Máster Universitario en Ingeniería Informática

Sistemas Inteligentes

Unit 5. N-grams and FSA - Theory

2022/2023

Outline

Outline

N-grams

- Assign a probability score to every word sequence
- Reflect a previous knowledge of a text source, predicting the most likely occurrence of words using its context

The observation probability of a word sequence $\mathbf{w} = \langle w_1 \ldots w_m \rangle$:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{i=2}^{m} P(w_i | w_1 \ldots w_{i-1}) \quad \textbf{\textit{with}} \quad w_i \in \Sigma$$

where $P(w_i | w_1 \ldots w_{i-1})$ is the probability of having a word w_i, given a previous word history $w_1 \ldots w_{i-1}$

The estimation of $P(\mathbf{w})$ is prohibitively expensive when vocabulary $|\Sigma|$ size becomes huge: $|\Sigma|^{i-1}$ different possible histories

N-grams

- Assign a probability score to every word sequence
- Reflect a previous knowledge of a text source, predicting the most likely occurrence of words using its context

The observation probability of a word sequence $\mathbf{w} = \langle w_1 \ldots w_m \rangle$:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{i=2}^{m} P(w_i | w_1 \ldots w_{i-1}) \quad \textbf{\textit{with}} \quad w_i \in \Sigma$$

where $P(w_i|w_1 \ldots w_{i-1})$ is the probability of having a word w_i, given a previous word history $w_1 \ldots w_{i-1}$

The estimation of $P(\mathbf{w})$ is prohibitively expensive when vocabulary $|\Sigma|$ size becomes huge: $|\Sigma|^{i-1}$ different possible histories

- Assign a probability score to every word sequence
- Reflect a previous knowledge of a text source, predicting the most likely occurrence of words using its context

The observation probability of a word sequence $\mathbf{w} = \langle w_1 \ldots w_m \rangle$:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{i=2}^{m} P(w_i | w_1 \ldots w_{i-1}) \quad \textbf{\textit{with}} \quad w_i \in \Sigma$$

where $P(w_i | w_1 \ldots w_{i-1})$ is the probability of having a word w_i, given a previous word history $w_1 \ldots w_{i-1}$

The estimation of $P(\mathbf{w})$ is prohibitively expensive when vocabulary $|\Sigma|$ size becomes huge: $|\Sigma|^{i-1}$ different possible histories

N-grams

- Assign a probability score to every word sequence
- Reflect a previous knowledge of a text source, predicting the most likely occurrence of words using its context

The observation probability of a word sequence $\mathbf{w} = \langle w_1 \dots w_m \rangle$:

$$P(\mathbf{w}) = P(w_1) \cdot \prod_{i=2}^{m} P(w_i | w_1 \dots w_{i-1}) \quad \textit{\textbf{with}} \quad w_i \in \Sigma$$

where $P(w_i | w_1 \dots w_{i-1})$ is the probability of having a word w_i, given a previous word history $w_1 \dots w_{i-1}$

The estimation of $P(\mathbf{w})$ is prohibitively expensive when vocabulary $|\Sigma|$ size becomes huge: $|\Sigma|^{i-1}$ different possible histories

$P(\mathbf{w})$ can be approximated by:

$$P(\mathbf{w}) \approx \prod_{i=1}^{m} P(w_i | \Phi_n(w_1 \ldots w_{i-1})) = \prod_{i=1}^{m} P(w_i | w_{i-n+1} \ldots w_{i-1})$$

The probability estimation of $P(w_i | w_{i-n+1} \ldots w_{i-1})$ is usually computed from relative frequency counts $f(\cdot|\cdot)$:

$$P(w_i | w_{i-n+1} \ldots w_{i-1}) = f(w_i | w_{i-n+1} \ldots w_{i-1}) = \frac{C(w_{i-n+1} \ldots w_{i-1} w_i)}{C(w_{i-n+1} \ldots w_{i-1})}$$

