Assignment 10 + Conditional Random Fields

(SNLP tutorial 11)

Vilém Zouhar, Awantee Deshpande, Julius Steuer

6th, 8th July

Organisation

- Check that you have the finalised versions of the tutorial slides (https://github.com/zouharvi/uds-snlp-tutorial/tree/main)
- Check if you are eligible for the exam, and register accordingly.
- Project will be released on Friday, expected deadline at the end of August (tentatively 20th Aug), will be specified in the project instructions.
- Next week's tutorial discussion: Open Q&A.
- Send a list of questions to me (Teams or Private Piazza Post) by Sunday, 11th July.
- Discussion of sample exam
- Other questions. . . ?

Assignment 10

- Exercise 1: Lesk's Algorithm
- Exercise 2: Expectation Maximisation
- Exercise 3: Yarowsky Algorithm

Overview

- Sequence Labelling / Entity Recognition
  - Rule-based
  - HMM
  - Bayesian Network
  - Log-linear 1st Order Sequential Model
  - Linear Chain CRF / CRF
- Model comparison
- Implementations

Sequence Labelling / Entity Recognition

- `My name is John, I live in Saarbrücken, and my matriculation number is 1234.`

Sequence Labelling / Entity Recognition

- `My name is John, I live in Saarbrücken, and my matriculation number is 1234.`
- `My name is [John:PERSON], I live in [Saarbrücken:LOC], and my matriculation number is [1234:MATNUM].`

Sequence Labelling / Entity Recognition

- `My name is John, I live in Saarbrücken, and my matriculation number is 1234.`
- `My name is [John:PERSON], I live in [Saarbrücken:LOC], and my matriculation number is [1234:MATNUM].`
- NER as Sequence labelling:
  X: sequence of words
  Y: labels {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}

Rule-based

- Regex substitute:
  `matriculation (number)? (is)? (\d+)` → `[\3:mat-num]`

Rule-based

- Regex substitute:
  `matriculation (number)? (is)? (\d+)` → `[\3:mat-num]`
- Gets out of hand quickly:
  `(am|name (is)?) (.*?) (and|\s[.,?])?` → `[\3:person]`

Rule-based

- Regex substitute:
  `matriculation (number)? (is)? (\d+)` → `[\3:mat-num]`
- Gets out of hand quickly:
  `(am|name (is)?) (.*?) (and|\s[.,?])?` → `[\3:person]`
- No automated learning

Generative vs. Discriminative Models

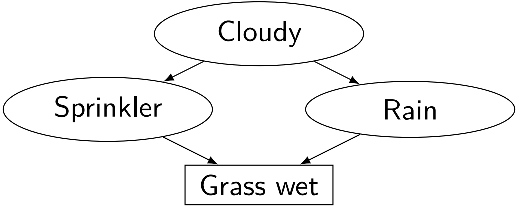
- Generative: Model actual distribution of data, learn joint probability and predict conditional probability using Bayes Theorem i.e. predict $P(Y|X)$ using $P(X|Y)$ and $P(Y)$
  e.g. Naive Bayes, HMMs
- Discriminative: Model decision boundary between classes, learn conditional probability directly, estimate parameters for $P(Y|X)$ directly from data
  e.g. MaxEnt Classifier, CRFs

Bayesian Network

- Directed acyclic graph (DAG), $(x \to y) \in E : y$ dependent on x

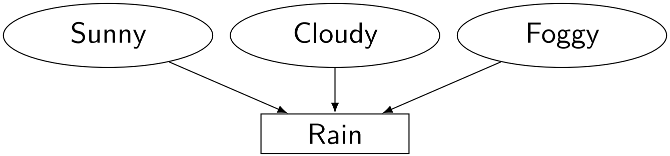
Local Markov Property

- Node is conditionally independent of its nondescendants given its parents.
  $p(\text{Sprinkler}|\text{Cloudy}, \text{Rain}) = p(\text{Sprinkler}|\text{Cloudy})$
- How does this benefit us?

