Text Mining, Information and Fact Extraction
Part 3: Machine Learning Techniques (continued)

Marie-Francine Moens
Department of Computer Science
Katholieke Universiteit Leuven, Belgium
sien.moens@cs.kuleuven.be

Problem definition

- Much of our communication is in the form of natural language text:
  - When processing text, many variables are interdependent (often dependent on previous content in the discourse):
    - e.g., the named entity labels of neighboring words are dependent: *New York* is a location, *New York Times* is an organization

Problem definition

- Our statements have some **structure**
  - Sequences
  - Hierarchical
  - ...

- A certain combination of statements often conveys a certain meaning

Problem definition

- **Fact extraction** from text could benefit from **modeling context**:
  - at least at the **sentence level**

- But text mining should move beyond fact extraction towards **concept extraction**, and while integrating **discourse context**

- **Could result in a fruitful blending of text mining and natural language understanding**

Overview

- Dealing with **sequences**:
  - Hidden Markov model
- Dealing with **undirected graphical network**:
  - Conditional random field
- Dealing with **directed graphical networks**:
  - Probabilistic Latent Semantic Analysis
  - Latent Dirichlet Allocation

+ **promising research directions**

Context-dependent classification

- The class to which a feature vector is assigned depends on:
  1) the feature vector itself
  2) the values of other feature vectors
  3) the existing relation among the various classes

- Examples:
  - hidden Markov model
  - conditional random field

Hidden Markov model

- = is a probabilistic finite state automaton to model the probabities of a linear sequence of events
- The task is to assign:

  a **class sequence** $Y = (y_1,\ldots,y_T)$ to the **sequence of observations** $X = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T)$

Markov model

- The model of the content is implemented as a Markov chain of states
- The model is defined by:
  - a set of states
  - a set of transitions between states and the probabilities of the transitions (probabilities of the transitions that go out from each state sum to one)
  - a set of output symbols å that can be emitted when in a state (or transition) and the probabilities of the emissions

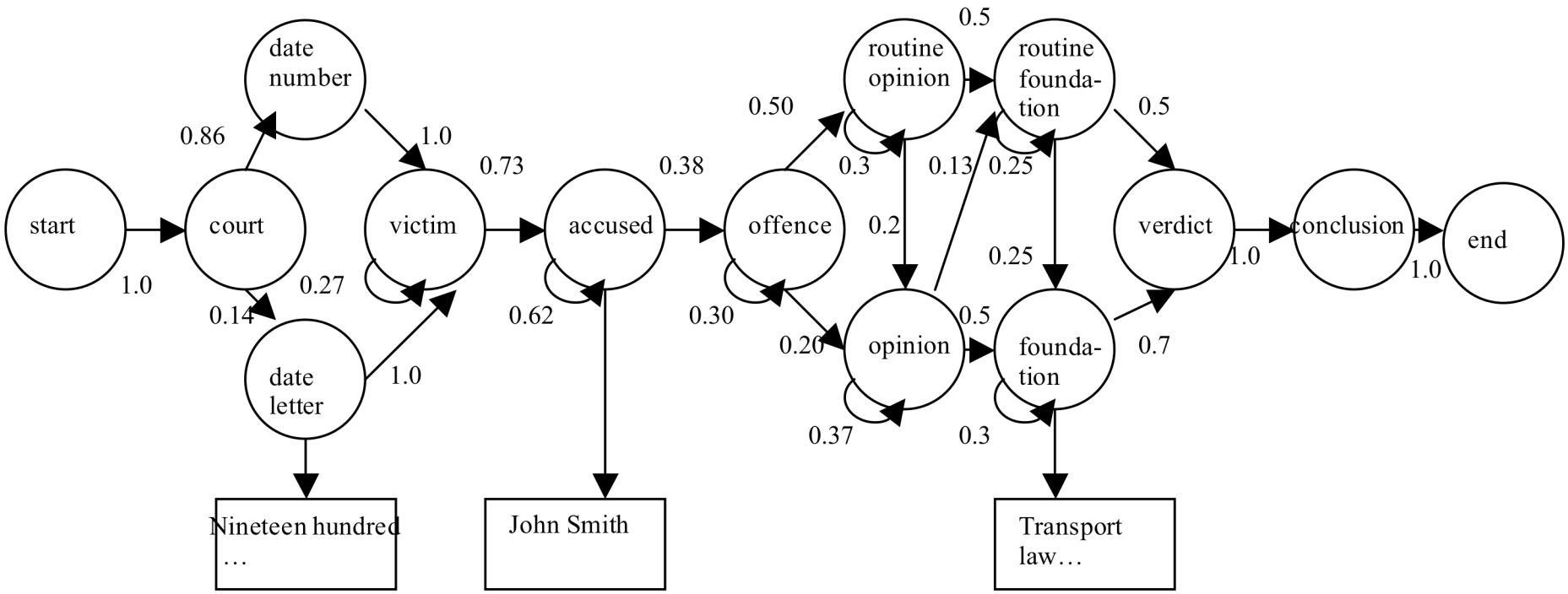


Figure 5.3. An example Markov model that represents a Belgian criminal court decision. Some examples of emissions are shown without their probabilities.

The probability of a sequence of states or classes $Y = (y_1, \ldots, y_T)$ is easily calculated for a **Markov chain** :

$$P(y_1,\ldots,y_T) = P(y_1)P(y_2|y_1)\,P(y_3|y_1,y_2)\,\ldots\,\mathrm{P}\,(y_T|\,y_1,\ldots,y_{T-1})$$

A **first order** Markov model assumes that class dependence is limited only within two successive classes yielding:

$$P(y_1,\ldots,y_T) = P(y_1)P(y_2|y_1)\,P(y_3|y_2)\ldots P\,(y_T|\,y_{T-1})$$

$$= P(y_1)\prod_{i=2}^{T} P(y_i|y_{i-1})$$

The models that we consider in the context of information extraction have a discrete output, i.e., an observation outputs discrete values.

Markov model

So, using the first-order Markov model in the above example gives:

$P(start, court, date\ number, victim) = 0.86$

When a sequence can be produced by several paths: sum of path probabilities is taken.