$P(\mathbf{w})$ can be approximated by:

$$P(\mathbf{w}) \approx \prod_{i=1}^{m} P(w_i|\Phi_n(w_1 \dots w_{i-1})) = \prod_{i=1}^{m} P(w_i|w_{i-n+1} \dots w_{i-1})$$

The probability estimation of $P(w_i|w_{i-n+1} \dots w_{i-1})$ is usually computed from relative frequency counts $f(\cdot|\cdot)$:

$$P(w_i|w_{i-n+1} \dots w_{i-1}) = f(w_i|w_{i-n+1} \dots w_{i-1}) = \frac{C(w_{i-n+1} \dots w_{i-1}\, w_i)}{C(w_{i-n+1} \dots w_{i-1})}$$

Example

$$\mathcal{S} = \{(\text{el perro corre rapido}, 2), (\text{el tren azul corre veloz}, 2), (\text{el coche azul corre veloz}, 2)\}$$

1-gram counts:

<s>	6
el	6
perro	2
corre	6
rapido	2
</s>	6
tren	2
azul	4
veloz	4
coche	2

2-gram counts:

<s> el	6
el perro	2
el tren	2
el coche	2
perro corre	2
corre rapido	2
corre veloz	4
rapido </s>	2
tren azul	2
azul corre	4
veloz </s>	4
coche azul	2

3-gram counts:

<s> el perro	2
<s> el tren	2
<s> el coche	2
el perro corre	2
el tren azul	2
el coche azul	2
perro corre rapido	2
corre rapido </s>	2
corre veloz </s>	4
tren azul corre	2
azul corre veloz	4
coche azul corre	2

- *n*-grams conditional probabilities can be estimated from raw text based on the relative frequency of word sequences

$$P(w_i | w_{i-n+1} \dots w_{i-1}) = \frac{C(qw_i)}{C(q)} = \frac{C(w_{i-n+1} \dots w_{i-1} w_i)}{C(w_{i-n+1} \dots w_{i-1})}$$

- An *n*-gram LM is said to be complete if all possible word sequences are represented ($|\Sigma|^n$) with an adequate estimation of their probabilities
- **Problem:** there are events (i.e. word sequences) that hardly ever or never occur in training → ***Smoothing or discounting methods***

- *n*-grams conditional probabilities can be estimated from raw text based on the relative frequency of word sequences

$$P(w_i | w_{i-n+1} \dots w_{i-1}) = \frac{C(qw_i)}{C(q)} = \frac{C(w_{i-n+1} \dots w_{i-1} w_i)}{C(w_{i-n+1} \dots w_{i-1})}$$

- An *n*-gram LM is said to be complete if all possible word sequences are represented ($|\Sigma|^n$) with an adequate estimation of their probabilities
- **Problem:** there are events (i.e. word sequences) that hardly ever or never occur in training $\rightarrow$ ***Smoothing or discounting methods***

N-grams

- n-grams conditional probabilities can be estimated from raw text based on the relative frequency of word sequences

$$P(w_i | w_{i-n+1} \dots w_{i-1}) = \frac{C(qw_i)}{C(q)} = \frac{C(w_{i-n+1} \dots w_{i-1} w_i)}{C(w_{i-n+1} \dots w_{i-1})}$$

- An n-gram LM is said to be complete if all possible word sequences are represented ($|\Sigma|^n$) with an adequate estimation of their probabilities
- **Problem:** there are events (i.e. word sequences) that hardly ever or never occur in training $\rightarrow$ ***Smoothing or discounting methods***

Smoothing Methods

- Back-Off
- Interpolation

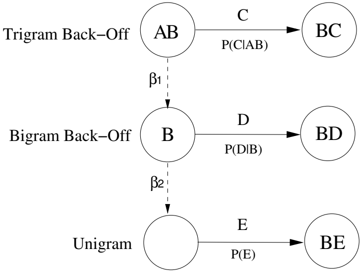
Discounting Methods

- Good Turing
- Absolute Discounting
- Witten Bell
- Linear Discounting
- Kneser-Ney
- . . .

