Mining Unsupervised Data
5. Word sequences

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Outline

Outline

Goal

- Some types of word sequences within sentences are significantly relevant to understand Natural Language.
  - **Named entities (NEs)**: Classically, person, location, organization, date, time, money
    - Ex: "[John Smith]/PER was in [Picadilli Circus]/LOC at [3:00pm]/TIME"
    - Ex: "[Heart attack]/DISEASE at [8:30am]/TIME. Admitted to the intensive care unit at [St. James]/HOSPITAL
  - **Noun phrases (NPs)**: basic NPs only? complex NPs too?
    - Ex: "[Spaniards] usually enjoy [the original dishes] cooked by [Ferràn Adrià]"
    - Ex: "[Spaniards] usually enjoy [the original dishes cooked by Ferràn Adrià]"

  ...
- Goal: recognize and classify word sequences of these types (e.g., NERC and NP-chunking)

Motivation

Examples of applications:

- Anonymization: hide personal information occurring in private text
  - Ex: Names of person, adresses, telephones, etc. in clinical reports
- Information Extraction
  - Ex: Extract employees of companies, their positions and their salaries from financial news.
- Question answering: find the focus of some question types, or indexing documents
  - Ex: Who was [Albert Einstein]?
  - Ex: [Albert Einstein] was [the physicist who formulate the theory of relativity]
- Machine Translation, . . .

Word sequences

Methods

Outline

Methods

Based on hand-crafted rules:

- Used for robust cases (e.g., basic NPs or simple NEs such as telephones, e-mails, gene and protein names, ...)
- Can also be integrated in machine learning approaches

Methods

Based on hand-crafted rules:

- Used for robust cases (e.g., basic NPs or simple NEs such as telephones, e-mails, gene and protein names, . . .)
- Can also be integrated in machine learning approaches

Based on machine learning:

- Feature-based methods: Conditional Random Fields (CRFs) , SVMs, ...
- Deep-learning-based methods: data representation + context encoding + entity decoding
  - word embeddings + MLP + softmax
  - word embeddings + BiLSTM + CRF
  - LLMs
  - ...

Methods

Based on hand-crafted rules:

- Used for robust cases (e.g., basic NPs or simple NEs such as telephones, e-mails, gene and protein names, . . .)
- Can also be integrated in machine learning approaches

Based on machine learning:

- Feature-based methods: Conditional Random Fields (CRFs) , SVMs, ...
- Deep-learning-based methods: data representation + context encoding + entity decoding
  - word embeddings + MLP + softmax
  - word embeddings + BiLSTM + CRF
  - LLMs
  - ...

CRFs can perform better than deep learning methods in specific domains such as biomedicine.
Deep learning methods require large amounts of training data.

Outline

Hand-crafted rules for simple cases of NERC

- Patterns match words and/or POS-tags
- Lists of keywords and contextual words can be useful for some NE types

  Ex: Names of months, week days, special days for DATE

Hand-crafted rules for simple cases of NERC

- Patterns match words and/or POS-tags
- Lists of keywords and contextual words can be useful for some NE types

  Ex: Names of months, week days, special days for DATE

Example of pattern design: (with regular expression)

Input:
"My phone number is 934104433 . Call me on Tuesday 13 at 8:00 pm . "
Output:
"My phone number is [TEL 934104433] . Call me on [DATE Tuesday 13] at [TIME 8:00 pm] . "

Hand-crafted rules for simple cases of NERC

- Patterns match words and/or POS-tags
- Lists of keywords and contextual words can be useful for some NE types

  Ex: Names of months, week days, special days for DATE

Example of pattern design: (with regular expression)

Input:
"My phone number is 934104433 . Call me on Tuesday 13 at 8:00 pm . "
Output:
"My phone number is [TEL 934104433] . Call me on [DATE Tuesday 13] at [TIME 8:00 pm] . "

1. ... phone number is (\d+) ... → ... phone number is [TEL *match*] ...

