Introduction to Probabilistic Models for Information Retrieval

Victor Lavrenko
University of Edinburgh

Objectives

- Highlight influential work on probabilistic models for IR
- Provide a working understanding of the probabilistic techniques through a set of common implementation tricks
- Establish relationships between the popular approaches: stress common ideas, explain differences
- Outline issues in extending the models to interactive, cross-language, multi-media

Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and CLIR

Recap of Probability Theory

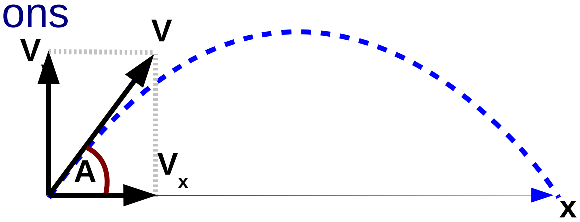
- Random variables and event spaces
  - sample space, events, and probability axioms
  - random variables and probability distributions
- Conditional probabilities and Bayes rule
- Independence and conditional independence
- Dealing with data sparseness
  - pairwise and mutual independence
  - dimensionality reduction and its perils
  - symmetry and exchangeability

What's a probability?

- Means many things to many people
  - inherent physical property of a system
    - … a coin toss comes up heads
  - (asymptotic) frequency of something happening
    - … Red Sox win against Yankees
  - subjective belief that something will happen
    - … the sun will rise tomorrow
- Laplace: *“common sense reduced to numbers”*
  - a very good substitute for scientific laws, when your scientific method can't handle the complexity

Coin-tossing example

- Toss a coin, determine how far it will land?
  - Newtonian physics: solve equations
    - Force * dt / Mass → velocity V
    - 2 * G / (V * sin(Angle)) → time T
    - T * V * cos (Angle) → distance X



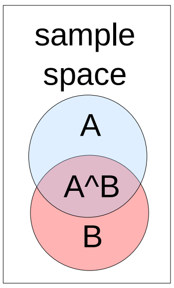
  - Probability / statistics: count coincidences
    - a gazillion throws, varying angle A, distance X
    - count how often we see X for a given A ,,, conditional P(X|A)
  - Why would we ever do that?
    - lazy, expensive, don't **really** understand what's going on...
    - can capture hidden factors that are difficult to account for
      - air resistance, effect of coin turning, wind, nearby particle accelerator...

Outcomes and Events

- Sample and Event Spaces:
  - sample space: all possible **outcomes** of some experiment
  - event space: all possible **sets** of outcomes (power-set**)
- Examples:
  - toss a coin, measure how far it lands
    - outcome: e.g. coin lands at exactly 12.34567m (uncountably many)
    - event: range of numbers, coin landed between 12m and 13m
  - toss a coin twice, record heads / tails on each toss
    - sample space: {HH, HT, TH, TT} – only four possible outcomes
    - event space: {{}, {HH}, {HT}…, {HH,HT}, {HH,TH}…, {HH,HT,TH}…, }
      - {HH,HT} = event that a head occurred on the first toss
      - {HH,HT,TH} = event that a head occurred on at least one of the tosses

Probabilities

- Probability = how frequently we expect an event
  - e.g. fair coin → P(H) = P(T) = ½
  - assigned to **events**, not **outcomes**:
    - i.e. P(H) really means P({H}), but notation {} frequently dropped
- Probabilities must obey rules:
  - for any event: 0 <= P(event) <= 1
  - P(sample space) = 1 … some outcome must occur
  - for any events A,B: P(A U B) = P(A) + P(B) – P(A ^ B)
    - P(A U B) = P(A) + P(B) if events don't overlap (e.g. {HH, HT}+{TT})
    - Σ_{outcome} P({outcome}) = 1 … additivity over sample space

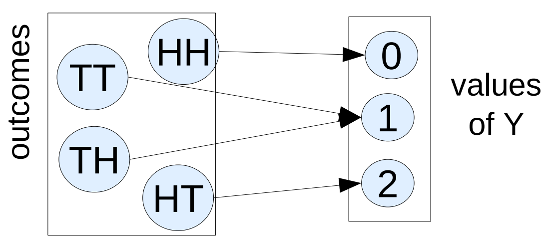


Random Variables

- RV = a function defined over sample space
  - compute some property / feature of an outcome, e.g.:
    - X: coin toss distance, truncated to nearest imperial unit
      - X(0.023) = “inch”, X(0.8) = “yard”, X(1500.1) = “mile”, …
    - Y: number of heads observed during two coin tosses
      - Y(HH) = 2, Y(HT) = Y(TH) = 1, Y(TT) = 0
  - RVs … capital letters, their values … lowercase
- Central notion in probabilistic approaches:
  - very flexible and convenient to work with:
    - can map discrete outcomes to numeric, and back
    - often describe everything in terms of RVs (forget sample space)

Random Variables and Probabilities

- RVs usually deterministic (counting, rounding)
- What they operate on (outcomes) is probabilistic
  - probability RV takes a particular value is defined by the probabilities of outcomes that lead to that value:
    - P(Y=2) = P(two heads in two tosses) = P ({HH})
    - P(Y=1) = P(exactly one head) = P({HT}) + P({TH})
    - P(X=”foot”) = P(distance rounds to “foot”) = P(0.1 < distance < 0.5)
- In general: $P(X=x) \;=\; \Sigma_{\text{outcome : X(outcome) = x}}$ P({outcome})