n-grams Language Model: Back-Off Smoothing (1)

$$P(w_i|w_{i-n+1}^{i-1}) = \begin{cases} P^*(w_i|w_{i-n+1}^{i-1}) & \textit{if } C(w_{i-n+1}^{i}) > 0 \\ \beta(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1}) & \textbf{\textit{otherwise}} \end{cases}$$

where $P^*(\cdot|\cdot)$ is a discounted probability estimated to reserve mass for unseen events, and β is a back-off weight, which ensures the consistence of the model probabilities



Exercise 1. Compute the probability of the sentence "*el perro corre rapido*" with the 1-gram, the 2-gram, and the 3-gram that are introduced in the previous examples

Solution.

Probability with the 1-gram

$$
\begin{aligned}
p(\textit{el perro corre rapido}) &= p(<s>)\, p(\textit{el})\, p(\textit{perro})\, p(\textit{corre})\, p(\textit{rapido})\, p(</s>) \\
&= \frac{6}{40}\,\frac{6}{40}\,\frac{2}{40}\,\frac{6}{40}\,\frac{2}{40}\,\frac{6}{40} = 1.27\ 10^{-6}
\end{aligned}
$$

Probability with the 2-gram

$$
\begin{aligned}
p(\textit{el perro corre rapido}) &= p(\textit{el} \mid <s>)\, p(\textit{perro} \mid \textit{el})\, p(\textit{corre} \mid \textit{perro})\, p(\textit{rapido} \mid \textit{corre})\, p(</s> \mid \textit{rapido}) \\
&= \frac{6}{6}\,\frac{2}{6}\,\frac{2}{2}\,\frac{2}{6}\,\frac{2}{2} = 0.111
\end{aligned}
$$

Probability with the 3-gram

$$
\begin{aligned}
p(\textit{el perro corre rapido}) &= p(\textit{perro} \mid <s>\ \textit{el})\, p(\textit{corre} \mid \textit{el perro})\, p(\textit{rapido} \mid \textit{perro corre}) \\
&\quad p(</s> \mid \textit{corre rapido}) = \frac{2}{6}\,\frac{2}{2}\,\frac{2}{2}\,\frac{2}{2} = 0.333
\end{aligned}
$$

Applications of n-grams:

- Automatic speech recognition
- Machine translation
- Handwritten text recognition
- Spelling correction
- Text classification
- Plagiarism detection
- Laguage identification
- DNA sequence modeling
- ...

Outline

Introduction to Probabilistic Finite-State Automata

- ***Free Monoid*** Σ^****:*** Given a finite set Σ, Σ^+ is the set of all strings with finite length composed of elements from Σ

  $\Sigma^* = \Sigma^+ \cup \{\lambda\}$ (λ is *the empty string*)

- ***Grammar:*** $G = (N, \Sigma, R, S)$
  - N: Finite set of *non-terminal symbols*
  - Σ: Finite set of *terminal symbols* or *primitives*
  - $S \in N$: Initial non-terminal symbol or *"axiom"*
  - $R \subset (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$: set of *rules* or *productions*

    A rule is written as:

    $\boldsymbol{\alpha} \rightarrow \boldsymbol{\beta}, \quad \alpha \in (N \cup \Sigma)^* N (N \cup \Sigma)^*, \ \ \beta \in (N \cup \Sigma)^*$

- ***Elemental derivation:*** $\underset{G}{\Longrightarrow}$:

  $\mu\,\alpha\,\delta \underset{G}{\Longrightarrow} \mu\,\beta\,\delta \quad \longleftrightarrow \quad \exists(\alpha \to \beta) \in R, \;\; \mu, \delta \in (N \cup \Sigma)^*$

