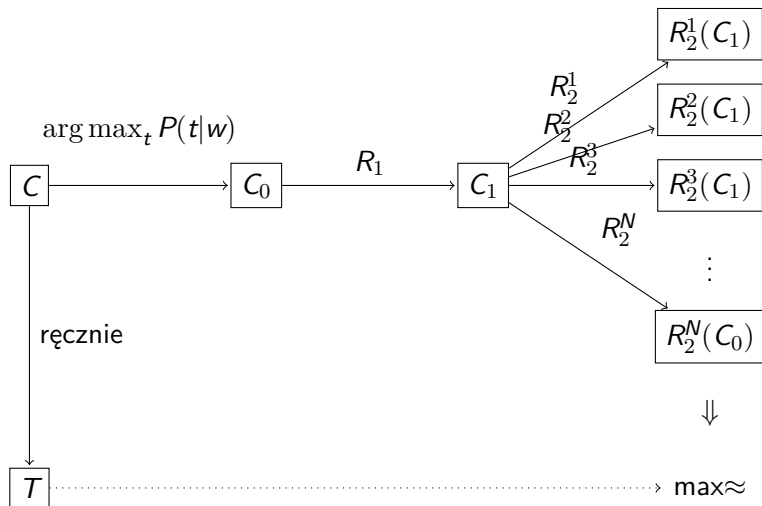
Algorytm Erica Brilla – uczenie



Algorytm Erica Brilla – uczenie



Algorytm Erica Brilla – uczenie



Dla danego zbioru tekstów C uzyskujemy (np. ręcznie) poprawne oznaczenia słów T . Zbiór C oznaczamy np. wybierając najbardziej prawdopodobny znacznik dla każdego słowa, uzyskując C_0 . Następnie wypróbujemy wszystkie możliwe dopasowania szablonów reguł, wybierając to (R_1), które zastosowane do C_0 najbardziej zbliży nas do T . Po zastosowaniu R_1 do C_0 otrzymujemy C_1 . Następnie dla C_1 szukamy reguły R_2 będącej dopasowaniem któregoś z szablonów, która najbardziej zbliży nas do T . Stosując R_2 do C_1 dostajemy C_2 . I tak dalej – dopóki jeszcze się uczymy.

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/VBP	the/DT	can/NN	?/Fit
C ₀ :	Can/MD	I/NP	can/MD	the/DT	can/MD	?/Fit

A B PREVTAG C: MD VBP PREVTAG NP, MD NN PREVTAG DT
A B NEXTTAG C: MD VBP NEXTTAG DT, MD NN NEXTTAG Fit
A B SURROUNDTAG C D: MD VBP SURROUNDTAG NP DT, MD NN
SURROUNDTAG DT Fit

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/VBP	the/DT	can/ NN	?/Fit
C ₀ :	Can/MD	I/NP	can/MD	the/ DT	can/ MD	?/Fit

A B PREVTAG C: MD VBP PREVTAG NP, MD NN PREVTAG DT
A B NEXTTAG C: MD VBP NEXTTAG DT, MD NN NEXTTAG Fit
A B SURROUNDTAG C D: MD VBP SURROUNDTAG NP DT, MD NN
SURROUNDTAG DT Fit

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/ VBP	the/DT	can/NN	?/Fit
C ₀ :	Can/MD	I/NP	can/ MD	the/ DT	can/MD	?/Fit

A B PREVTAG C: MD VBP PREVTAG NP, MD NN PREVTAG DT
A B NEXTTAG C: MD VBP NEXTTAG DT, MD NN NEXTTAG Fit
A B SURROUNDTAG C D: MD VBP SURROUNDTAG NP DT, MD NN
SURROUNDTAG DT Fit

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/VBP	the/DT	can/ NN	?/Fit
C ₀ :	Can/MD	I/NP	can/MD	the/DT	can/ MD	?/ Fit

A B PREVTAG C: MD VBP PREVTAG NP, MD **NN** PREVTAG DT
A **B** NEXTTAG **C**: MD VBP NEXTTAG DT, **MD** **NN** NEXTTAG **Fit**
A B SURROUNDTAG C D: MD VBP SURROUNDTAG NP DT, MD NN
SURROUNDTAG DT Fit

