Model Estimation: Maximum Likelihood vs. Maximum Entropy

DMKM - Universitat Politècnica de Catalunya

Basics

- Random variable: Function on a stochastic process.
  $X : \Omega \longrightarrow \mathcal{R}$
- Continuous and discrete random variables.
- Probability mass (or density) function, Frequency function:
  $p(x) = P(X = x)$.
  Discrete R.V.: $\sum_x p(x) = 1$
  Continuous R.V: $\int_{-\infty}^{\infty} p(x)dx = 1$
- Distribution function: $F(x) = P(X \leq x)$
- Expectation and variance, standard deviation
  $E(X) = \mu = \sum_x xp(x)$
  $VAR(X) = \sigma^2 = E((X - E(X))^2) = \sum_x (x - \mu)^2 p(x)$

- Joint probability mass function: $p(x, y)$
- Marginal distribution:

$$p_X(x) = \sum_y p(x, y) \qquad p_{X|Y}(x \mid y) = \frac{p(x, y)}{p_Y(y)}$$
$$p_Y(y) = \sum_x p(x, y)$$

Simplified Polynesian. Sequences of C-V syllabes: Two random variables C,V

P(C,V)	p	t	k	
a	1/16	3/8	1/16	1/2
i	1/16	3/16	0	1/4
u	0	3/16	1/16	1/4
	1/8	3/4	1/8	

$P(p \mid i) = ?$
$P(a \mid t \vee k) = ?$
$P(a \vee i \mid p) = ?$

Samples and Estimators

- Random samples
- Sample variables:

  Sample mean: $\bar{\mu}_n = \frac{1}{n} \sum_{i=1} n x_i$

  Sample variance: $s_n^2 = \frac{1}{n-1} \sum_{i=1} n(x_i - \bar{\mu}_n)^2$.
- Law of Large Numbers: as n increases, $\bar{\mu}_n$ and s_n^2 converge to μ and σ^2
- Estimators: Sample variables used to estimate real parameters.

Maximum Likelihood Estimation (MLE)

- Choose the alternative that maximizes the probability of the observed outcome.
- $\bar{\mu}_n$ is a MLE for $E(X)$
- s_n^2 is a MLE for σ^2
- Data sparseness problem. Smoothing tecnhiques.

$P(a,b)$	dans	en	à	sur	au-cours-de	pendant	selon	
in	0.04	0.10	0.15	0	0.08	0.03	0	0.40
on	0.06	0.25	0.10	0.15	0	0	0.04	0.60
total	0.10	0.35	0.25	0.15	0.08	0.03	0.04	1.0

Maximum Entropy Estimation (MEE)

- Choose the alternative that maximizes the entropy of the obtained distribution, maintaning the observed probabilities.

Observations:
$p(en \vee \grave{a}) = 0.6$

$P(a,b)$	dans	en	à	sur	au-cours-de	pendant	selon	
in	0.04	0.15	0.15	0.04	0.04	0.04	0.04	
on	0.04	0.15	0.15	0.04	0.04	0.04	0.04	
total		$\underbrace{\qquad\qquad}_{0.6}$						1.0

Maximum Entropy Estimation (MEE)

- Choose the alternative that maximizes the entropy of the obtained distribution, maintaning the observed probabilities.

Observations:

$p(en \vee \text{à}) = 0.6; \qquad p((en \vee \text{à}) \wedge in) = 0.4$

$P(a, b)$	dans	en	à	sur	au-cours-de	pendant	selon	
in	0.04	**0.20**	**0.20**	0.04	0.04	0.04	0.04	
on	0.04	0.10	0.10	0.04	0.04	0.04	0.04	
total		$\underbrace{\qquad\qquad}_{0.6}$						1.0

Finding good estimators: MEE

Maximum Entropy Estimation (MEE)

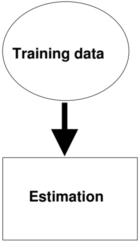
- Choose the alternative that maximizes the entropy of the obtained distribution, maintaning the observed probabilities.