Hand-crafted rules for simple cases of NERC

- Patterns match words and/or POS-tags
- Lists of keywords and contextual words can be useful for some NE types
  Ex: Names of months, week days, special days for DATE

Example of pattern design: (with regular expression)

Input:
"My phone number is 934104433 . Call me on Tuesday 13 at 8:00 pm . "
Output:
"My phone number is [TEL 934104433] . Call me on [DATE Tuesday 13] at [TIME 8:00 pm] . "

1. ... phone number is (\d+) ... → ... phone number is [TEL *match*] ...
2. DAY= '{Monday|Tuesday|Wednesday| ...}'
   ... on ($DAY \d+) ... → ... on [DATE *match*]

Hand-crafted rules for simple cases of NERC

- Patterns match words and/or POS-tags
- Lists of keywords and contextual words can be useful for some NE types
  Ex: Names of months, week days, special days for DATE

Example of pattern design: (with regular expression)

Input:
"My phone number is 934104433 . Call me on Tuesday 13 at 8:00 pm . "
Output:
"My phone number is [TEL 934104433] . Call me on [DATE Tuesday 13] at [TIME 8:00 pm] . "

1. ... phone number is (\d+) ... → ... phone number is [TEL *match*] ...
2. DAY= '{Monday|Tuesday|Wednesday| ...}'
   ... on ($DAY \d+) ... → ... on [DATE *match*]
3. SLOT= '{pm|p.m.|p.m|am|a.m.|a.m}'
   ... at (\d{1:2}:\d\d $SLOT) ... → ... at [TIME *match*] ...

Hand-crafted rules for basic-NP chunking

- Patterns match POS-tags
- Patterns use syntactic information

Hand-crafted rules for basic-NP chunking

- Patterns match POS-tags
- Patterns use syntactic information

Example of pattern design: (with regular expression)

Input:
"The:DT cat:NN eats:VBZ in:IN the:DT dark:JJ room:NN "
Output:
"[NP The:DT cat:NN] eats:VBZ in:IN [NP the:DT dark:JJ room:NN] "

Hand-crafted rules for basic-NP chunking

- Patterns match POS-tags
- Patterns use syntactic information

Example of pattern design: (with regular expression)

Input:
"The:DT cat:NN eats:VBZ in:IN the:DT dark:JJ room:NN "
Output:
"[NP The:DT cat:NN] eats:VBZ in:IN [NP the:DT dark:JJ room:NN] "

1. ... (\w+:DT \w+:NN) ... → ... [NP *match*] ...
2. ... (\w+:DT (\w+:JJ)+ \w+:NN) ... → ... [NP *match*] ...

Hand-crafted rules for basic-NP chunking

- Patterns match POS-tags
- Patterns use syntactic information

Example of pattern design: (with regular expression)

Input:
"The:DT cat:NN eats:VBZ in:IN the:DT dark:JJ room:NN "
Output:
"[NP The:DT cat:NN] eats:VBZ in:IN [NP the:DT dark:JJ room:NN] "

1. ... (\w+:DT \w+:NN) ... → ... [NP *match*] ...
2. ... (\w+:DT (\w+:JJ)+ \w+:NN) ... → ... [NP *match*] ...

OR

1. ... (\w+:DT (\w+:JJ)* \w+:NN) ... → ... [NP *match*] ...

Exercise

1. Provide NERC patterns for expressions similar to the following ones:
   a) "tomorrow:NN morning::NN", ïn:IN the:DT evening:NN""after:IN this:DT Sunday:NN"
   b) "5:CD €:NN", "one:CD million:CD dollars:NNS"
   c) "ana.sanchez@gmail.com", "ana.sanchez at gmail dot com"
2. Provide patterns to recognize the basic NP-chunks of the following POS-tagged sentences:
   d) "We:PRP 're:VB going:VBG to:TO the:DT best:JJ cinema:NN with:IN Gina:NNP 's:RP father:NN and:CC 24:CD friends:NNS"
   e) "Workers:NNS of:IN car:NN parks:NNS hate:VB working:VBG after:IN 7:00:Z pm:NN "
3. Is the use of *hand-crafted rules* a suitable technique for all the types of sequences involved?