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/VBP	the/DT	can/NN	?/Fit
C ₀ :	Can/MD	I/NP	can/MD	the/DT	can/MD	?/Fit

A B PREV TAG C: MD VBP PREV TAG NP, MD NN PREV TAG DT
A B NEXT TAG C: MD VBP NEXT TAG DT, MD NN NEXT TAG Fit
A B SURROUND TAG C D: MD VBP SURROUND TAG NP DT, MD NN
SURROUND TAG DT Fit

Szablony – ukonkretnianie w uczeniu

T:	Can/MD	I/NP	can/VBP	the/DT	can/ NN	?/Fit
C ₀ :	Can/MD	I/NP	can/MD	the/ DT	can/ MD	?/ Fit

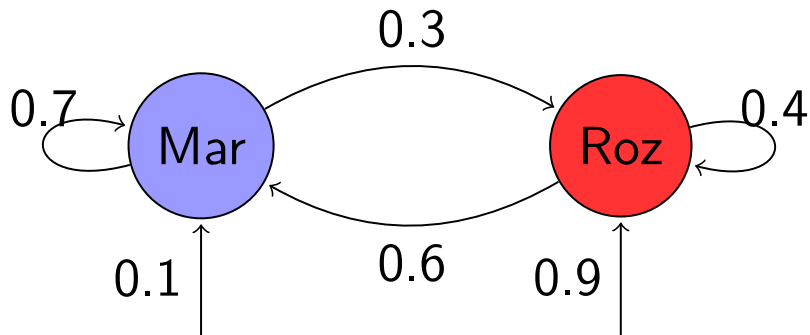
A B PREVTAG C: MD VBP PREVTAG NP, MD **NN** PREVTAG DT
A B NEXTTAG C: MD VBP NEXTTAG DT, MD NN NEXTTAG Fit
A **B** SURROUNDTAG **C** **D**: MD VBP SURROUNDTAG NP DT, **MD** **NN**
SURROUNDTAG **DT** **Fit**

- Uczymy na całym zbiorze \Rightarrow przeuczenie.
- Rozwiązanie: **wygładzanie**
 - ze zbioru uczącego zabieramy np. 10%,
 - uczymy na pozostałych 90%,
 - stosujemy reguły na wyodrębnionym zbiorze odrzucając te, które powodują pogorszenie wyników.

Mając nieoznaczone teksty, stosujemy nauczone reguły R_1, \dots, R_n po kolei. Kolejne reguły mogą wielokrotnie zmieniać te same znaczniki.

Możliwe jest znaczne przyspieszenie algorytmu przez realizację reguł w postaci (niedeterministycznych) automatów Mealy'ego, zastosowania złożenia, miejscowego rozszerzenia (szczególny przypadek determinizacji) i minimalizacji. Otrzymany automat Mealy'ego jest rzędu wielkości szybszy niż naiwne stosowanie zbioru reguł; jest też szybszy niż oznaczanie statystyczne i potrzebuje mało pamięci.

Niejawne modele Markowa



	ng	prz	bdb
Mar	0.2	0.7	0.1
Roz	0.1	0.3	0.6

HMM POS Tagging

w_i	słowo na pozycji i w zbiorze tekstów
t_i	znacznik (kategoria) słowa w_i
$w_{i,i+m}$	słowa na pozycjach od i do $i+m$
$t_{i,i+m}$	znaczniki słów $w_i \cdots w_{i+m}$
w^l	l -te słowo w słowniku
t^j	j -ty znacznik w zbiorze znaczników
$C(w^l)$	liczba wystąpień w^l w zbiorze tekstów
$C(t^j)$	liczba wystąpień t^j w zbiorze tekstów
$C(t^j, t^k)$	liczba wystąpień t^k bezpośrednio po t^j
$C(w^l : t^j)$	liczba wystąpień w^l oznaczonych jako t^j
T	liczba znaczników w zbiorze znaczników
W	liczba słów w słowniku
n	długość zdania

- **Uczenie nadzorowane**

- Znamy stany
- Stany to oznaczenia słów
- Uczenie z użyciem oznaczonych zbiorów tekstów