Markov model

- **Visible Markov model**:
  - we can identify the path that was taken inside the model to produce each training sequence: i.e., we can directly observe the states and the emitted symbols
- **Hidden Markov model**:
  - you do not know the state sequence that the model passed through when generating the training examples, i.e., the states of the training examples are not fully observable

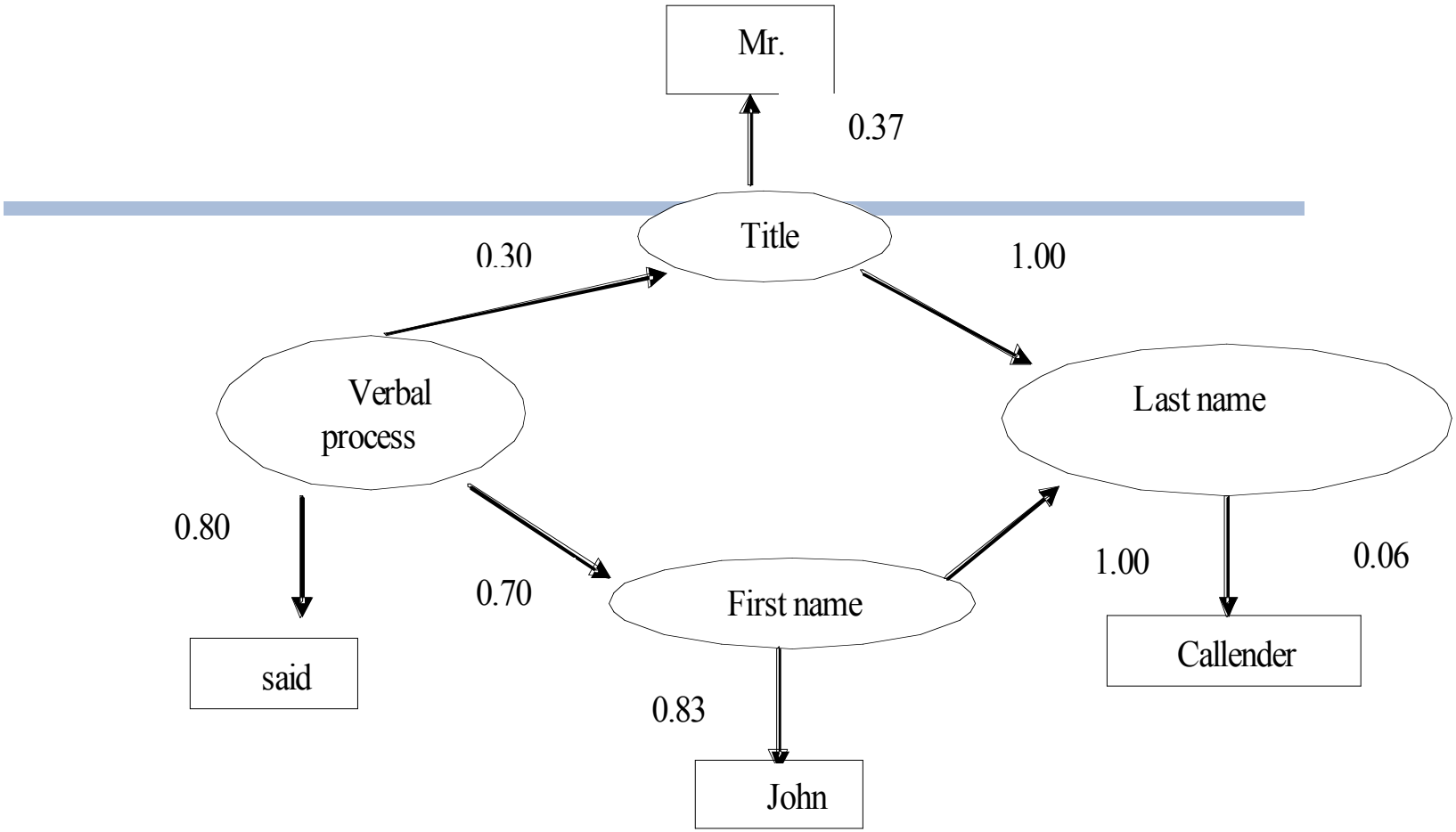


Fig. 5.4. Example of a visible Markov Model for a named entity recognition task.