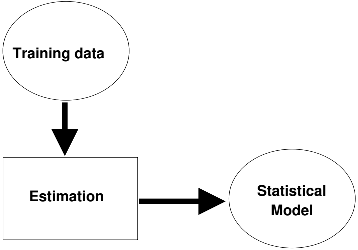
Observations:

$p(en \vee \grave{a}) = 0.6; \qquad p((en \vee \grave{a}) \wedge in) = 0.4; \qquad p(in) = 0.5$

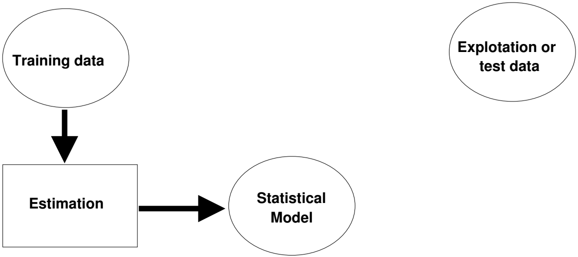
$P(a,b)$	dans	en	à	sur	au-cours-de	pendant	selon	
in	0.02	**0.20**	**0.20**	0.02	0.02	0.02	0.02	**0.5**
on	0.06	0.10	0.10	0.06	0.06	0.06	0.06	
total		$\underbrace{\qquad\qquad}_{\mathbf{0.6}}$						1.0

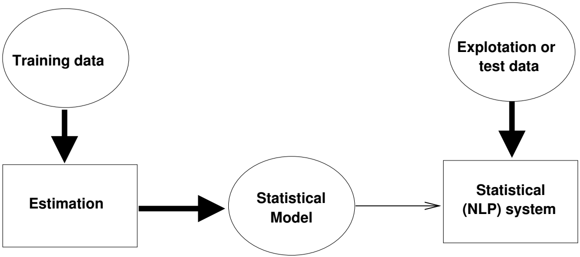
Statistical models for NLP


Training data
Estimation

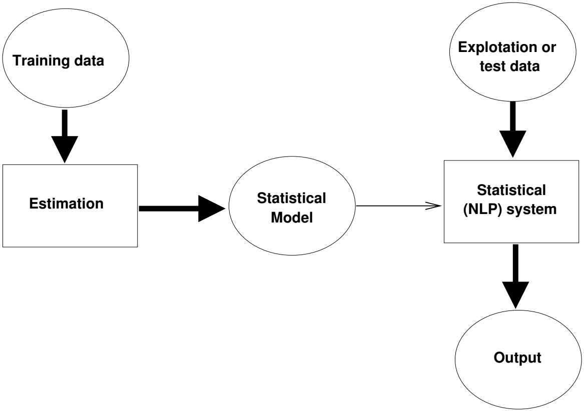

Training data
Estimation
Statistical Model

Statistical models for NLP




Training data
Explotation or test data
Estimation
Statistical Model
Statistical (NLP) system

Statistical models for NLP



- Prediction Models: Able to *predict* probabilities of future events, knowing past and present.
- Similarity Models: Able to compute *similarities* between objects (may predict, too).
  - Compare feature-vector/feature-set represented objects.
  - Compare distribution-vector represented objects
  - Used to group objects (clustering, data analysis, pattern discovery, ...)
  - If objects are "present and past" situations, computing similarities may be used as a prediction (memory-based ML techniques).

Example: Document representation

- Documents are represented as vectors in a high dimensional $\Re^N$ space.
- Dimensions are word forms, lemmas, NEs, ...
- Values may be either binary or real–valued (count, frequency, ...)

$$\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \qquad \vec{x}^T = [x_1 \dots x_N] \qquad |\vec{x}| = \sqrt{\sum_{i=1}^{N} x_i^2}$$

Prediction Models

Example: Noisy Channel Model (Shannon 48)

NLP Applications

Appl.	Input	Output	$p(i)$	$p(o \mid i)$
MT	L word sequence	M word sequence	$p(L)$	Translation model
OCR	Actual text	Text with mistakes	prob. of language text	model of OCR errors
PoS tagging	PoS tags sequence	word sequence	prob. of PoS sequence	$p(w \mid t)$
Speech recog.	word sequence	speech signal	prob. of word sequence	acoustic model

- Using data to infer information about distributions
  - Parametric / non-parametric estimation
  - Finding good estimators: MLE, MEE, ...
- Example: Language Modeling (Shannon game), N-gram models.
- Predictions based on past behaviour
  - Target / classification features $\rightarrow$ Independence assumptions
  - Equivalence classes (bins).
    Granularity: discrimination *vs.* statistical reliability

N-gram models

- Predicting the next word in a sequence, given the *history* or *context*. $P(w_n \mid w_1 \dots w_{n-1})$
- Markov assumption: Only *local* context (of size $n-1$) is taken into account. $P(w_i \mid w_{i-n+1} \dots w_{i-1})$
- bigrams, trigrams, four-grams ($n = 2, 3, 4$).
  Sue swallowed the large green <?>
- Parameter estimation (number of equivalence classes)
- Parameter reduction: stemming, semantic classes, PoS, ...