- **Uczenie bez nadzoru**

- Używamy nieoznaczonych zbiorów tekstów
- Dokładność znacznie spada

$$P(t^k | t^j) = \frac{C(t^j, t^k)}{C(t^j)}$$

$$P(w^l | t^j) = \frac{C(w^l, t^j)}{C(t^j)}$$

Szukamy:

$$\begin{aligned} \arg \max_{t_{1,n}} P(t_{1,n} | w_{1,n}) &= \arg \max_{t_{1,n}} \frac{P(w_{1,n} | t_{1,n}) P(t_{1,n})}{P(w_{1,n})} \\ &= \arg \max_{t_{1,n}} P(w_{1,n} | t_{1,n}) P(t_{1,n}) \end{aligned}$$

Przyjmujemy, że słowa są niezależne, a tożsamość słowa zależy tylko od jego oznaczenia:

$$\begin{aligned} P(w_{1,n} | t_{1,n}) P(t_{1,n}) &= \prod_{i=1}^n P(w_i | t_{1,n}) \cdot \\ &\quad \cdot P(t_n | t_{1,n-1}) \cdot P(t_{n-1} | t_{1,n-2}) \cdot \dots \cdot P(t_2 | t_1) \\ &= \prod_{i=1}^n P(w_i | t_i) \cdot \\ &\quad \cdot P(t_n | t_{n-1}) \cdot P(t_{n-1} | t_{n-2}) \cdot \dots \cdot P(t_2 | t_1) \\ &= \prod_{i=1}^n [P(w_i | t_i) \cdot P(t_i | t_{i-1})] \end{aligned}$$

Inicjalizacja (zdania są ograniczone kropkami, więc dokładamy kropkę na początku zdania):

$$\delta_1(t) = \begin{cases} 1.0 & t = \text{KROPKA} \\ 0.0 & t \neq \text{KROPKA} \end{cases}$$

Indukcja (tutaj $a_{jk} = P(t^k|t^j)$ i $b_{jkw'} = P(w^j|t^j)$):

$$\delta_{i+1}(t^j) = \max_{1 \leq k \leq T} [\delta_i(t^k) \cdot P(w_{i+1}|t^j) \cdot P(t^j|t^k)], \quad 1 \leq j \leq T$$

$$\psi_{i+1}(t^j) = \arg \max_{1 \leq k \leq T} [\delta_i(t^k) \cdot P(w_{i+1}|t^j) \cdot P(t^j|t^k)], \quad 1 \leq j \leq T$$

Zakończenie i odczytanie ścieżki:

$$X_n = \arg \max_{1 \leq j \leq T} \delta_n(t^j)$$

$$X_i = \psi_{i+1}(X_{i+1}), \quad 1 \leq i \leq n-1$$

$$P(X_1, \dots, X_n) = \max_{1 \leq j \leq T} \delta_{n+1}(t^j)$$

Często okazuje się, że potrzebujemy prawdopodobieństwa wystąpienia kombinacji słów i ich oznaczeń, które nie wystąpiły w zbiorze uczącym. Nie oznacza to, że nigdy nie wystąpią.

$$P(t^j | t^{j+1}) = (1 - \epsilon) \frac{C(t^{j-1}, t^j)}{C(t^{j-1})} + \epsilon$$

$$P(t^j | w^j) = \frac{C(t^j, w^j) + 1}{C(w^j) + K_I}$$

gdzie K_I to liczba możliwych oznaczeń w^j .

- 1 Emmanuel Roche, Yves Schabès, *Deterministic Part-of-Speech Tagging with Finite-State Transducers*, Computational Linguistics 21(2), pp. 227-253, czerwiec 1995. Dostępny pod:
<http://acl.ldc.upenn.edu/J/J95/J95-2004.pdf>
- 2 Walter Daelemans, Jakub Zavrel, Peter Berck, Steven Gillis, MBT: Memory-Based Part of Speech Tagger Generator, in Proceedings of the Fourth Workshop on Very Large Corpora, Copenhagen, Dania, str. 14-27, 1996. Dostępny pod:
<http://cwis.kub.nl/~fdl/general/people/daelem/papers/wvlc96.ps>