Random Variables Confusion

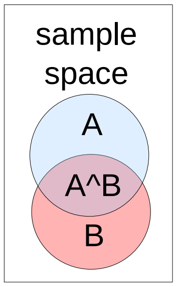
- Full RV notation is tedious
  - frequently shortened to list just variables, or just values:
    - $P(X_1 = x_1, X_2 = x_2, Y = y) \rightarrow P(X_1, X_2, Y)$
    - $P(X_1$=yard, W_2=mile$) \rightarrow$ P(yard,mile)
- Fine, as long as clear what RVs mean:
  - for 2 coin-tosses P("head") can mean:
    - P(head on the first toss) = P({HH}) + P({HT})
    - P(a head was observed) = P({HH}) + P({HT}) + P({TH})
    - P(exactly one head observed) = P({HT}) + P({TH})
  - these mean different things, can't be interchanged
- In general: clearly define the domain for each RV.

Types of Random Variables

- Completely determined by domain (types of output)
- Discrete: RV values = finite or countable
  - ex: coin tossing, dice-rolling, counts, words in a language
  - additivity: $\sum_x P(x) = 1$
    - P(X = x) is a sensible concept
- Continuous: RV values are real numbers
  - ex: distances, times, parameter values for IR models
  - additivity: $\int_x p(x) dx = 1$
    - P(X = x) is always zero, p(x) is a "density" function
- Singular RVs … never see them in IR

Conditional Probabilities

- P(A | B) … probability of event A happening
  assuming we know B happened



$$P(A|B)=\frac{P(A\cap B)}{P(B)}$$

- Example:
  - population size: 10,000,000
  - number of scientists: 10,000
  - Nobel prize winners: 10 (1 is an engineer)
  - P(scientist) = 0.001
  - P(scientist | Nobel prize) = 0.9

Bayes Rule

- A way to "flip" conditional probabilities:

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

- Example:
  - P(scientist | Nobel prize) = 0.9
  - P(Nobel prize) = 10^{-6}, P(scientist) = 10^{-3}
  - P(Nobel prize | scientist) = 0.9 * 10^{-6} / 10^{-3} = 0.0009
- Easy to derive (definition of conditional probabilities):

$$P(A|B)=\frac{P(A\cap B)}{P(B)}=\frac{P(A\cap B)}{P(A)}\times\frac{P(A)}{P(B)}=\frac{P(B|A)P(A)}{P(B)}$$

Chain Rule and Independence

- Chain Rule: a way to decompose joint probabilities
  - directly from definition of conditionals
  - exact, no assumptions are involved

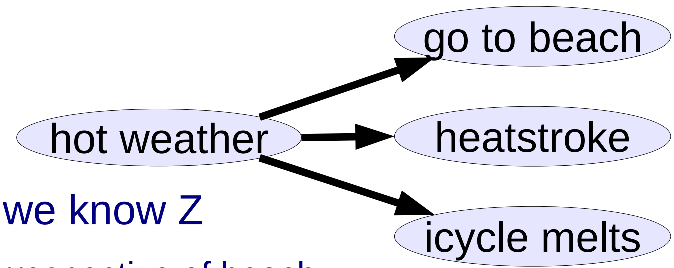
$$P(X_1 \ldots X_n) = P(X_1|X_2 \ldots X_n)\ P(X_2|X_3 \ldots X_n)\ \ldots\ P(X_n)$$

- Independence:
  - X and Y are independent (don't influence each other)
  - coin example: distance travelled and whether it's H or T
    - probably doesn't hold for very short distances
  - mutual independence: multiply probabilities (cf. Chain rule):

$$P(X_1 \ldots X_n) = \prod_{i=1}^{n} P(X_i)$$

Conditional Independence

- Variables X and Y may be dependent
  - but all influence can be explained by another variable Z
    - X: you go to the beach
    - Y: you get a heatstroke
    - Z: the weather is hot



  - X and Y are independent if we know Z
    - if weather is hot, heatstroke irrespective of beach

$$P(X,Y|Z) = P(X|Z)\ P(Y|Z)$$

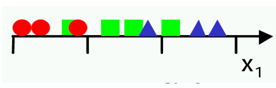
  - if Z is unknown, X and Y are dependent

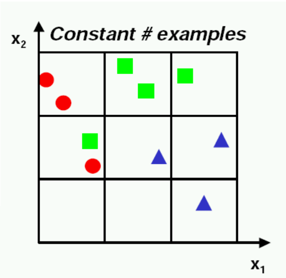
$$P(X,Y) = \sum_z P(X|Z=z)\, P(Y|Z=z)\, P(Z=z)$$

- Don't mix conditional and mutual independence

Curse of dimensionality

- Why do we need to assume independence?
- Probabilistic models based on counting
  - count observations (documents)
  - of different classes (relevant / non-relevant)
  - along different regions of space (words)
- As dimensionality grows, fewer dots per region
  - 1d: 3 regions, 2d: 3^2 regions, 1000d – hopeless
  - statistics need repetition
    - flip a coin once → head
    - P(head) = 100%?