Model	Parameters
bigram	$20,000^2 = 4 \times 10^8$
trigram	$20,000^3 = 8 \times 10^{12}$
four-gram	$20,000^4 = 1.6 \times 10^{17}$

Language model sizes for a 20,000 words vocabulary

MLE Overview

Estimate the probability of the target feature based on observed data. The prediction task can be reduced to having good estimations of the *n*-gram distribution:

$$P(w_n \mid w_1 \ldots w_{n-1}) = \frac{P(w_1 \ldots w_n)}{P(w_1 \ldots w_{n-1})}$$

- **MLE (Maximum Likelihood Estimation)**

  $P_{MLE}(w_1 \ldots w_n) = \frac{C(w_1 \ldots w_n)}{N}$

  $P_{MLE}(w_n \mid w_1 \ldots w_{n-1}) = \frac{C(w_1 \ldots w_n)}{C(w_1 \ldots w_{n-1})}$

  - No probability mass for unseen events
  - Unsuitable for NLP
  - Data sparseness, Zipf's Law

Notation

- $C(w_1 \dots w_n)$: Observed occurrence count for n-gram $w_1 \dots w_n$.
- $C_A(w_1 \dots w_n)$: Observed occurrence count for n-gram $w_1 \dots w_n$ on data subset A.
- N: Number of observed n-gram occurrences

$$N = \sum_{w_1 \dots w_n} C(w_1 \dots w_n)$$

- N_k: Number of classes (n-grams) observed k times.
- N_k^A: Number of classes (n-grams) observed k times on data subset A.
- B: Number of equivalence classes or bins (number of potentially observable n-grams).

Smoothing 1 - Adding Counts

- **Laplace's Law** (adding one)

  $$P_{LAP}(w_1 \dots w_n) = \frac{C(w_1 \dots w_n) + 1}{N + B}$$

  - For large values of B too much probability mass is assigned to unseen events

- **Lidstone's Law**

  $$P_{LID}(w_1 \dots w_n) = \frac{C(w_1 \dots w_n) + \lambda}{N + B\lambda}$$

  - Usually $\lambda = 0.5$, *Expected Likelihood Estimation*.
  - Equivalent to linear interpolation between MLE and uniform prior, with $\mu = N/(N + B\lambda)$,

    $$P_{LID}(w_1 \dots w_n) = \mu \frac{C(w_1 \dots w_n)}{N} + (1 - \mu)\frac{1}{B}$$

Smoothing 2 - Discounting Counts

- **Absolute Discounting**

$$P_{ABS}(w_1 \ldots w_n) = \begin{cases} \frac{r-\delta}{N} & \textit{if } r > 0 \\ \frac{(B-N_0)\delta/N_0}{N} & \textit{otherwise} \end{cases}$$

- **Linear Discounting**

$$P_{LIN}(w_1 \ldots w_n) = \begin{cases} \frac{(1-\alpha)r}{N} & \textit{if } r > 0 \\ \frac{\alpha}{N_0} & \textit{otherwise} \end{cases}$$

- *Notation:* γ stands for $w_1 \dots w_n$.
- Divide the train corpus in two subsets, A and B.
- Define: $T_r^{AB} = \sum_{\gamma : C_A(\gamma)=r} C_B(\gamma)$
- **Held Out Estimator**

$$P_{HO}(w_1 \dots w_n) = \frac{T^{AB}_{C_A(\gamma)}}{N^A_{C_A(\gamma)}} \times \frac{1}{N}$$

- **Cross Validation** (deleted estimation)

$$P_{DEL}(w_1 \dots w_n) = \frac{T^{AB}_{C_A(\gamma)} + T^{BA}_{C_B(\gamma)}}{N^A_{C_A(\gamma)} + N^B_{C_B(\gamma)}} \times \frac{1}{N}$$

- **Cross Validation** (Leave-one-out)

Combining Estimators

- **Simple Linear Interpolation**

$$P_{LI}(w_n \mid w_{n-2}, w_{n-1}) = \\ = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n \mid w_{n-1}) + \lambda_3 P_3(w_n \mid w_{n-2}, w_{n-1})$$