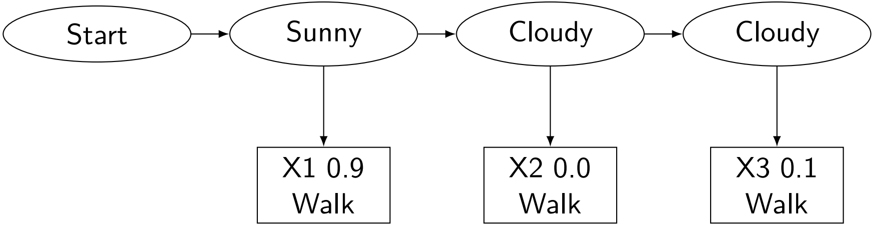


Naïve Bayes

- Assume absolute independence except for the one observed variable
- $p(y = \text{Yes}|x) = p(y_j|x) = \frac{p(x|y_j)p(y_j)}{p(x)} \propto p(x|y_j)p(y_j) \approx p(y_j)\prod_i p(x_i|y_j)$

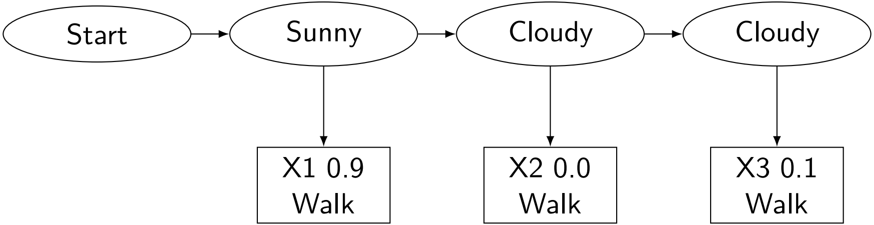


HMM



Sketch of HMM structure
observed variable *Walk duration*, latent variable: $Weather \in \{Sunny, Cloudy\}$

HMM



Sketch of HMM structure
observed variable *Walk duration*, latent variable: $\textit{Weather} \in \{\textit{Sunny}, \textit{Cloudy}\}$

$$p(y|x) = \prod_i p(y) \cdot o(y, x_i) \text{ (Naïve Bayes)}$$

$$\Rightarrow$$

$$p(\bar{y}|x) = \prod_i a(y_{i-1}, y_i) \cdot o(y_i, x_i) \text{ (HMM)}$$

HMM

- Hidden states: `{MATNUM, PERSON, LOCATION, NONE}`

HMM

- Hidden states: {MATNUM, PERSON, LOCATION, NONE}
- Better hidden states: {MATNUM, START+PERSON, INTERNAL+PERSON, END+PERSON, LOCATION, NONE, ...}

HMM

- Hidden states: `{MATNUM, PERSON, LOCATION, NONE}`
- Better hidden states: `{MATNUM, START+PERSON, INTERNAL+PERSON, END+PERSON, LOCATION, NONE, ...}`
- Transitions: MLE from annotated data

HMM

- Hidden states: {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Better hidden states: {`MATNUM`, `START+PERSON`, `INTERNAL+PERSON`, `END+PERSON`, `LOCATION`, `NONE`, ...}
- Transitions: MLE from annotated data
- Emission probabilities: MLE from annotated data (+ smoothing)

HMM

- Hidden states: {MATNUM, PERSON, LOCATION, NONE}
- Better hidden states: {MATNUM, START+PERSON, INTERNAL+PERSON, END+PERSON, LOCATION, NONE, ...}
- Transitions: MLE from annotated data
- Emission probabilities: MLE from annotated data (+ smoothing)
- $p(x, y) = \prod_i a(y_{i-1}, y_i) \cdot o(y_i, x_i)$

HMM

- Hidden states: {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Better hidden states: {`MATNUM`, `START+PERSON`, `INTERNAL+PERSON`, `END+PERSON`, `LOCATION`, `NONE`, `...`}
- Transitions: MLE from annotated data
- Emission probabilities: MLE from annotated data (+ smoothing)
- $p(x, y) = \prod_i a(y_{i-1}, y_i) \cdot o(y_i, x_i)$

HMM

- Hidden states: {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Better hidden states: {`MATNUM`, `START+PERSON`, `INTERNAL+PERSON`, `END+PERSON`, `LOCATION`, `NONE`, `...`}
- Transitions: MLE from annotated data
- Emission probabilities: MLE from annotated data (+ smoothing)
- $p(x, y) = \prod_i a(y_{i-1}, y_i) \cdot o(y_i, x_i)$

Questions

- What are the drawbacks of HMMs?

