Przetwarzanie Języka Naturalnego
Lab 5 – Model języka

Aleksander Smywiński-Pohl
apohllo@agh.edu.pl

Wydział IEiT
Katedra Informatyki

28.03.2017

Definicja

Łańcuchem Markowa nazywamy proces stochastyczny w czasie dyskretnym i o dyskretnym zbiorze stanów bez pamięci.

Brak pamięci oznacza, że kolejny stan osiągany przez system zależy wprost jedynie od stanu bezpośrednio go poprzedzającego (oraz czasu).

Matematycznie

Ciąg zmiennych losowych $X_0, X_1 ...$ o wartościach całkowitych.

Warunek Markowa:
$\forall_{n \in \mathbb{N}} \forall_{i_0, i_1 ... i_n, i_{n+1} \in \mathbb{Z}}$:
$P(X_{n+1} = i_{n+1} | X_0 = i_0 \wedge X_1 = i_1 \wedge ... \wedge X_n = i_n) =$
$P(X_{n+1} = i_{n+1} | X_n = i_n) \overset{\text{ozn.}}{=} p(i_n, i_{n+1}) \overset{\text{ozn.}}{=} p_{i_n i_{n+1}}$

Łańcuch (czasowo) jednorodny - prawdopodobieństwo nie zależy od n

Macierz przejścia

$$P = \begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,m} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m,0} & p_{m,1} & \cdots & p_{m,m} \end{bmatrix}$$

$$x^{(n)} = x^{(n-1)} * P$$

$$x^{(n)} = x^{(0)} * P^n$$

Łańcuch Markowa rzędu m

Łańcuch Markowa rzędu m (z pamięcią m)

$$P(X_{n+1} = i_{n+1} | X_0 = i_0 \wedge X_1 = i_1 \wedge ... \wedge X_n = i_n) =$$
$$P(X_{n+1} = i_{n+1} | X_{n-m+1} = i_{n-m+1} \wedge ... \wedge X_n = i_n) \; m \in \mathbb{Z}_+$$

$$Y_n = (X_n, X_{n-1}, X_{n-2}, ..., X_{n-m+1})$$

Zastosowania

- modelowanie procesów fizycznych (termodynamika, mechanika)
- chemia
- modelowanie języka
- generowanie tekstów, muzyki
- teoria gier, sztuczna inteligencja
- ekonomia

Modelowanie języka

$$P(h_i) = P(h_i)^{\alpha}_{LM} * P(h_i)_{AM}$$
$$P(h_i)_{LM} = \prod_{w_j \in h_i} P(w_j | w_{j-N+1}, ..., w_{j-1})$$

Np. dla trigramów:

$$P(h_i)_{LM_{trigram}} = \prod_{w_j \in h_i} P(w_j | w_{j-2}, w_{j-1})$$

W praktyce:

$$logP(h_i)_{LM_{trigram}} = \sum_{w_j \in h_i} logP(w_j | w_{j-2}, w_{j-1})$$

Prawdopodobieństwo n-gramów

```
-2.387182       abażurze kinkietu
-1.753636       abażurze lampy  -0.4198931
-2.387182       abażurze lampy.
-2.387182       abażurze ma     -0.07864241
-2.387182       abażurze marnowanie
-1.490232       abażurze na
-2.387182       abażurze nie
-2.387182       abażurze nocnej
```

- logarytm o podstawie 10
- wartość na końcu to tzw. *back-off weight*

Użycie *back-off weight* dla n-gramów

$$P(w_j|w_{j-2}, w_{j-1})_{bow} = \begin{cases} P(w_j|w_{j-2}, w_{j-1}) & P > 0 \\ BOW(w_{j-2}, w_{j-1}) * P(w_j|w_{j-1}) & \end{cases}$$

Zadania

1. Zaimplementować algorytm obliczający Word Error Rate, jako odległość Levenshteina dla słów (0,5 pkt.)
2. Zaimplementować funkcję obliczającą prawdopodobieństwa zdania na podstawie otrzymanego pliku z modelem językowym (1,5 pkt.)
3. Określić optymalną wartość parametr α na podstawie akustycznych hipotez rozpoznawania mowy oraz modelu językowego (1 pkt.)

Materiały: `http://apohllo.pl/text/lab5.tar.gz`