- **General Linear Interpolation**

$$P_{LI}(w_n \mid h) = \sum_{i=1}^{k} \lambda_i(h) P_i(w \mid h_i)$$

- **Katz's Backing-off**

$$P_{BO}(w_i \mid w_{i-n+1} \dots w_{i-1}) = \begin{cases} (1 - d_{w_{i-n+1}\dots w_{i-1}}) \dfrac{C(w_{i-n+1} \dots w_i)}{C(w_{i-n+1} \dots w_{i-1})} & \text{if } C(w_{i-n+1} \dots w_i) > k \\ \alpha_{w_{i-n+1}\dots w_{i-1}} P_{BO}(w_i \mid w_{i-n+2} \dots w_{i-1}) & \text{otherwise} \end{cases}$$

MEM Overview

- Maximum Entropy: alternative estimation technique.
- Able to deal with different kinds of evidence
- ME principle:
  - Do not assume anything about non-observed events.
  - Find the most uniform (maximum entropy, less informed) probability distribution that matches the observations.
- Example:

$p(a,b)$	0	1	
x	?	?	
y	?	?	
total	0.6		1.0

Observations

$p(a,b)$	0	1	
x	0.5	0.1	
y	0.1	0.3	
total	0.6		1.0

One possible $p(a,b)$

$p(a,b)$	0	1	
x	0.3	0.2	
y	0.3	0.2	
total	0.6		1.0

Max.Entropy $p(a,b)$

ME Modeling

- Observed facts are constraints for the desired model p.
- Constraints take the form of feature functions:

$$f_i : \varepsilon \rightarrow \{0,1\}$$

- The desired model must satisfy the constraints:

$$E_p(f_i) = E_{\tilde{p}}(f_i) \quad \forall i$$

where:

$E_p(f_i) = \sum_{x \in \varepsilon} p(x) f_i(x)$ expectation of model p.

$E_{\tilde{p}}(f_i) = \sum_{x \in \varepsilon} \tilde{p}(x) f_i(x)$ observed expectation.

- Example:

$$\varepsilon = \{x, y\} \times \{0, 1\}$$

$p(a,b)$	0	1	
x	?	?	
y	?	?	
total	0.6		1.0

- Observed fact: $p(x,0) + p(y,0) = 0.6$
- Encoded as a constraint: $E_p(f_1) = 0.6$
  where:
  - $f_1(a,b) = \begin{cases} 1 & \textit{if } b = 0 \\ 0 & \textit{otherwise} \end{cases}$
  - $E_p(f_1) = \sum_{(a,b)\in\{x,y\}\times\{0,1\}} p(a,b) f_1(a,b)$

Probability Model

- There is an infinite set P of probability models consistent with observations:

$$P = \{p \mid E_p(f_i) = E_{\tilde{p}}(f_i),\ \forall i = 1 \dots k\}$$

- Maximum entropy model

$$p^* = \operatorname*{argmax}_{p \in P} H(p)$$

$$H(p) = -\sum_{x \in \varepsilon} p(x) \log p(x)$$

- For NLP applications, we are usually interested in conditional distributions $P(A|B)$, thus:

$$E_{\widetilde{p}}(f_j) = \sum_{a,b} \widetilde{p}(a,b) f_j(a,b)$$

$$E_p(f_j) = \sum_{a,b} \widetilde{p}(b) p(a \mid b) f_j(a,b)$$

- Maximum entropy model

$$p^* = \underset{p \in P}{\operatorname{argmax}}\, H(p)$$

$$H(p) = H(A \mid B) = -\sum_{a,b} \widetilde{p}(b) p(a \mid b) \log p(a \mid b)$$

Parameter Estimation

Example: Maximum entropy model for translating *in* to French

- No constraints

$P(x)$	dans	en	à	au-cours-de	pendant	
	0.2	0.2	0.2	0.2	0.2	
total						1.0

- With constraint $p(dans) + p(en) = 0.3$

$P(x)$	dans	en	à	au-cours-de	pendant	
	0.15	0.15	0.233	0.233	0.233	
total	**0.3**					1.0

- With constraints $p(dans) + p(en) = 0.3$; $p(en) + p(à) = 0.5$

...Not so easy !