Outline

Representation of the examples with BIO labels

Manually labelled sentence in training corpus:

$$w_1\ w_2\ w_3 \ldots [CLASS\ w_i\ w_{i+1}] \ldots w_n$$

Is transformed into:

$$w_1{:}O\ w_2{:}O\ w_3{:}O\ \ldots w_i{:}B\text{-}CLASS\ w_{i+1}{:}I\text{-}CLASS \ldots w_n{:}O$$

BIO code: B: beginning; I: inside; O: outside
BIOS code: S: single token (many sequences of 1 token)
BIOES code [BILOU]: E: end

Examples:

- "The president of [LOC the US] , [PER D. Trump]"

  "The:O president:O of:O the:B-LOC US:I-LOC ,:O D.:B-PER Trump:I-PER"
- "[NP The president] of [NP the US] , [NP D. Trump]"

  "The:B president:I of:O the:B US:I ,:O D.:B Trump:I"

Outline

Conditional Random Fields

- Generalization of HMMs
- HMMs: Naïve Bayes applied to a sequence.
  - Based on join probability (Generative model)

$$P(X|O) \approx P(X,O) = P(X_1,\ldots,X_T) \cdot P(O_1,\ldots,O_T|X_1,\ldots,X_T)$$

- CRFs: logistic regression applied to a sequence
  - Based on conditional probability (Discriminative model)

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

$$Z(O) = \sum_X exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

f_k are binary feature functions over states $X_{t-1} = s_i$ and $X_t = s_j$ (Markov property) and over observations from O

Learning of parameters λ_i

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

Briefly:

- Maximize the log-likelihood of labelled sequences occurring in some training data
- Optimization procedures: quasi-Newton methods, conjugate gradient, iterative scaling

This topic is out of this course

Types of feature functions

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

1. **Of observations**:

Ex: $f_1(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and has\_property}(o_t) \\ 0 & \text{otherwise} \end{cases}$

Types of feature functions

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

1. **Of observations**:

$$\text{Ex: } f_1(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and has\_property}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

2. **Of transitions**:

$$\text{Ex: } f_2(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and } x_{t-1} = s_6 \\ 0 & \text{otherwise} \end{cases}$$

Types of feature functions

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

1. **Of observations**:

$$\text{Ex: } f_1(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and has\_property}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

2. **Of transitions**:

$$\text{Ex: } f_2(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and } x_{t-1} = s_6 \\ 0 & \text{otherwise} \end{cases}$$

3. **Hybrid**:

$$\text{Ex: } f_3(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = s_3 \text{ and } x_{t-1} = s_6 \text{ and } o_t = w_4 \\ 0 & \text{otherwise} \end{cases}$$

Feature Templates

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

1. **Of observations**:

   Ex: $f_{1,a,b_1,\ldots,b_k}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = a \text{ and has\_property}(o_t) \\ 0 & \text{otherwise} \end{cases}$

2. **Of transitions**:

   Ex: $f_{2,a,c(x_{t-1}}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = a \text{ and } x_{t-1} = c \\ 0 & \text{otherwise} \end{cases}$

3. **Hybrid**:

   Ex: $f_{3,a,b_i,c}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = a \text{ and } x_{t-1} = c \text{ and } o_t = b_i \\ 0 & \text{otherwise} \end{cases}$

Correct functions *vs.* useful functions

$$P(X|O) = \frac{1}{Z(O)} \cdot exp(\sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t))$$

- Correct functions:
  - x_t defined
  - other elements apart from parameters are not included
- Useful function:
  - it makes sense for the task
  - $\lambda_i \neq 0$

Modeling NERC with CRFs

- States s_i are tags B-CLASS, I-CLASS, O for all possible NE classes.
- Feature templates can be designed as feature function generalizations.

Ex: The current word is capitalized and its tag is a

$$f_{1,a}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = a \text{ and capitalized}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

Modeling NERC with CRFs

- States s_i are tags B-CLASS, I-CLASS, O for all possible NE classes.
- Feature templates can be designed as feature function generalizations.

  Ex: The current word is capitalized and its tag is a

$$f_{1,a}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t = a \text{ and capitalized}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

- Feature functions are automatically generated from feature templates. Some of them will be irrelevant ($\lambda_i = 0$)

  Ex: Two feature function generated from $f_{1,a}$

$$f_{1,\text{B-PER}}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t\text{=B-PER and capitalized}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

$$f_{1,\text{O}}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } x_t\text{=O and capitalized}(o_t) \\ 0 & \text{otherwise} \end{cases}$$

Modeling NP-chunking with CRFs

- States s_i are tags B, I, O as there is only one class (NP).
- Feature templates.