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {MATNUM, PERSON, LOCATION, NONE}

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {`MATNUM, PERSON, LOCATION, NONE`}
- Observed sequence of variables: x (words)

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {MATNUM, PERSON, LOCATION, NONE}
- Observed sequence of variables: x (words)
- Goal: Model $p(y|x)$ for all pairs (x, y)

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Observed sequence of variables: x (words)
- Goal: Model $p(y|x)$ for all pairs (x, y)

- $p(y|x) \propto \exp \{ \sum_i \log a(y_{i-1}, y_i) + \log o(y_i, x_i) \}$

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Observed sequence of variables: x (words)
- Goal: Model $p(y|x)$ for all pairs (x, y)

- $p(y|x) \propto \exp\left\{ \sum_i \log a(y_{i-1}, y_i) + \log o(y_i, x_i) \right\}$

- $p(y|x) = \frac{1}{Z(x)} \cdot \exp\left\{ \sum_i \log a(y_{i-1}, y_i) + \log o(y_i, x_i) \right\}$

Log-linear 1st Order Sequential Model

- Sequence of hidden states: y, {`MATNUM`, `PERSON`, `LOCATION`, `NONE`}
- Observed sequence of variables: x (words)
- Goal: Model $p(y|x)$ for all pairs (x, y)

- $p(y|x) \propto \exp\left\{ \sum_i \log a(y_{i-1}, y_i) + \log o(y_i, x_i) \right\}$

- $p(y|x) = \frac{1}{Z(x)} \cdot \exp\left\{ \sum_i \log a(y_{i-1}, y_i) + \log o(y_i, x_i) \right\}$

- $p(y|x) = \frac{1}{Z(x)} \cdot \prod_i \exp\{a(y_{i-1}, y_i) o(y_i, x_i)\}$

Log-linear 1st Order Sequential Model

- Replace $o(y_j, x_t)$ with $\lambda_1 h_1(y_j, x_t) + \lambda_2 h_2(y_j, x_t) + \ldots$

Log-linear 1st Order Sequential Model

- Replace $o(y_j, x_t)$ with $\lambda_1 h_1(y_j, x_t) + \lambda_2 h_2(y_j, x_t) + \ldots$
- Same with $a(y_j, y_i) = \lambda'_1 g_1(y_j, y_i) + \lambda'_2 g_2(y_j, y_i) + \ldots$

Log-linear 1st Order Sequential Model

- Replace $o(y_j, x_t)$ with $\lambda_1 h_1(y_j, x_t) + \lambda_2 h_2(y_j, x_t) + \ldots$
- Same with $a(y_j, y_i) = \lambda'_1 g_1(y_j, y_i) + \lambda'_2 g_2(y_j, y_i) + \ldots$
- Why not just $\sum_{\text{feature } f} \lambda_i f_i(y_i, y_j, x_t)$?

Model overview

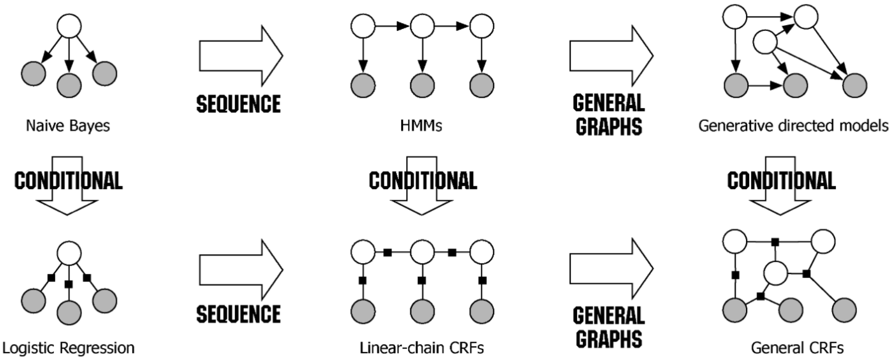


Figure 1: CRF in relation to other models; Source [2]