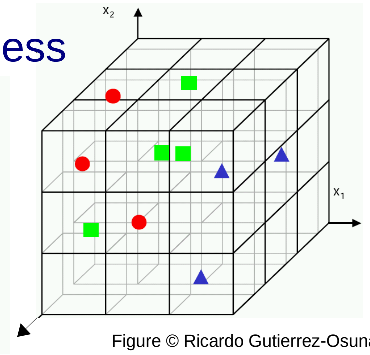
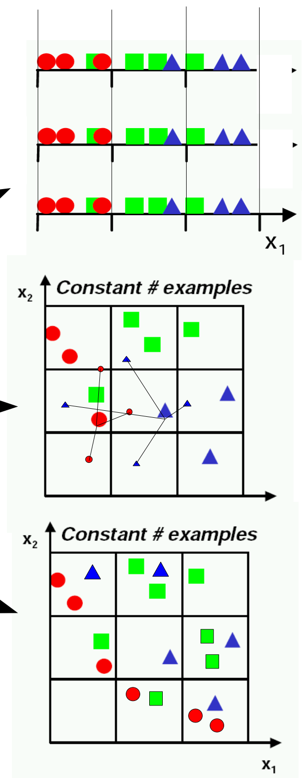


Figure © Ricardo Gutierrez-Osuna

Dealing with high dimensionality

- Use domain knowledge
  - feature engineering: doesn't really work for IR
- Make assumption about dimensions
  - independence
    - count along each dimension separately, combine
  - smoothness
    - propagate class counts to neighbouring regions
  - symmetry
    - e.g. invariance to order of dimensions: $x_1 \leftrightarrow x_2$
- Reduce the dimensionality of the data
  - create a new set of dimensions (variables)



Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and feedback

Models in Information Retrieval

- Mathematical formalism for processes:
  - formulation: information need → query
  - indexing: documents → index terms
  - **retrieval: query + corpus → search results**
- Over the following variables
  - documents (D), queries (Q), relevance (R)
  - user, task, context, search history, click rate, …
- Usually involve abstract analogy
  - document is an urn containing words
  - query is a logical formula that needs to be "proved"
  - user is a greedy memory-less random process

Probability Ranking Principle

- Robertson (1977)
  - "If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request,
  - where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose,
  - the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data."
- Basis for most probabilistic approaches to IR

Let's dissect the PRP

- rank documents … by probability of relevance
  - P (relevant | document)
- estimated as accurately as possible
  - P_{est} (relevant | document) → P_{true} (rel | doc) in some way
- based on whatever data is available to system
  - P_{est} (relevant | document, query, context, user profile, …)
- best possible accuracy one can achieve with that data
  - recipe for a perfect IR system: just need P_{est} (relevant | …)
  - strong stuff, can this really be true?

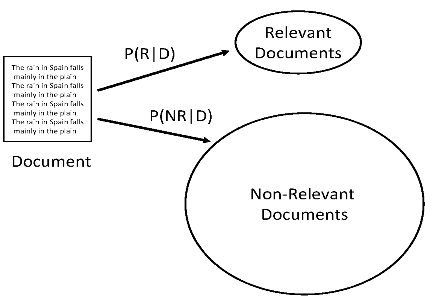
Copyright 2010, Victor Lavrenko

Probability of relevance

- What is: P_{true} (relevant | doc, qry, user, context) ?
  - isn't relevance just the user's opinion?
    - user decides relevant or not, what's the "probability" thing?
- "user" does not mean the human being
  - doc, qry, user, context … *representations*
    - parts of the real thing that are available to the system
  - typical case: P_{true} (relevant | document, query)
    - query: 2-3 keywords, user profile unknown, context not available
    - whether document is relevant is uncertain
      - depends on the factors which are not *available to our system*
    - think of P_{true} (rel | doc,qry) as proportion of all unseen users/contexts/... for which the document would have been judged relevant

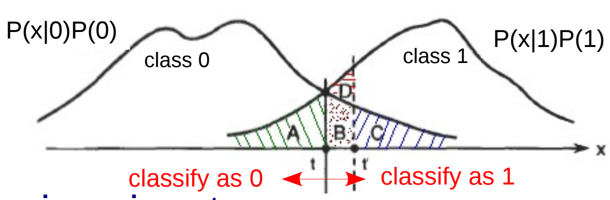
IR as classification

- For a given query, documents fall into two classes
  - relevant (R=1) and non-relevant (R=0)
  - compute P(R=1|D) and P(R=0|D)
    - retrieve if P(R=1|D) > P(R=0|D)



- Related to Bayes error rate
  - if P(x|0) P(0) > P(x|1) P(1) then class 0 otherwise 1
  - $\text{error}_{\text{Bayes}}$ = A + (B + C) <= A + B + C + D = $\text{error}_{\text{any other classifier}}$
  - no way to do better than Bayes given input x
    - input x does not allow us to determine class any better

$$y^{best} = argmax\, P(x|y)\, P(y)$$



Optimality of PRP

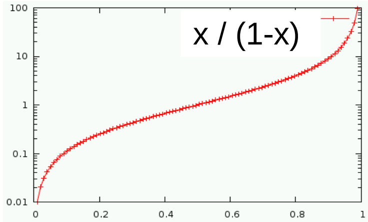
- Retrieving a set of documents:
  - PRP equivalent to Bayes error criterion
  - optimal wrt. classification error
- Ranking a set of documents: optimal wrt:
  - precision / recall at a given rank
  - average precision, etc.
- Need to estimate P(relevant | document, query)
  - many different attempts to do that
  - Classical Probabilistic Model (Robertson, Sparck-Jones)
    - also known as Binary Independence model, Okapi model
    - very influential, successful in TREC (BM25 ranking formula)

Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and feedback

Classical probabilistic model

- Assumption A0:
  - relevance of D doesn't depend on any other document
    - made by almost every retrieval model (exception: cluster-based)
- Rank documents by P(R=1|D)
  - R = {0,1} … Bernoulli RV indicating relevance
  - D … represents content of the document
- Rank-equivalent:

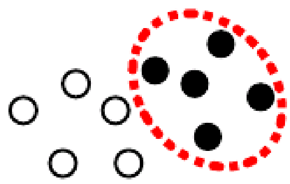


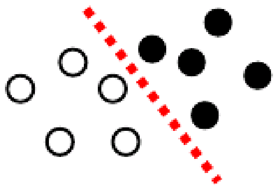
$$P(R=1|D) \stackrel{rank}{=} \frac{P(R=1|D)}{P(R=0|D)} = \frac{P(D|R=1)\,P(R=1)}{P(D|R=0)\,P(R=0)}$$

- Why Bayes? Want a generative model.
  - P (observation | class) sometimes easier with limited data
  - note: P(R=1) and P(R=0) don't affect the ranking

Generative and Discriminative

- A complete probability distribution over documents
  - defines likelihood for any possible document *d* (observation)
  - P(relevant) via P(document): $P(R|d) \propto P(d|R)\,P(R)$
  - can "generate" synthetic documents
    - will share some properties of the original collection
- Not all retrieval models do this
  - possible to estimate P(*R*|*d*) directly
  - e.g. log-linear model





$$P(R|d) = \frac{1}{z_R} \exp\left(\sum_i \lambda_i\, g_i(R,d)\right)$$

Probabilistic model: assumptions

- Want P(D|R=1) and P(D|R=0)
- Assumptions:
  - A1: D = {D_w} … one RV for every word w
    - Bernoulli: values 0,1 (word either present or absent in a document)
  - A2: D_w ... are mutually independent given R
    - blatantly false: presence of "Barack" tells you nothing about "Obama"
    - but must assume something: D represents subsets of vocabulary
      - without assumptions: 10^6! possible events
  - allows us to write:

$$P(R=1|D) \overset{rank}{=} \frac{P(D|R=1)}{P(D|R=0)} = \frac{\prod_w P(D_w|R=1)}{\prod_w P(D_w|R=0)}$$

- Observe: identical to the Naïve Bayes classifier

Probabilistic model: assumptions

- Define: p_w = P(D_w=1|R=1) and q_w = P(D_w=1|R=0)
- Assumption A3 : $P(\vec{0}|R=1) = P(\vec{0}|R=0)$
  - empty document (all words absent) is equally likely to be observed in relevant and non-relevant classes
- Result:

$$P(R=1|D) \overset{rank}{=} \prod_{w \in D}\left(\frac{p_w}{q_w}\right)\prod_{w \notin D}\left(\frac{1-p_w}{1-q_w}\right) / \underbrace{\prod_{w}\left(\frac{1-p_w}{1-q_w}\right)}_{\frac{P(\vec{0}|R=1)}{P(\vec{0}|R=0)}=1} = \prod_{w \in D}\frac{p_w(1-q_w)}{q_w(1-p_w)}$$

  - dividing by 1: no effect
  - provides "natural zero"
  - practical reason: final product only over words present in D
    - fast: small % of total vocabulary + allows term-at-a-time execution

Estimation (with relevance)

- Suppose we have (partial) relevance judgments:
  - N_1 … relevant, N_0 … non-relevant documents marked
  - word w observed in $N_1(w)$, $N_0(w)$ docs
  - P(w) = % of docs that contain at least one mention of w
    - includes crude smoothing: avoids zeros, reduces variance

$$p_w = \frac{N_1(w)+0.5}{N_1+1.0} \qquad q_w = \frac{N_0(w)+0.5}{N_0+1.0}$$

- What if we don't have relevance information?
  - no way to count words for relevant / non-relevant classes
  - things get messy...

Example (with relevance)

- relevant docs: D_1 = "a b c b d", D_2 = "a b e f b"
- non-relevant: D_3 = "b g c d", D_4 = "b d e", D_5 = "a b e g"
- word:

word:	a	b	c	d	e	f	g	h	
$N_1(w)$:	2	2	1	1	1	1	0	0	$N_1 = 2$
$N_0(w)$:	1	3	1	2	2	0	2	0	$N_0 = 3$
p_w:	$^{2.5}/_3$	$^{2.5}/_3$	$^{1.5}/_3$	$^{1.5}/_3$	$^{1.5}/_3$	$^{1.5}/_3$	$^{0.5}/_3$	$^{0.5}/_3$	
q_w:	$^{1.5}/_4$	$^{3.5}/_4$	$^{1.5}/_4$	$^{2.5}/_4$	$^{2.5}/_4$	$^{0.5}/_4$	$^{2.5}/_4$	$^{0.5}/_4$	