  Ex: The POS of the current word is a and the current tag is b

$$f_{1,a,b}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if } \mathrm{pos}(o_t){=}a \text{ and } x_t = b \\ 0 & \text{otherwise} \end{cases}$$

Modeling NP-chunking with CRFs

- States s_i are tags B, I, O as there is only one class (NP).
- Feature templates.

  Ex: The POS of the current word is a and the current tag is b

$$f_{1,a,b}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if pos}(o_t){=}a \text{ and } x_t = b \\ 0 & \text{otherwise} \end{cases}$$

- Feature functions.

  Ex: Three feature functions automatically generated from $f_{1,a,b}$:

$$f_{1,\mathrm{DT},\mathrm{B}}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if pos}(o_t){=}\mathrm{DT} \text{ and } x_t{=}\mathrm{B} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{1,\mathrm{NN},\mathrm{I}}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if pos}(o_t){=}\mathrm{NN} \text{ and } x_t{=}\mathrm{I} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{1,\mathrm{VB},\mathrm{O}}(x_{t-1}, x_t, O, t) = \begin{cases} 1 & \text{if pos}(o_t){=}\mathrm{VB} \text{ and } x_t{=}\mathrm{O} \\ 0 & \text{otherwise} \end{cases}$$

Exercise

Write the feature templates for the following descriptions. Provide examples of feature functions generated from them.

Usually for NERC:

- The previous tag is *a*, the current tag is *b* and the current word is capitalized
- The current tag is *a* and the next word is *w*
- A person name can be preceded by a title (mr., dr.,. . .)

Usually for NP-chunking:

- The POS of the current word is *a* and the current tag is *b*
- The POS of the previous word is *a*, the previous tag is *b* and the current tag is *c*

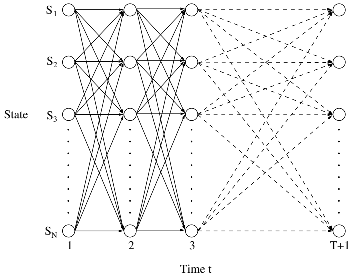
How is the best sequence found?

- We want to find

$$\hat{X} = \operatorname*{argmax}_X P(X|O,\lambda) = \operatorname*{argmax}_X \exp \sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t)$$

$$= \operatorname*{argmax}_X \sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t)$$

- Viterbi algorithm can be easily modified for CRFs



Trellis of a fully connected CRF.

A node $\{s_j, t\}$ of the trellis stores information about states sequences which include $X_t = s_j$.

$$\{s_j, t\}: \quad \delta_t(j) = \max_{X_1,\cdots,X_{t-1}} P(X_1,\cdots,X_{t-1},s_j|O,\lambda)$$

$$\varphi_t(j) = last(\operatorname*{argmax}_{X_1,\cdots,X_{t-1}} P(X_1,\cdots,X_{t-1},s_j|O,\lambda))$$

How is the best sequence found?

- We want to find

$$\hat{X} = \operatorname*{argmax}_X \sum_t \sum_k \lambda_k \cdot f_k(x_{t-1}, x_t, O, t)$$

- Viterbi algorithm can be easily modified for CRFs
  1. Initialization: $\forall j = 1 \ldots N$

$$\delta_1(j) = \sum_k \lambda_k \cdot f_k(x_0 = *, x_1 = s_j, O, t)$$

  2. Induction: $\forall j = 1 \ldots N$

$$\delta_t(j) = \max_i \left[\delta_{t-1}(i) + \sum_k \lambda_k \cdot f_k(x_{t-1} = s_i, x_t = s_j, O, t)\right]$$

$$\varphi_t(j) = \operatorname*{argmax}_i \left[\delta_{t-1}(i) + \sum_k \lambda_k \cdot f_k(x_{i-1} = s_i, x_i = s_j, O, t)\right]$$

  3. Termination:

$$\hat{X}_T = \operatorname*{argmax}_i \delta_T(i)$$

  4. Backward path readout:

$$\hat{X}_t = \varphi_{t+1}(\hat{X}_{t+1})$$