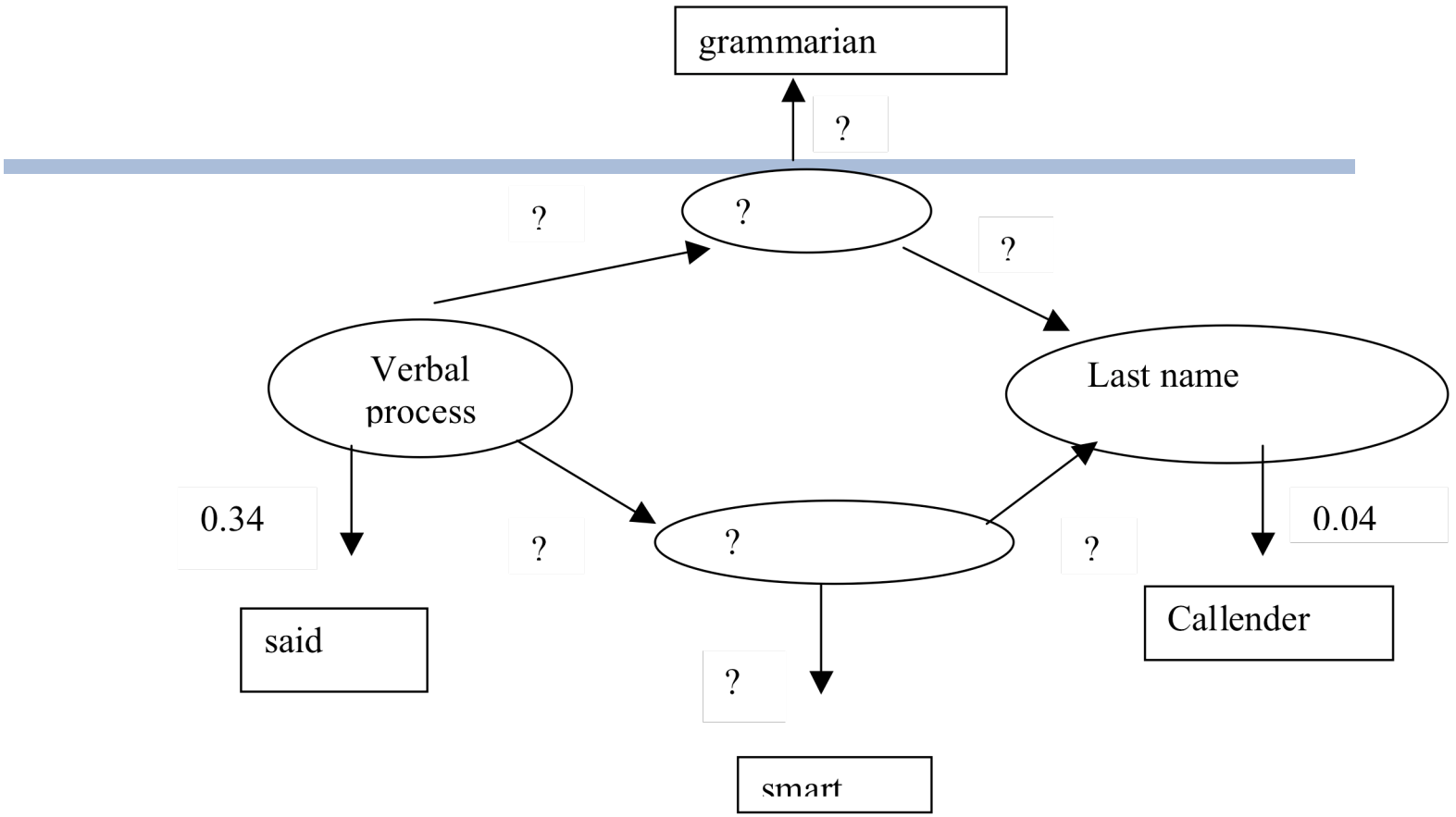


Fig. 5.5. Example of a hidden Markov model for a named entity recognition task.

Markov model: training

- The task is learning the probabilities of the initial state, the state transitions and of the emissions of the model μ

Visible Markov model: training

The labeling is used to directly compute the probabilities of the parameters of the Markov model by means of maximum likelihood estimates in the training set X_{all}. The transition probabilities $P(y'|y)$ and the emission probabilities $P(\boldsymbol{x}|y)$ are based on the counts of respectively the class transitions $\xi(y->y')$ or $\xi(y,y')$ and of the emissions occurring in a class $\gamma(y)$ where $y \uparrow \boldsymbol{x}_i$ considered at the different times t:

$$P(y'|y) = \frac{\sum_{t=1}^{T-1} \xi_t(y,y')}{\sum_{t=1}^{T-1} \gamma_t(y)}$$

$$P(\boldsymbol{x}|y) = \frac{\sum_{t=1 \text{ and } y \uparrow \boldsymbol{x}}^{T} \gamma_t(y)}{\sum_{t=1}^{T} \gamma_t(y)}$$

Hidden Markov model: training

- The Baum-Welch approach:
  1. Start with initial estimates for the probabilities chosen randomly or according to some prior knowledge.
  2. Apply the model on the training data:
     - **Expectation step (E)**: Use the current model and observations to calculate the expected number of traversals across each arc and the expected number of traversals across each arc while producing a given output.
     - **Maximization step (M)**: Use these calculations to update the model into a model that most likely produces these ratios.
  3. Iterate step 2 until a convergence criterion is satisfied (e.g., when the differences of the values with the values of a previous step are smaller than a threshold value ε).

Hidden Markov model: training

Expectation step (E)

We consider the number of times that a path passes through state y at time t and through state y' at the next time $t + 1$ and the number of times this state transition occurs while generating the training sequences X_{all} given the parameters of the current model μ. We then can define:

$$\xi_t(y,y') \equiv \xi_t(y,y'|X_{all},\mu) = \frac{\xi_t(y,y',X_{all}|\mu)}{P(X_{all}|\mu)}$$

$$= \frac{\alpha(y_t = y)P(y'|y)P(\mathbf{x}_{t+1}|y')\beta(y_{t+1} = y')}{P(X_{all}|\mu)}$$

where $\alpha(y_t = y)$ accounts for the path history terminating at time t and state y (i.e., the probability of being at state y at time t and outputting the first t symbols) and $\beta(y_{t+1} = y')$ accounts for the future of the path, which at time t+1 is at state y' and then evolves unconstrained until the end (i.e., the probability of being at the remaining states and outputting the remaining symbols).

Hidden Markov model: training

We define also the probability of being at time t at state y:

$$\gamma_t(y) \equiv \gamma_t(y|X_{all},\mu) = \frac{\alpha(y_t = y)\beta(y_t = y)}{P(X_{all}|\mu)}$$

$\sum_{t=1}^{T-1}\gamma_t(y)$ can be regarded as the expected number of transitions from state y given the model μ and the observation sequences X_{all}.

$\sum_{t=1}^{T-1}\xi_t(y,y')$ can be regarded as the expected number of transitions from state y to state y', given the model μ and the observation sequences X_{all}.

Hidden Markov model: training

Maximization step (M)

During the M-step the following formulas compute reasonable estimates of the unknown model parameters:

$$\overline{P}(y'|y) = \frac{\sum_{t=1}^{T-1} \xi_t(y, y')}{\sum_{t=1}^{T-1} \gamma_t(y)}$$

$$\overline{P}(\boldsymbol{x}|y) = \frac{\sum_{t=1 \text{ and } y \uparrow \boldsymbol{x}}^{T} \gamma_t(y)}{\sum_{t=1}^{T} \gamma_t(y)}$$

$$\overline{P}(y) = \gamma_1(y)$$

Hidden Markov Model

- The task is to assign a **class sequence** $Y = (y_1,\ldots,y_T)$ to the **observation sequence** $X = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T)$: how do we choose the class sequence that best explains the observation sequence?

$$P(Y|X) = P(y_1)P(\boldsymbol{x}_1|y_1)\prod_{i=2}^{T} P(y_i|y_{i-1})P(\boldsymbol{x}_i|y_i)$$

- Best path is computed with the **Viterbi algorithm**:
  - efficient algorithm for computing the optimal path
  - computed by storing the best extension of each possible path at time t

$$Y^* = \underset{Y}{argmax}\, P(Y|X)$$

Hidden Markov model

- Advantage:
  - useful for extracting information that is sequentially structured
- Disadvantage:
  - need for an a priori notion of the model topology, attempts to learn the model topology
  - large amounts of training data needed
  - two independence assumptions: a state depends only on its immediate preprocessor; each observation variable x_t depends only on the current state y_t
- Used for named entity recognition and other information extraction tasks, especially in the biomedical domain

Maximum Entropy Markov model

- MEMM = Markov model in which the transition distributions are given by a maximum entropy model

- Linear-chain CRF is an improvement of this model

Conditional random field

- Let $\boldsymbol{X}$ be a random variable over data sequences to be labeled and $\boldsymbol{Y}$ a random variable over corresponding label sequences
- All components $\boldsymbol{Y}_i$ of $\boldsymbol{Y}$ are assumed to range over a finite label alphabet $\sum$
- A conditional random field is viewed as an undirected graphical model or Markov random field, conditioned on $\boldsymbol{X}$
- We define $G = (V, E)$ to be an **undirected graph** such that there is a node $v \in V$ corresponding to each of the random variables representing an element Y_v of $\boldsymbol{Y}$
- If each random variable Y_v obeys the Markov property with respect to G, then the model $(\boldsymbol{Y},\boldsymbol{X})$ is a conditional random field