- new document D_6 = "b g h":

$$P(R=1|D_6) \overset{rank}{=} \prod_{\underbrace{w \in D_6}_{\text{only words present in } D_6}} \frac{p_w(1-q_w)}{q_w(1-p_w)} = \frac{\frac{2.5}{3}\cdot(1-\frac{3.5}{4})\cdot\frac{0.5}{3}\cdot(1-\frac{2.5}{4})\cdot\frac{0.5}{3}\cdot(1-\frac{0.5}{4})}{\frac{3.5}{4}\cdot(1-\frac{2.5}{3})\cdot\frac{2.5}{4}\cdot(1-\frac{0.5}{3})\cdot\frac{0.5}{4}\cdot(1-\frac{0.5}{3})} = \frac{1.64}{13.67}$$

Estimation (no relevance)

- Assumption A4: $p_w = q_w \; if \; w \notin Q$
  - if the word is not in the query, it is equally likely to occur in relevant and non-relevant populations
  - practical reason: restrict product to query – document overlap
- Assumption A5: $p_w = 0.5 \; if \; w \in Q$
  - a query word is equally likely to be present and absent in a randomly-picked relevant document (usually p_w << 0.5)
  - practical reason: p_w and $(1-p_w)$ cancel out
- Assumption A6: $q_w \approx N_w / N$
  - non-relevant set approximated by collection as a whole
  - very reasonable: most documents are non-relevant
- Result: $P(R=1|D) \overset{rank}{=} \prod_{w \in D} \frac{p_w(1-q_w)}{q_w(1-p_w)} = \prod_{w \in D \cap Q} \frac{1-q_w}{q_w} = \prod_{w \in D \cap Q} \underbrace{\boxed{\frac{N - N_w + 0.5}{N_w + 0.5}}}_{\text{IDF}}$

Example (no relevance)

- documents: D_1 = “a b c b d”, D_2 = “b e f b”, D_3 = “b g c d”, D_4 = “b d e”, D_5 = “a b e g”, D_6 = “b g h”

word:	a	b	c	d	e	f	g	h	
N(w):	2	6	2	3	3	1	3	1	N = 6
$^{N-Nw}/_{Nw}$:	$^{4.5}/_{2.5}$	$^{0.5}/_{6.5}$	$^{4.5}/_{2.5}$	$^{3.5}/_{3.5}$	$^{3.5}/_{3.5}$	$^{5.5}/_{1.5}$	$^{3.5}/_{3.5}$	$^{5.5}/_{1.5}$	

- query: Q = “a c h”

$$P(R=1|D_1) \overset{rank}{=} \underbrace{\prod_{w \in Q \cap D_1}}_{\text{only words present in both D \& Q}} \frac{N - N_w + 0.5}{N_w + 0.5} = \frac{4.5}{2.5} \cdot \frac{4.5}{2.5}$$

$$P(R=1|D_2) \overset{rank}{=} 1$$

$$P(R=1|D_3) \overset{rank}{=} \frac{4.5}{2.5}$$

$$P(R=1|D_4) \overset{rank}{=} 1$$

$$P(R=1|D_5) \overset{rank}{=} \frac{4.5}{2.5}$$

$$P(R=1|D_6) \overset{rank}{=} \frac{5.5}{1.5}$$

Ranking:
D_6
D_1
D_3
D_5
D_2
D_4

Probabilistic model (review)

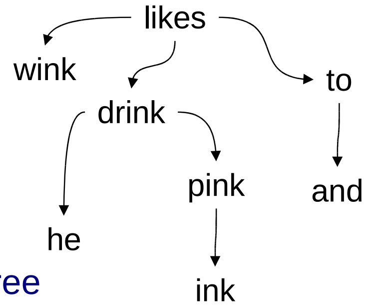
- Probability Ranking Principle: best possible ranking
- Assumptions: $$P(R=1|D) \overset{rank}{=} \prod_{w \in D} \frac{p_w}{q_w} \prod_{w \notin D} \frac{1-p_w}{1-q_w} = \prod_{w \in D \cap Q} \frac{N-N_v}{N_v}$$
  - A0: relevance for document in isolation
  - A1: words absent or present (can't model frequency)
  - A2: all words mutually independent (given relevance)
  - A3: empty document equally likely for R=0,1
  - A4: non-query words cancel out

    efficiency
  - A5: query words: relevant class doesn't matter
  - A6: non-relevant class ~ collection as a whole

    estimate p_w, q_w w/out relevance observations
- How can we improve the model?

Modeling word dependence

- Classical model assumes all words independent
  - blatantly false, made by almost all retrieval models
  - the most widely criticized assumption behind IR models
    - should be able to do better, right?
- Word dependence models
  - details in part II of the tutorial
  - preview: (van Rijsbergen, 1977)
    - structure dependencies as maximum spanning tree
    - each word depends on its parent (and R)




```
P("he likes to wink and drink pink ink")
= P(likes) * P(to|likes) * P(wink|likes) * P(and|to)
* P(drink|likes) * P(he|drink) * P(pink|drink) * P(ink|pink)
```

Why dependency models fail

- Word independence constantly criticized
  - blatantly wrong assumption about language
  - numerous attempts to model dependency
  - never a consistent improvement
- Language Modeling Framework
  - dependency models address wrong problem
    - focus on surface form of the string
  - we are dealing with already well-formed strings
- Classical Probabilistic Framework
  - does not in fact assume word independence