HMM → Linear CRF

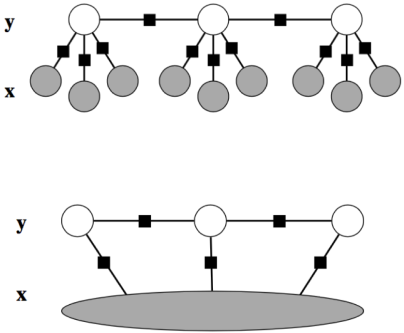


Figure 2: HMM vs. Linear Chain CRF; Source [12]

Question

- What is the difference between HMM and CRF?

Conditional Random Fields

- Factorization to maximal cliques.
- Allow access to a whole clique

Clique

$G = (V, E) \quad C \subseteq V : \forall x, y \in C : (x, y) \in E$

Maximal Clique

$C \subseteq C' \Rightarrow C = C'$

CRF

$p(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \Psi_c(x_c)$

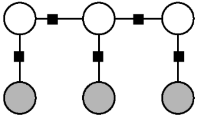


Figure 3: Linear CRF [2]

Linear CRF

- Sequence of hidden states: y, {`MATNUM, PERSON, LOCATION, NONE`}
- Observed sequence of variables: x (words)

- $p(y|x) \propto \prod_i \exp\left\{ \sum_j \lambda_j f_j(y_{i-1}, y_i, x, i)\right\}$

- $p(y|x) = \frac{1}{Z(x)} \prod_i \exp\left\{ \sum_j \lambda_j f_j(y_{i-1}, y_i, x, i)\right\}$

- Features: $f_j(y_{i-1}, y_i, x, i)$
- Parameters: λ
- Clique template: $\{\Psi_i(y_{i-1}, y_i, x, i) | \forall i \in \{1...n\}\}$

Linear CRF - Binary Features

$$f_j(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{if } \text{cond}_f(y_{i-1}, y_i, x, i) \\ 0 & \text{else} \end{cases}$$

Linear CRF - Binary Features

$$f_j(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{if } \text{cond}_f(y_{i-1}, y_i, x, i) \\ 0 & \text{else} \end{cases}$$

$$f_1(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{if } x_{i-2} \text{ is capitalized} \\ 0 & \text{else} \end{cases}$$

$$f_a(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{if } y_{i-1} = \texttt{number} \wedge y_t = \texttt{none} \\ 0 & \text{else} \end{cases}$$

$$\lambda_a = a(\text{number}, \texttt{none})$$

$$f_o(y_{i-1}, y_i, x, i) = \begin{cases} 1 & \text{if } y_i = \texttt{number} \wedge x_i = \texttt{<num>} \\ 0 & \text{else} \end{cases}$$

$$\lambda_o = o(\text{number}, \texttt{<num>})$$

Linear Chain CRF - Non-binary Features

$f_w(y_{i-1}, y_i, x, i) = |x_i|$ word length

$f_s(y_{i-1}, y_i, x, i) = |c|$ number of non-alphabetic characters

Linear Chain CRF - Non-binary Features

$$f_w(y_{i-1}, y_i, x, i) = |x_i| \text{ word length}$$
$$f_s(y_{i-1}, y_i, x, i) = |c| \text{ number of non-alphabetic characters}$$

Questions

- How do we interpret the values of λ_j for the features f_j? ($\lambda_j > 0$, $\lambda_j = 0$, $\lambda_j < 0$?)
- How are λs estimated?
- How many such features can we create?

CRF - Operations

Training:

$$argmax_{\lambda}\ p(y_D|x_D, \lambda)$$

Interpretation: Given label sequences and inputs, find parameters of the CRF M that maximise $p(y|x, \lambda)$.
Done using gradient methods, Forward-Backward algorithm etc.

CRF - Operations

Training:

$$argmax_{\lambda}\ p(y_D|x_D, \lambda)$$

Interpretation: Given label sequences and inputs, find parameters of the CRF M that maximise $p(y|x, \lambda)$.
Done using gradient methods, Forward-Backward algorithm etc.