Conditional random field

- In theory the structure of graph G may be arbitrary, however, when modeling sequences, the simplest and most common graph structure encountered is that in which the nodes corresponding to elements of $\boldsymbol{Y}$ form a simple first-order Markov chain (linear-chain CRF)
- In an information extraction task, $\boldsymbol{X}$ might range over the sentences of a text, while $\boldsymbol{Y}$ ranges over the semantic classes to be recognized in these sentences

- Note: in the following $\boldsymbol{x}$ refers to an observation sequence and not to a feature vector and y to a labeling sequence

Conditional random field

- **Feature functions** depend on the current state or on the previous and current states

$$s_j(y_i, \boldsymbol{x}, i) = \begin{cases} 1 \text{ if the observation at position } i \text{ is the word "say"} \\ 0 \text{ otherwise} \end{cases}$$

$$t_j(y_{i-1}, y_i, \boldsymbol{x}, i) = \begin{cases} 1 \text{ if } y_{i-1} = \text{"person" and } y_i = \text{"movement"} \\ 0 \text{ otherwise} \end{cases}$$

- We use a more global notation f_j for a feature function where $f_j(y_{i-1}, y_i, \boldsymbol{x}, i)$ is either a state function $s_j(y_i, \boldsymbol{x}, i) = s_j(y_{i-1}, y_i, \boldsymbol{x}, i)$ or a transition function $t_j(y_{i-1}, y_i, \boldsymbol{x}, i)$

Conditional random field

- Considering k feature functions, the conditional probability distribution defined by the CRF is:

$$p(y|\boldsymbol{x}) = \frac{1}{Z}\exp\left(\sum_{j=1}^{k}\sum_{i=1}^{T}\lambda_j f_j(y_{i-1}, y_i, \boldsymbol{x}, i)\right)$$

where λ_j = parameter adjusted to model the observed statistics

Z = normalizing constant

The most probable label sequence y^* for input sequence $\boldsymbol{x}$ is:

$$y^* = \underset{y}{argmax}\ p(y|\boldsymbol{x})$$

Conditional random field: training

- Like for the maximum entropy model, we need numerical methods in order to derive λ_j given the set of constraints
- The problem of efficiently calculating the expectation of each feature function with respect to the linear-chain CRF model distribution for every observation sequence $\boldsymbol{x}$ in the training data: dynamic programming techniques that are similar to the Baum-Welch algorithm (cf. HMM)
- In general CRFs we use approximate inference (e.g., Markov Chain Monte Carlo sampler)

Conditional random field

- Advantages:
  - Combines the possibility of dependent features, context-dependent classification and the maximum entropy principle
  - One of the current most successful information extraction techniques
- Disadvantage:
  - Training is computationally expensive, especially when the graphical structure is complex

F1 scores on the CoNLL Dataset						
Approach	LOC	ORG	MISC	PER	ALL	Relative Error reduction
Bunescu and Mooney (2004) (Relational Markov Networks)						
Only Local Templates	-	-	-	-	**80.09**	
Global and Local Templates	-	-	-	-	**82.30**	11.1%
Finkel et al. (2005)(Gibbs Sampling)						
Local+Viterbi	88.16	80.83	78.51	90.36	**85.51**	
Non Local+Gibbs	88.51	81.72	80.43	92.29	**86.86**	9.3%
Our Approach with the 2-stage CRF						
Baseline CRF	88.09	80.88	78.26	89.76	**85.29**	
+ Document token-majority features	89.17	80.15	78.73	91.60	86.50	
+ Document entity-majority features	89.50	81.98	79.38	91.74	86.75	
+ Document superentity-majority features	89.52	82.27	79.76	92.71	**87.15**	12.6%
+ Corpus token-majority features	89.48	82.36	79.59	92.65	87.13	
+ Corpus entity-majority features	89.72	82.40	79.71	92.65	87.23	
+ Corpus superentity-majority features (All features)	89.80	82.39	79.76	92.57	**87.24**	13.3%

Named entity recognition: 2-stage approach: 1) CRF with local features; 2) local information and output of first CRF as features. Comparison against competitive approaches. Baseline results are shown on the first line of each approach.

[Krishnan & Manning 2006]

Evaluation of the supervised learning methods

- **Results approach the results of using hand-crafted patterns**
- But, for some tasks the results fall short of human capability:
  - **both for the hand-crafted and learned patterns**
  - explanation:
    - high variation of natural language expressions that form the context of the information or that constitute the information
    - ambiguous patterns and lack of discriminative features
    - lack of world knowledge not made explicit in the text

Evaluation of the supervised learning methods

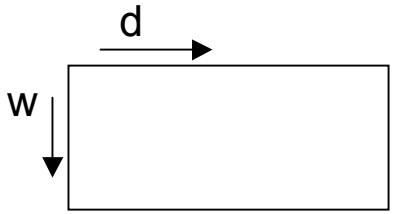
- Annotating: tedious task !
  - **integration** of existing **knowledge resources,** if conveniently available (e.g., use of dictionary of classified named entities when learning named entity classification patterns)
  - the learned patterns are best treated as **reusable knowledge components**
  - bootstrapping (**weakly supervised learning**)
    - given a limited set of patterns manually constructed or patterns learned from annotations
    - expand “seed patterns” with techniques of unsupervised learning and/or external knowledge resources

Less supervision?

Latent semantic topic models

- = a class of unsupervised (or semi-supervised) models in which the semantic properties of words and documents are expressed in terms of topics
  - models are also called **aspect models**
- Latent Semantic Indexing:
  - the semantic information can be derived from a word-document matrix

    [Deerweester et al. 1990]



  - But, LSI is unable to capture multiple senses of a word
- **Probabilistic topic models**

Panini

- Panini = Indian grammarian (6th-4thcentury B.C. ?) who wrote a grammar for **sanskrit**
- **Realizational chain** when creating natural language texts:
  - Ideas -> broad conceptual components of a text -> subideas -> sentences -> set of semantic roles-> set of grammatical and lexical concepts ->character sequences