BIR **doesn't** assume independence

$$\frac{P_{R=1}(\vec{d})}{P_{R=0}(\vec{d})} = \underbrace{\prod_v \frac{P_1(d_v)}{P_0(d_v)}}_{\textbf{independence}} \times \underbrace{\prod_v \frac{k_1(v)}{k_0(v)}}_{\textbf{will not affect ranking if}} = \underbrace{\prod_v \frac{P_1(d_v|d_{\pi(v)})}{P_0(d_v|d_{\pi(v)})}}_{\mathbf{1^{st}}\textbf{ order dependence}}$$

$$\boxed{k_r(v) = \frac{P_r(d_v, d_{\pi(v)})}{P_r(d_v)\,P_r(d_{\pi(v)})}}$$

$$\underbrace{\sum_v \log \frac{P_1(d_v, d_{\pi(v)})}{P_1(d_v)\,P_1(d_{\pi(v)})}}_{\substack{\textbf{aggregate dependence}\\ \textbf{between word and parent}\\ \textbf{in the relevant class}}} \sim \underbrace{\sum_v \log \frac{P_0(d_v, d_{\pi(v)})}{P_0(d_v)\,P_0(d_{\pi(v)})}}_{\substack{\textbf{aggregate dependence}\\ \textbf{in the non-relevant class}}}$$

- Sufficient condition: proportional interdependence

  *the **total** amount of interdependence among **all** words in a document is approximately the same under R=1 and R=0*

Meaning of Independence

- Independence:
  - seeing “subprime” doesn't affect chances of seeing “loan”
- Linked Dependence:
  - seeing “subprime” increases chance of seeing “loan”
  - by the same amount under R=1 and R=0
    - reasonable... unless topic is financial crisis
- Proportional Interdependence:
  - “subprime” increases chance of “loan”
  - can be more co-dependent in relevant class
  - as long as offset by other word sets under R=0
    - “world cup” more co-dependent in non-relevant class

Probabilistic model (review)

- Probability Ranking Principle: best possible ranking
- Assumptions:

$$P(R=1|D) \overset{rank}{=} \prod_{w \in D} \frac{p_w}{q_w} \prod_{w \notin D} \frac{1-p_w}{1-q_w} = \prod_{w \in D \cap Q} \frac{N-N_v}{N_v}$$

  - A0: relevance for document in isolation
  - $\boxed{\text{A1: words absent or present (can't model frequency)}}$
  - A2: all words mutually independent (given relevance)
  - A3: empty document equally likely for R=0,1
  - A4: non-query words cancel out

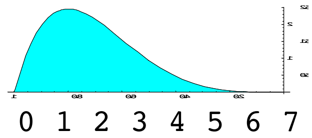
  (A3, A4: efficiency)

  - A5: query words: relevant class doesn't matter
  - A6: non-relevant class ~ collection as a whole

  (A5, A6: estimate p_w, q_w w/out relevance observations)
- How can we improve the model?

Modeling word frequencies

- Want to model TF (empirically useful) $P(R=1|D) \overset{rank}{=} \prod_{w \in D} \frac{P(d_w|R=1)}{P(d_w|R=0)}$
  - A1': assume $D_w=d_w$… # times word w occurs in document D
  - estimate $P(d_w|R)$: e.g. “obama” occurs 5 times in a rel. doc
  - naive: separate prob.for every outcome: $p_{w,1}$, $p_{w,2}$, $p_{w,3}$, ...
    - many outcomes → many parameters (BIR had only one p_w)
    - “smoothness” in the outcomes: d_w=5 similar to d_w=6, but not d_w=1
  - parametric model: assume d_w ~ Poisson
    - single parameter m_w... expected frequency
  - problem: Poisson a poor fit to observations
    - does not capture bursty nature of words



$$P(d_w) = \frac{e^{-\mu_w} \mu_w^{d_w}}{d_w!}$$

Two-Poisson model [Harter]

- Idea: words generated by a mixture of two Poissons
  - “elite” words for a document: occur unusually frequently
  - “non-elite” words – occur as expected by chance
  - document is a mixture: $P(d_w) = P(E=1)\frac{\exp^{-\mu_{1,w}} \mu_{1,w}^{d_w}}{d_w!} + P(E=0)\frac{\exp^{-\mu_{0,w}} \mu_{0,w}^{d_w}}{d_w!}$
    - estimate $m_{0,w}$, $m_{1,w}$, P(E=1) by fitting to data (max. likelihood)
- Problem: need probabilities conditioned on relevance
    - “eliteness” not the same as relevance
    - Robertson and Sparck Jones: condition eliteness on R=0, R=1
      - final form has too many parameters, and no data to fit them...
      - same problem that plagued BIR
- BM25: an “approximation” to conditioned 2-Poisson

$$\frac{p_w(d_w) q_w(0)}{q_w(d_w) p_w(0)} \approx \exp\left(\frac{d_w \cdot (1+k)}{d_w + k \cdot ((1-b) + b \cdot n_d / n_{avg})} \times \log \frac{N}{N_w}\right)$$