Inference:

$$argmax_{y}\ p(y|x, \lambda)$$

Decoding:

$$max\ p(y|x, \lambda)$$

Interpretation: Given input x and CRF M, find optimal y.
Done using Viterbi algorithm.

Feature selection:

Alternative 1

1. Start with all features.
2. a. If there exists a feature removing which worsens the performance by $< t$, remove it. Repeat 2.
3. b. If not, exit.

Feature selection:

Alternative 1

1. Start with all features.
2. a. If there exists a feature removing which worsens the performance by $< t$, remove it. Repeat 2.
3. b. If not, exit.

Alternative 2

1. Start with no features.
2. a. If there exists a feature adding which improves the performance by $> t$, add it. Repeat 2.
3. b. If not, exit.

Feature selection:

Alternative 1

1. Start with all features.
2. a. If there exists a feature removing which worsens the performance by $< t$, remove it. Repeat 2.
3. b. If not, exit.

Alternative 2

1. Start with no features.
2. a. If there exists a feature adding which improves the performance by $> t$, add it. Repeat 2.
3. b. If not, exit.

Properties

- Hard to setup & train
- Fast inference

Linear Chain CRF - Regularization

Objective function:

$\mathcal{L} = \sum_s \log p(y^{(s)}|x^{(s)}, \lambda)$

LASSO:

$\mathcal{L}_{+lasso} = \sum_s \log p(y^{(s)}|x^{(s)}, \lambda) - \lambda_1 \sum_i |\lambda_i|$

Ridge:

$\mathcal{L}_{+ridge} = \sum_s \log p(y^{(s)}|x^{(s)}, \lambda) - \frac{\lambda_2}{2} \sum_i \lambda_i^2$

Elastic net:

$\mathcal{L}_{+elastic} = \sum_s \log p(y^{(s)}|x^{(s)}\lambda) - \frac{\lambda_2}{2} \sum_i \lambda_i^2 - \lambda_1 \sum_i |\lambda_i|$

Code

```
from sklearn_crfsuite import CRF
X_train = [
    [word2features(s, i) for i in range(len(s))]
    for s in train_sents]
y_train = [
    [label for token, postag, label in s]
    for s in train_sents]
crf = sklearn_crfsuite.CRF(
    algorithm='lbfgs',
    c1=0.1, c2=0.1,
    max_iterations=100,
)
crf.fit(X_train, y_train)
```

- Fast Linear Chain CRFs (C): http://www.chokkan.org/software/crfsuite/
- Fast Linear Chain CRFs (C++): https://taku910.github.io/crfpp/

Resources

1. Hidden Markov Model: https://web.stanford.edu/~jurafsky/slp3/A.pdf
2. Bayesian Networks: https://www.ics.uci.edu/~rickl/courses/cs-171/0-ihler-2016-fq/Lectures/Ihler-final/09b-BayesNet.pdf
3. Overview: https://www.analyticsvidhya.com/blog/2018/08/nlp-guide-conditional-random-fields-text-classification
4. Very detailed: http://homepages.inf.ed.ac.uk/csutton/publications/crftut-fnt.pdf
5. Academic-level introduction to CRF: https://www.youtube.com/watch?v=7L0MKKfqe98
6. Generalized CRF: https://people.cs.umass.edu/~wallach/technical_reports/wallach04conditional.pdf
7. Accessible introduction: http://pages.cs.wisc.edu/~jerryzhu/cs769/CRF.pdf
8. Forward-backward for CRF: https://www.cs.cornell.edu/courses/cs5740/2016sp/resources/collins_fb.pdf

Resources

9. NER using CRF: https://medium.com/data-science-in-your-pocket/named-entity-recognition-ner-using-conditional-random-fields-in-nlp-3660df22e95c
10. Python code: https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html#let-s-use-conll-2002-data-to-build-a-ner-system
11. Naïve Bayes, HMM, CRF: http://cnyah.com/2017/08/26/from-naive-bayes-to-linear-chain-CRF/
12. Highly Informative Naïve Bayes, HMM, MaxEnt, CRF: https://ls11-www.cs.tu-dortmund.de/_media/techreports/tr07-13.pdf