- ***Derivation*** $\overset{*}{\underset{G}{\Longrightarrow}}$:

  It is a *finite sequence of elemental derivations*

  A derivation d can be written as the corresponding sequence of rules of G

  The *set of derivations* of $y \in \Sigma^*$ (such that $S \overset{*}{\underset{G}{\Longrightarrow}} y$) is written as $D_G(y)$

  A grammar G is *ambiguous* if $\exists y \in \Sigma^*$ such that $|D_G(y)| > 1$

- ***Language generated by a grammar*** G, $\mathcal{L}(G)$:

$$\mathcal{L}(G) = \left\{ y \in \Sigma^* \mid S \overset{*}{\underset{G}{\Longrightarrow}} y \right\}$$

CHOMSKY HIERARCHY FOR RECURSIVE LANGUAGES

0: Non-restricted

1: Context-sensitive
$\alpha \rightarrow \beta \,, \qquad |\alpha| \leq |\beta|$

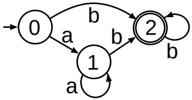
2: Context-free
$B \rightarrow \beta \,, \qquad B \in N$

3: ***Regular or "Finite-state"***
$A \rightarrow aB$ or $A \rightarrow a \,, \quad A, B \in N, \quad a \in \Sigma \cup \{\lambda\}$

- ***Regular grammars:*** $G = (N, \Sigma, R, S)$,
  Rules of R: $A \rightarrow aB \ \vee \ A \rightarrow a, \quad A, B \in N, \quad a \in \Sigma$
- ***Finite-state automaton:*** (non deterministic)
  $\mathcal{A} = (Q, \Sigma, \delta, q_0, F), \quad q_0 \in Q, \quad F \subseteq Q, \quad \delta : Q \times \Sigma \rightarrow 2^Q$
- ***Equivalence:*** For each regular grammar there exists a finite-state automaton that recognizes the same language[a]

Example:

$$\begin{aligned} G &= (N, \Sigma, R, S); \\ \Sigma &= \{a, b\}; \ N = \{S, A_1, A_2\}; \\ R &= \{\ S \ \rightarrow aA_1 \mid bA_2 \mid b, \\ A_1 &\rightarrow aA_1 \mid bA_2 \mid b, \\ A_2 &\rightarrow bA_2 \mid b\} \end{aligned}$$



$$\begin{aligned} \mathcal{A} &= \{Q, \Sigma, \delta, q_0, F\}; \\ Q &= \{0, 1, 2\}, \\ \Sigma &= \{a, b\}, \\ q_0 &= 0, \ F = \{2\} \end{aligned}$$

$$\mathcal{L}(G) = \{b, ab, bb, aab, abb, bbb, \ldots, aaabbbb, \ldots\} = \mathcal{L}(\mathcal{A})$$

[a] The reverse is not always true for stochastic languages!

Introduction to PFA

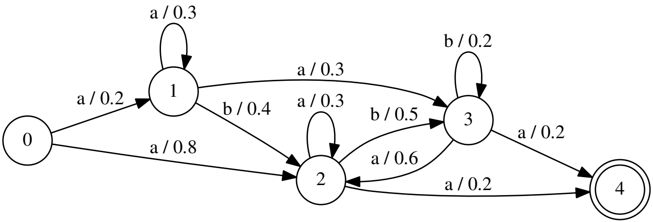
Probabilistic Finite-state Automata (PFA): it is a tuple $\mathcal{A} = \langle Q, \Sigma, \delta, I, F, P \rangle$, where:

- Q is a finite set of states
- Σ is the alphabet
- $\delta \subseteq Q \times \Sigma \times Q$ is a set of transitions
- $I : Q \rightarrow \mathbb{R}^{\geq 0}$ is the probability function of a state being an initial state
- $P : \delta \rightarrow \mathbb{R}^{\geq 0}$ is a probability function of transition between states
- $F : Q \rightarrow \mathbb{R}^{\geq 0}$ is the probability function of a state being a final state