BM25: an intuitive view

Repetitions of query words ➔ good

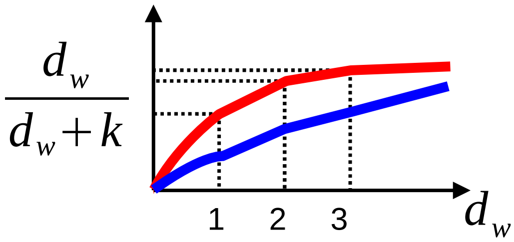
Common words less important

$$\log\frac{p(d|R=1)}{p(d|R=0)} \approx \sum_w \left(\frac{d_w\cdot(1+k)}{d_w + k\cdot((1-b)+b\cdot n_d/n_{avg})} \times \log\frac{N}{N_w} \right)$$

More words in common with the query ➔ good

Repetitions less important than different query words

But more important if document is relatively long (wrt. average)



Example (BM25)

- documents: D_1 = “a b c b d”, D_2 = “b e f b”, D_3 = “b g c d”, D_4 = “b d e”, D_5 = “a b e g”, D_6 = “b g h h”
- query: Q = “a c h”, assume k = 1, b = 0.5
-

word:	a	b	c	d	e	f	g	h	
N(w):	2	6	2	3	3	1	3	1	N = 6
$^{N-Nw}/_{Nw}$:	$^{4.5}/_{2.5}$	$^{0.5}/_{6.5}$	$^{4.5}/_{2.5}$	$^{3.5}/_{3.5}$	$^{3.5}/_{3.5}$	$^{5.5}/_{1.5}$	$^{3.5}/_{3.5}$	$^{5.5}/_{1.5}$	

-

$$\log\frac{p(D_1|R=1)}{p(D_1|R=0)} \approx 2\times\left(\frac{1\cdot(1+1)}{1+1\cdot(0.5+0.5\cdot 5/4)} \times \log\frac{6+1}{2+0.5}\right)$$

$$\log\frac{p(D_6|R=1)}{p(D_6|R=0)} \approx \left(\frac{2\cdot(1+1)}{2+1\cdot(0.5+0.5\cdot 4/4)} \times \log\frac{6+1}{1+0.5}\right)$$

Summary: probabilistic model

- Probability Ranking Principle
  - ranking by P(R=1|D) is optimal
- Classical probabilistic model
  - words: binary events (relaxed in the 2-Poisson model)
  - words assumed independent (not accurate)
    - numerous attempts to model dependence, all without success
- Formal, interpretable model
  - explicit, elegant model of relevance (if observable)
  - very problematic if relevance not observable
    - authors resort to heuristics, develop BM25

Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and feedback

What is a Language Model?

- Probability distribution over strings of text
  - how likely is a given string (observation) in a given “language”
  - for example, consider probability for the following four strings
  - English: $p_1 > p_2 > p_3 > p_4$

P_1 = P(“a quick brown dog”)

P_2 = P(“dog quick a brown”)

P_3 = P(“un chien quick brown”)

P_4 = P(“un chien brun rapide”)

  - … depends on what “language” we are modeling
  - in most of IR we will have $p_1 == p_2$
  - for some applications we will want p_3 to be highly probable

Language Modeling Notation

- Make explicit what we are modeling:

  M … represents the language we're trying to model

  s … “observation” (strings of tokens / words)

  P(s|M) … probability of observing “s” in language M

- M can be thought of as a “source” or a generator
  - a mechanism that can produce strings that are legal in M

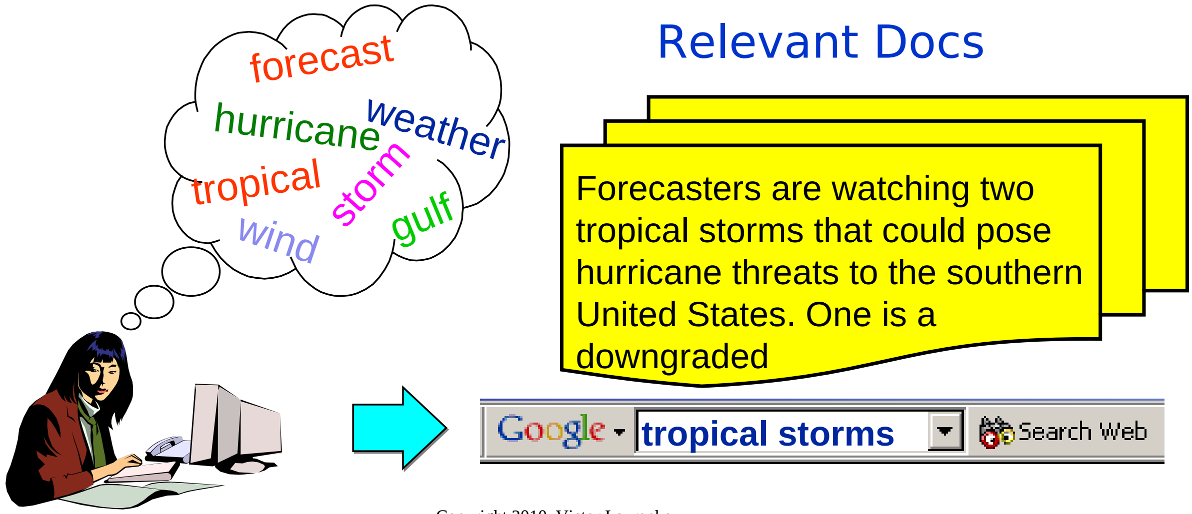
  P(s|M) … probability of getting “s” during repeated random sampling from M

How can we use LMs in IR?