Parameter estimation

- Exponential models. (Lagrange multipliers optimization)
  $p(a \mid b) = \frac{1}{Z(b)} \prod_{j=1}^{k} \alpha_j^{f_j(a,b)} \qquad \alpha_j > 0$
  $Z(b) = \sum_a \prod_{i=1}^{k} \alpha_i^{f_i(a,b)}$
- also formuled as
  $p(a \mid b) = \frac{1}{Z(b)} \exp(\sum_{j=1}^{k} \lambda_j f_j(a,b))$
  $\lambda_i = \ln \alpha_i$
- Each model parameter weights the influence of a feature.
- Optimal parameters (ME model) can be computed with:
  - GIS. Generalized Iterative Scaling(Darroch & Ratcliff 72)
  - IIS. Improved Iterative Scaling (Della Pietra et al. 96)
  - LM-BFGS. Limited Memory BFGS (Malouf 03)

Input: Feature functions $f_1 \ldots f_n$, empirical distribution $\widetilde{p}(a, b)$
Output: λ_i^* parameters for optimal model p^*

Start with $\lambda_i = 0$ for all $i \in \{1 \ldots n\}$
Repeat
  For each $i \in \{1 \ldots n\}$ **do**
    let $\Delta\lambda_i$ be the solution to

$$\sum_{a,b} \widetilde{p}(b) p(a \mid b) f_i(a, b) \exp(\Delta\lambda_i \sum_{j=1}^{n} f_j(a, b)) = \widetilde{p}(f_i)$$

    $\lambda_i \leftarrow \lambda_i + \Delta\lambda_i$
  end for
Until all λ_i have converged

Application to NLP Tasks

- Speech processing (Rosenfeld 94)
- Machine Translation (Brown et al 90)
- Morphology (Della Pietra et al. 95)
- Clause boundary detection (Reynar & Ratnaparkhi 97)
- PP-attachment (Ratnaparkhi et al 94)
- PoS Tagging (Ratnaparkhi 96, Black et al 99)
- Partial Parsing (Skut & Brants 98)
- Full Parsing (Ratnaparkhi 97, Ratnaparkhi 99)
- Text Categorization (Nigam et al 99)

- Probabilistic model over $H \times T$

$$h_i = (w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2})$$

$$f_j(h_i, t) = \begin{cases} 1 & \textit{if suffix}(w_i) = \texttt{ing} \wedge t = \texttt{VBG} \\ 0 & \textit{otherwise} \end{cases}$$

- Compute $p^*(h, t)$ using GIS
- Disambiguation algorithm: *beam search*

$$p(t \mid h) = \frac{p(h, t)}{\sum_{t' \in T} p(h, t')}$$

$$p(t_1 \dots t_n \mid w_1 \dots w_n) = \prod_{i=1}^{n} p(t_i \mid h_i)$$

- Probabilistic model over $W \times C$

$$d = (w_1, w_2 \ldots w_N)$$

$$f_{w,c'}(d,c) = \begin{cases} \frac{N(d,w)}{N(d)} & if\ c = c' \\ 0 & otherwise \end{cases}$$

- Compute $p^*(c \mid d)$ using IIS
- Disambiguation algorithm: Select class with highest

$$P(c \mid d) = \frac{1}{Z(d)} exp(\sum_i \lambda_i f_i(d,c))$$

MEM Summary

- Advantages
  - Teoretically well founded
  - Enables combination of random context features
  - Better probabilistic models than MLE (no smoothing needed)
  - General approach (features, events and classes)
- Disadvantages
  - Implicit probabilistic model (joint or conditional probability distribution obtained from model parameters).
  - High computational cost of GIS and IIS.
  - Overfitting in some cases.

References

- T. Cover & J. Thomas, **Elements of Information Theory**. John Wiley & Sons, 1991.
- C. Manning & H. Schütze, **Foundations of Statistical Natural Language Processing**. The MIT Press. Cambridge, MA. May 1999.
- D. Jurafsky & J.H. Martin. **Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics**, 2nd edition. Prentice-Hall, 2009.
- A. Berger, S.A. Della Pietra & V.J. Della Pietra, **A Maximum Entropy Approach to Natural Language Processing**. Computational Linguistics, 22(1):39-71, 1996.
- R Malouf, **A comparison of algorithms for maximum entropy parameter estimation**. In Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002), Pages 49-55, 2002.
- A. Ratnaparkhi, **Maximum Entropy Models for Natural Language Ambiguity Resolution**. Ph.D Thesis. University of Pennsylvania, 1998.