I, P, and F are functions such that:

$$\begin{aligned} \sum_{i \in Q} I(i) &= 1 \\ \forall i \in Q, F(i) + \sum_{v \in \Sigma, j \in Q} P(i, v, j) &= 1 \end{aligned}$$

Example:



Being $S_{\mathcal{A}}(y)$ all the state sequences in $\mathcal{A}$ that generate y:

- ***Probability of a string generated by a PFA $\mathcal{A}$***

$$p_{\mathcal{A}}(y) = \sum_{s \in S_{\mathcal{A}}(y)} p_{\mathcal{A}}(y, s) \quad \rightarrow \textit{forward algorithm}$$

- ***Probability of the best sequence state for a string generated by a PFA $\mathcal{A}$***

$$p_{\mathcal{A}}(y) = \max_{s \in S_{\mathcal{A}}(y)} p_{\mathcal{A}}(y, s) \quad \rightarrow \textit{Viterbi algorithm}$$

Language defined by a PFA $\mathcal{A}$

$$\mathcal{L}(\mathcal{A}) \; = \; \{ y \in \Sigma^* \mid p_{\mathcal{A}}(y) > 0 \}$$

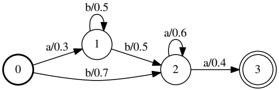
Relevant algorithms on PFA

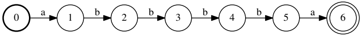
- PFA operations are underpinned by an algebraic structure called **semiring**: two operations, two monoids (one of them commutative), one operation distributive with respect to the other operation
- PFA and Weighted Finite State Automaton are equally powerful
- PFA is a special case of Weighted Finite State Transducers (not output label)

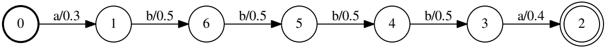
- PFA operations: composition, determinization, epsilon-removal, weight-pushing, minimization, projection, shortest-path, concatenation, pruning, . . .

PFA composition

Combine PFAs in such a way that the resulting PFA is the intersection of both languages





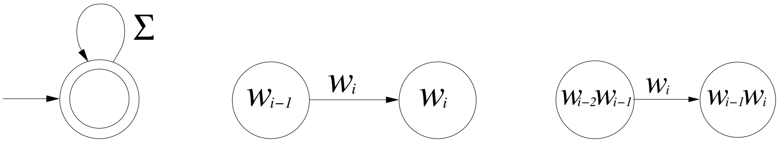


Outline

N-grams and FSA

n-grams can be represented using deterministic PFA

Examples of PFA representing 1-grams, 2-grams and 3-grams:



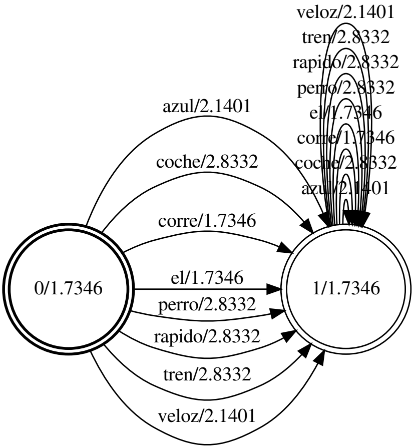
1-grams ***2***-grams: $w_{i-1}w_i$ ***3***-grams: $w_{i-2}w_{i-1}w_i$

For ***2***-grams: $q = w_{i-1}$ and $q' = w_i$.

For ***3***-grams: $q = w_{i-2}w_{i-1}$ and $q' = w_{i-1}w_i$.