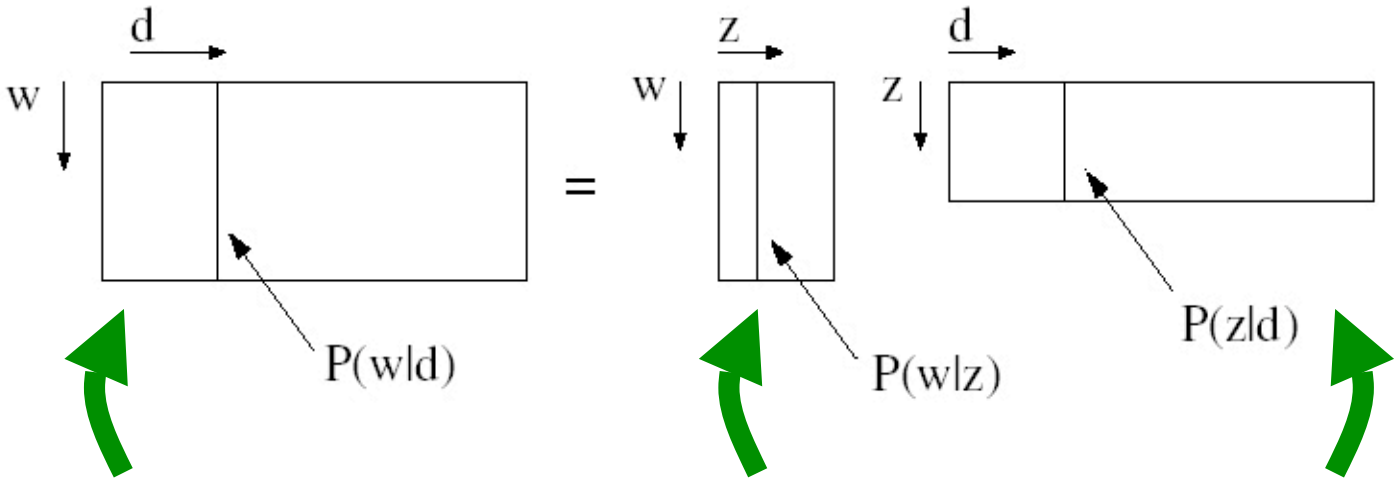
[Kiparsky 2002]

Probabilistic topic model

- = **Generative model** for documents: probabilistic model by which documents can be generated
  - document = probability distribution over topics
  - topic = probability distribution over words

- To make a new document, one chooses a distribution over topics, for each topic one draws words according to a certain distribution:
  - select a document d_j with probability $P(d_j)$
  - pick a **latent class z_k** with probability $P(z_k|d_j)$
  - generate a word w_i with probability $P(w_i|z_k)$

[Steyers & Griffiths 2007]

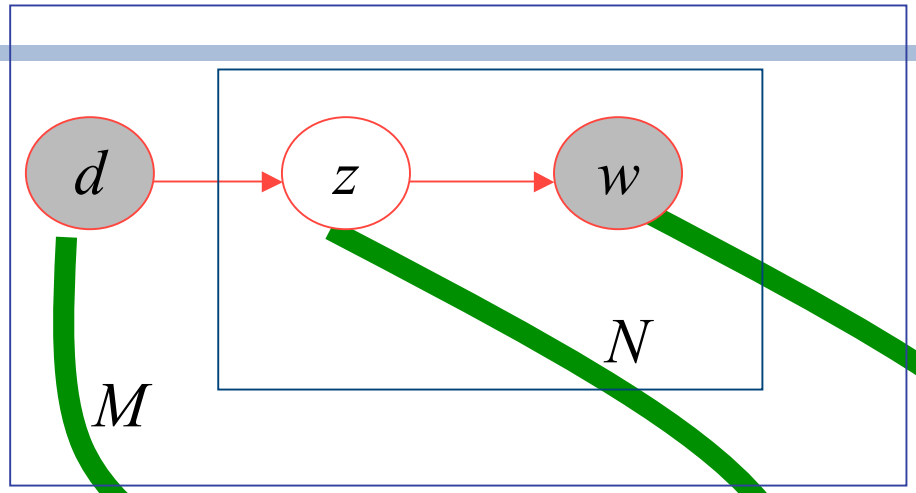


observed word distributions

word distributions per topic

topic distributions per document

Probabilistic Latent Semantic Analysis (pLSA)



[Hofmann SIGIR 1999]

M = number of documents
N = number of words

John goes into the building, sits down waitress shows him menu. John orders. The waitress brings the food. John eats quickly, puts $10 on the table and leaves. ..

John goes the park with the magnolia trees and meets his friend, ...

Topic 1

waitress
$ food
Menu
...

Topic 2

....

park
Tree
...

pLSA

- Translating the document or text generation process into a joint probability model results in the expression

$$P(d_j, w_i) = P(d_j)P(w_i|d_j)$$

where $$P(w_i|d_j) = \sum_{k=1}^{K} P(w_i|z_k)P(z_k|d_j)$$

K = number of topics (a priori defined)

- Training = maximizing $$L = \sum_{j=1}^{M}\sum_{i=1}^{N} n(d_j, w_i)\log P(d_j, w_i)$$

where $n(d_j, w_i)$ = frequency of w_i in d_j
(e.g. trained with EM algorithm)

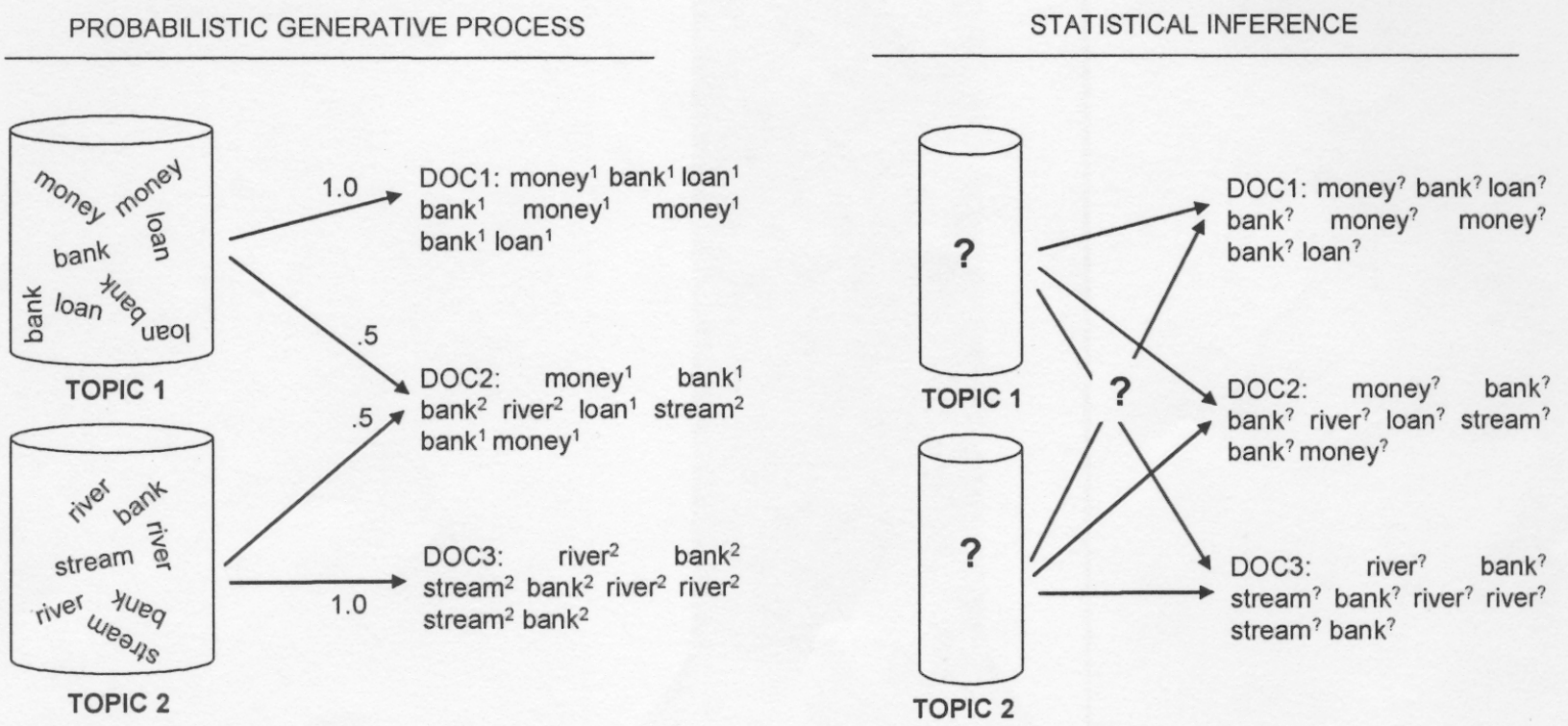
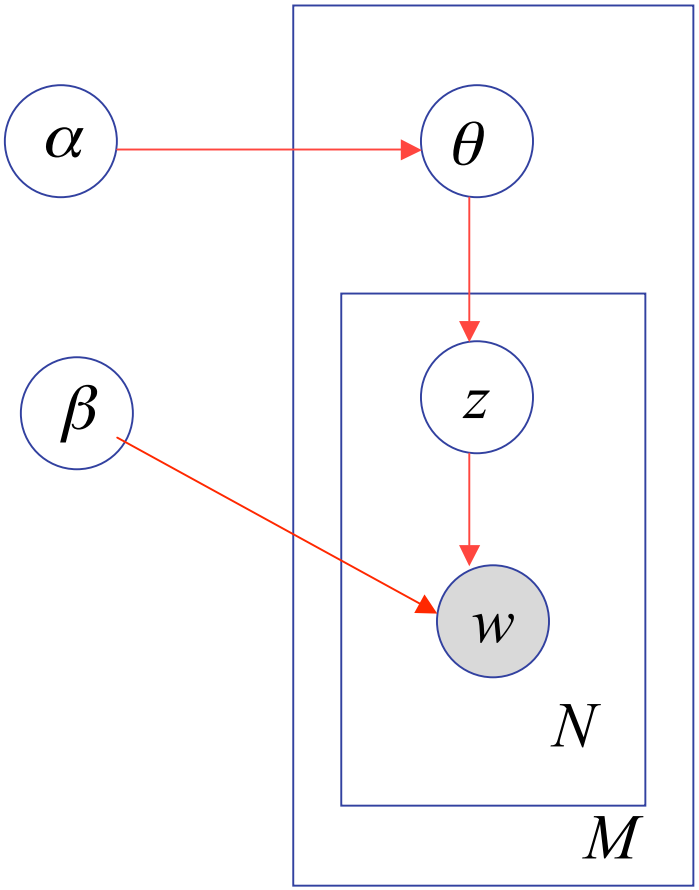


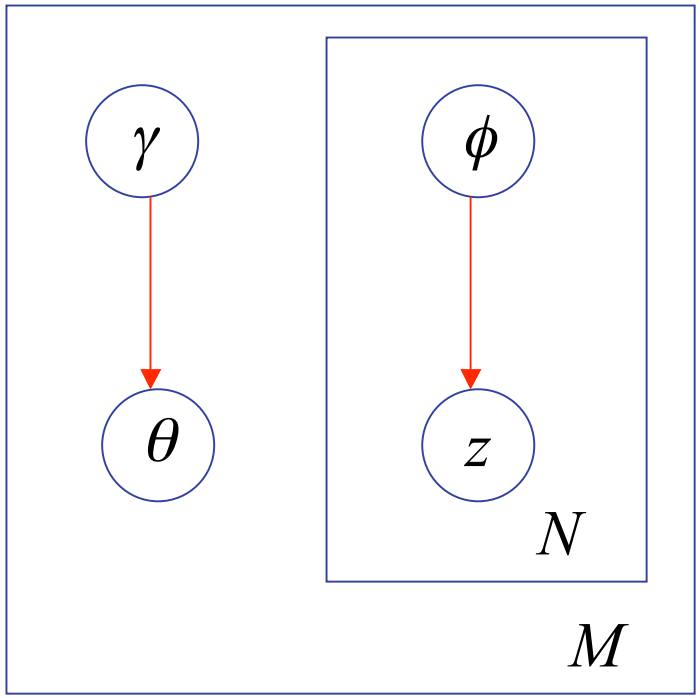
Figure 2. Illustration of the generative process and the problem of statistical inference underlying topic models

[Steyvers & Griffiths 2007]

Latent Dirichlet Allocation



(1)



(2)

[Blei et al. JMLR 2003]

Latent Dirichlet Allocation

- pLSA: learns $P(z_k|d_j)$ only for those documents on which it is trained
- Latent Dirichlet Allocation (LDA) treats topic mixture weights as a *k*-parameter hidden random variable θ
- Training
  - Key inferential problem: computing the distribution of the hidden variables θ and z given a document , i.e., $p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)$: intractable for exact inference
  - α: Dirichlet prior, can be interpreted as a prior observation count for the number of times a topic is sampled in a document, before having observed any actual words from that document

Latent Dirichlet Allocation

- Model 2 = simple modification of the original graphical model 1: the chain $\alpha \rightarrow \theta \rightarrow \boldsymbol{z}$ is replaced by $\gamma \rightarrow \theta$ and $\phi \rightarrow \boldsymbol{z}$
- Compute approximation of model 1 by model 2 for which the KL divergence KL$[p(\theta,\boldsymbol{z}|\gamma,\phi), q(\theta,\boldsymbol{z}|\ \boldsymbol{w},\alpha,\beta)$ is minimal
- Iterative updating of γ and ϕ for each document and recalculation of corpus-level variables α and β by means of EM algorithm