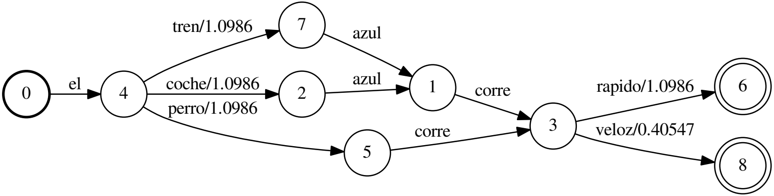
N-grams and FSA

Example: 1-gram



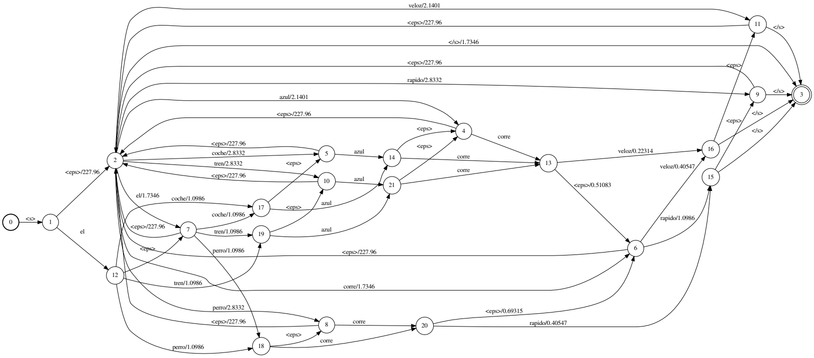
N-grams and FSA

Example: 2-gram

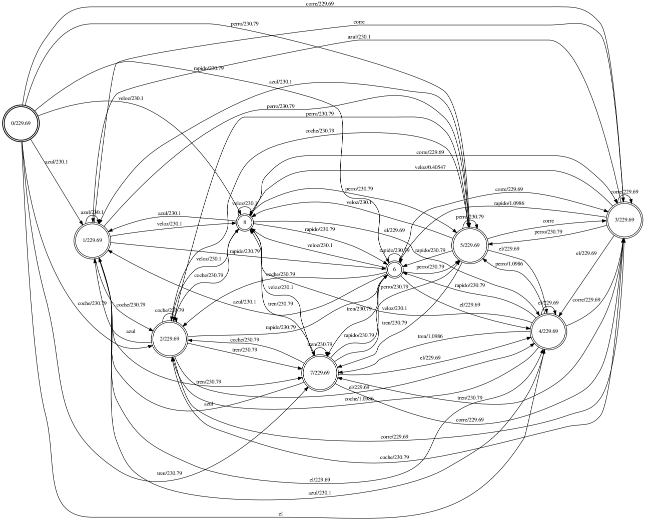


N-grams and FSA

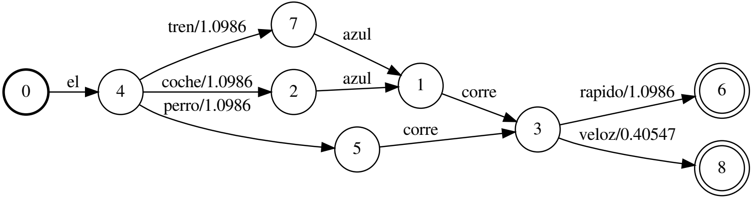
Example: 3-gram



Consequences of back-off as FSA: non-agressive prune (2-gram)



Consequences of back-off as FSA: agressive prune (2-gram)



Outline

References

- P. Dupont, F. Denis, and Y. Esposito. *Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms*, Pattern Recognition, 38:1349–1371, 2005.
- J.E. Hopcroft and J.D. Ullman.*Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.
- F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, 1998.
- R.A. Thompson *Determination of probabilistic grammars for funtionally specified probability-measure languages.* IEEE Transactions of Computers, c-23(6):603–614, 1974.
- E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, R. Carrasco, *Probabilistic finite-state machines - Part I*, IEEE Transactions on Pattern Analysis Machine Intelligence 27 (7):1013–1039, 2005.
- D. Jurafsky and J.H. Martin. *Speech and Language Processing*, Chapter 4, 2014.