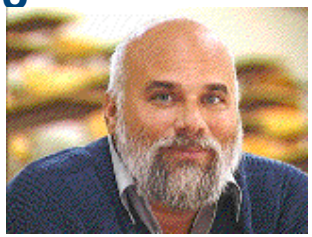
- Inference for new document:
  - Given α and β: we determine γ (topic distribution) and ϕ (word distribution) with a variational inference algorithm

Probabilistic topic models

- **Probabilistic models of text generation** (cf. model of text generation by Panini)
- **Understanding** by the machine = we **infer the latent structure** from which the document/text is generated
- Today:
  - Bag-of-words representations
  - Addition of other structural information is currently limited (e.g., syntax information in [Griffiths et al. ANIPS 2004])
  - But, acknowledged **potential for richly structured statistical models of language and text understanding** in general

Example

Script: human (X) taking the bus to go from LOC1 to LOC3



[Schank 1975]

1. X **PTRANS** X **from** LOC1 **to** bus stop
2. bus driver **PTRANS** bus **from** LOC2 **to** bus stop
3. X **PTRANS** X **from** bus stop **to** bus
4. X **ATRANS** money **from** X **to** bus driver
5. bus driver **ATRANS** ticket **to** X
6. Various subscripts handling actions possible during the ride.
7. bus driver **PTRANS** bus **from** bus stop **to** LOC3
8. X **PTRANS** X **from** bus **to** LOC3

X gives money to the bus driver. ATRANS is used to express a transfer of an abstract relationship, in this case the *possession* of money.

(3), (7), (8): mandatory

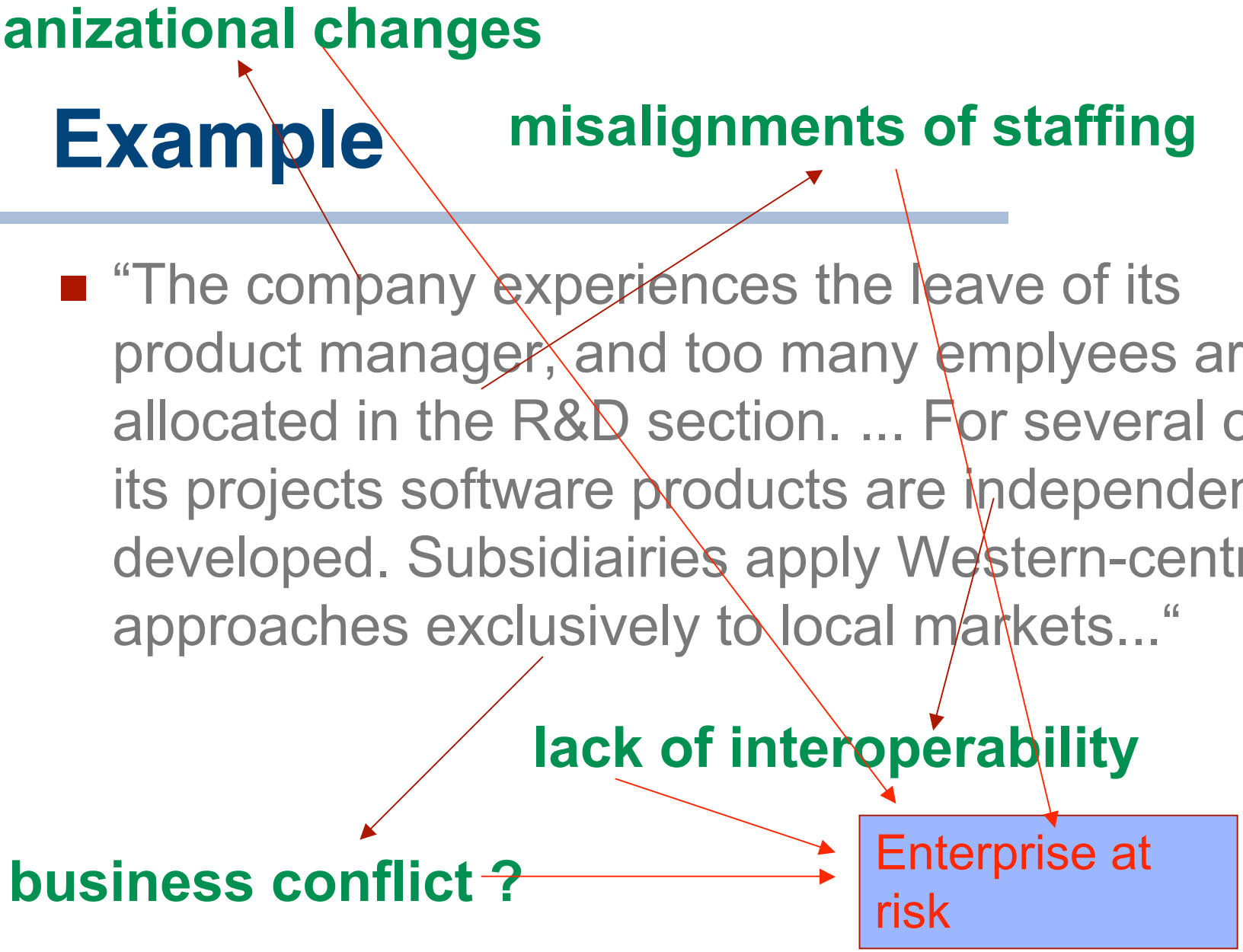
Example

- The doctors **did not do anything to save** a baby they knew was in **critical trouble**. **Despite knowing** the **childbirth** was in **crisis**, the doctors didn't do anything for more than an hour. **The effects were brain damage to the baby which result in the baby having cerebral palsy, spastic quadriplegia and a seizure disorder. The child is now more than five years old, but can't walk, talk, sit or stand.**

Medical malpractice

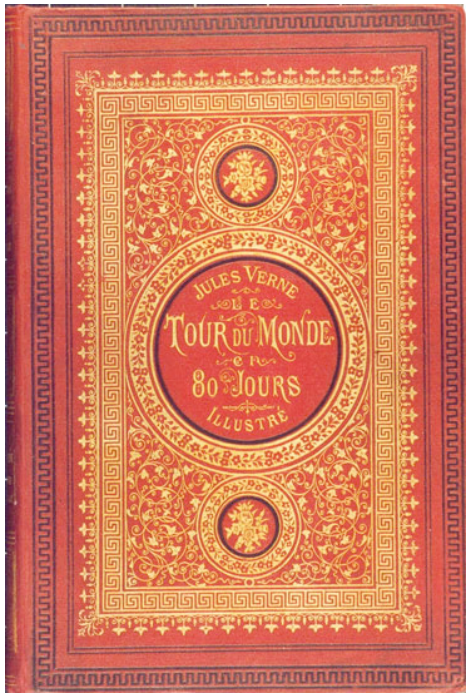
Example

- "The company experiences the leave of its product manager, and too many emplyees are allocated in the R&D section. ... For several of its projects software products are independently developed. Subsidiairies apply Western-centric approaches exclusively to local markets..."

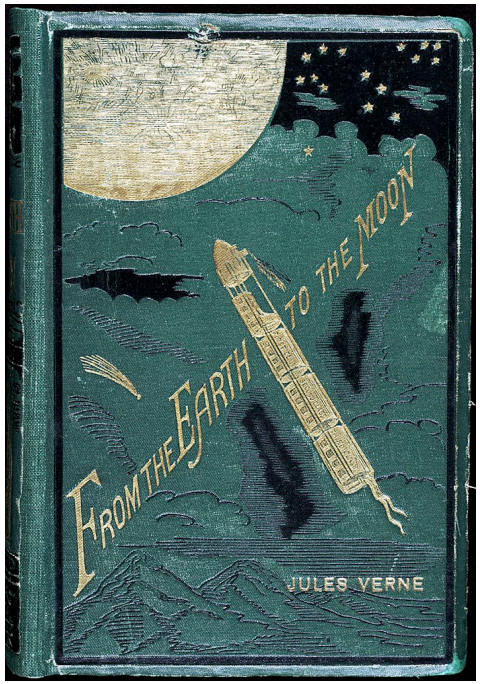


Extraction of complex concepts

- Semantic annotation performed by humans stretches **beyond** the **recognition of factoids** and the **identification of topic distributions**
- Humans understand media by labeling them with abstract scenarios, concepts or issues
- Very important for retrieval, mining and abstractive summarization of information, reasoning (e.g., Case Based Reasoning)
- But, is this possible for a computer?



Fact or Fiction



Problem

- The complex semantic concepts: are
  - not always literally present in a text
  - when present, how do we know that such a concept summarizes a whole passage/document?
- Given the multitude of semantic labels and the variety of natural language:
  - How can the machine learn to assign the labels with only **few hand-annotated examples**?
  - And still obtain **good accuracy** of the classification?

Solutions ?

- Complex semantic concepts:
  - Often hierarchically structured: composed of intermediary concepts and more simple concepts
  - Cf. model of text generation by Panini
- Exploit the **hierarchical structure** to:
  - Increase accuracy ?
  - Reduce number of training data ?
  - Cf. current work in computer vision

[Fei-Fei & Perona IEEE CVPR 2005] [Sudderth et al. IEEE ICCV 2005]

[Fan et al. SIGIR 2004]

Solutions ?

- **Naive model**: annotate texts and components with all kinds of semantic labels and train:
  - Probably few examples/ semantic category + variety of natural language => low accuracy

- **Train with structured examples** annotated with specific, intermediate and complex concepts
  - Some tolerance for incomplete patterns =>
    - possibly increased accuracy
    - still many annotations

Solutions ?

- **Cascaded /network approach**:
  - Learning intermediate models: the output of one type of semantic labeling forms the input of more complex tasks of classification (cf. FASTUS, cf. inverse of Panini model)
    - Possibly different or smaller feature sets can be used for models => less training examples needed
    - Reuse of component models possible
    - Natural integration of external knowledge resources
  - Several aggregation possibilities: features in feature vectors, Bayesian network, ...
  - But, errors propagate: keeping few best hypotheses ?

[Finke, Manning & Ng 2006] [Moens 2006]

Solutions ?

- **Extensions of the probabilistic topic models**:
  - Advantages of previous cascaded/network model
  - Unsupervised and different levels of supervision possible
  - Scalability?
  - Do the unlabeled examples:
    - learn us completely new patterns or only variations of existing patterns ?
    - cause learning incorrect patterns?

References

Bakir G.H., Hofmann, T.,Schölkopf, B., Smola, A.J., Taskar, B. & Vishwanathan, S.V.N. (2007) (Eds.), *Predicting Structured Data.* Cambridge, MA: MIT Press.

Blei, D.M., Ng, A.Y. and Jordan, M.I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993-1022.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science,* 41 (6), 391-407.

Fan, J., Gao, Y., Luo, Y. & Xu, G. (2004). Automatic image annotation by using concept-sensitive salient objects for image content representation. In *Proceedings of the Twenty-Seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 361-368). New York : ACM.

Fei-Fei, L. & Perona, P. (2005). A Bayesian hierarchical model for learning scene categories. IEEE-CVPR.

Finke, J.R., Manning, C.D. and Ng, A.Y (2006). Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

References

Griffiths, T.L., Steyvers, M. Blei, D.M. & Tenenbaum, J.B (2004). Integrating Topics and Syntax. Advances in Neural Information Processing Systems, 17.

Hobbs, J. R. (2002). Information extraction from biomedical text. *Journal of Biomedical Informatics*, 35, 260-264. [4] Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of SIGIR* (pp. 50-57). New York: ACM.

Kiparsky, Paul (2000). *On the Architecture of Panini's Grammar*. Three lectures delivered at the Hyderabad Conference on the Architecture of Grammar, January 2002, and at UCLA, March 2002.

Krishnan, V. & C.D. Manning (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. *Proceedings of COLING-ACL 2006* (pp. 1121-1128). East Stroudsburg, PA: ACL.

Moens, M.-F. (2008). *Learning Computers to Understand Text*, Inaugural lesson February 8, 2008.

Moens, M.-F. (2006). *Information Extraction: Algorithms and Prospects in a Retrieval Context* (*The Information Retrieval Series* 21). Berlin: Springer.

References

Schank, R.C. (1975). *Conceptual Information Processing.* Amsterdam: North Holland.

Steyvers, M. and Griffiths, T. (2007). Probabilistic topic models. In T. Landauer, D.S. McNamara, S. Dennis and W. Kintsch (Eds.), *The Handbook of Latent Semantic Analysis*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Sudderth, E.B., Torralba, A., Feeman, W.T. and Wilsky, A.S. (2005). Learning hierarchical models of scenes, objects and parts. In *Proceedings of the Tenth IEEE International Conference on Computer Vision, vol. 2* (pp. 1331-1338).

Sutton, C. & McCallum A. (2007). An introduction to Conditional Random Fields for relational learning. In L. Gtoor & B. Taskar (Eds.), *Statistical Relational Learning* (pp. 94-127). The MIT Press: Cambridge, MA.

Yang, Y. and Liu X. (1999). A re-examination of text categorization methods. In *Proceedings of SIGIR* (pp. 42-49). New York: ACM.