Use LMs to model the process of query generation:

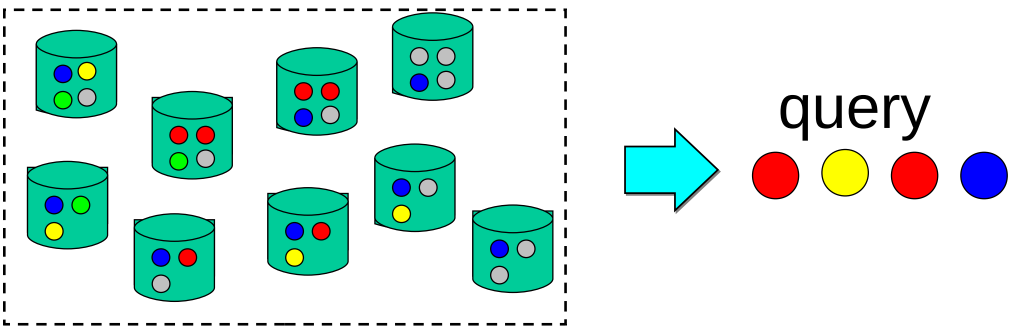
– user thinks of some relevant document

– picks some keywords to use as the query



Retrieval with Language Models

- Each document D in a collection defines a "language"
  - all possible sentences the author of D could have written
  - $P(s|M_D)$ … probability that author would write string "s"
    - intuition: write a billion variants of D, count how many times we get "s"
    - language model of what the author of D was trying to say
- Retrieval: rank documents by $P(q|M_D)$
  - probability that the author would write "q" while creating D

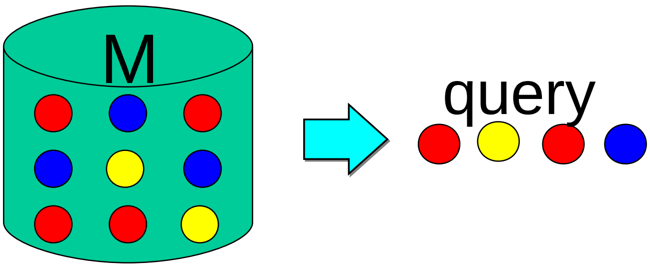


Major issues in applying LMs

- What kind of language model should we use?
  - Unigram or higher-order models?
  - Multinomial or multiple-Bernoulli?
- How can we estimate model parameters?
  - maximum likelihood and zero frequency problem
  - discounting methods: Laplace, Lindstone and Good-Turing estimates
  - interpolation methods: Jelinek-Mercer, Dirichlet prior, Witten-Bell
  - leave-one-out method
- Ranking methods
  - query likelihood / document likelihood / model comparison

Unigram Language Models

- words are “sampled” independently of each other
  - metaphor: randomly pulling out words from an urn (w. replacement)
  - joint probability decomposes into a product of marginals
  - estimation of probabilities: simple counting

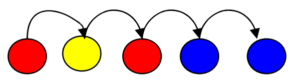


P (● ● ● ●) = P (●) P (●) P (●) P (●)

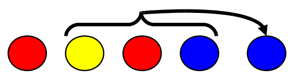
= 4 / 9 * 2 / 9 * 4 / 9 * 3 / 9

Higher-order Models

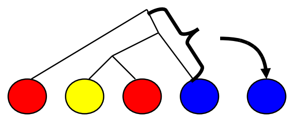
- Unigram model assumes word independence
  – cannot capture surface form: P(“brown dog”) == P(“dog brown”)
- Higher-order models
  – n-gram: condition on preceding words:



  – cache: condition on a window (cache):



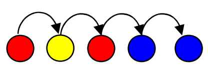
  – grammar: condition on parse tree



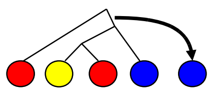
- Are they useful?
  – no improvements from n-gram, grammar-based models
  – some research on cache-like models (proximity, passages, etc.)
  – parameter estimation is prohibitively expensive

Why unigram models?

- Higher-order LMs useful in other areas
  – n-gram models: critical in speech recognition



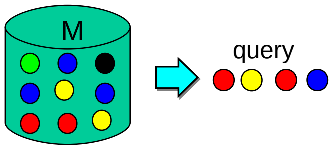
  – grammar-based models: successful in machine translation



- IR experiments: no improvement over unigram
  – unigram assumes word independence, intuitively wrong
  – no conclusive reason, still subject of debate
- Possible explanation: solving a non-existent problem
  – higher-order language models focus on surface form of text
  – ASR / MT engines must produce well-formed, grammatical utterances
  – in IR all utterances (documents, queries) are already grammatical
- What about phrases?
  – bi-gram: $O(v^2)$ parameters, there are better ways

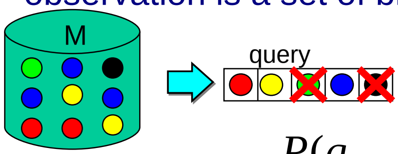
Multinomial or multiple-Bernoulli?

- Most popular model is the multinomial:
  - fundamental event: *what word is in the i'th position in the sample?*
  - observation is a sequence of events, one for each token in the sample



$$P(q_1 \ldots q_k \mid M) = \prod_{i=1}^{k} P(q_i \mid M)$$

- Original model is multiple-Bernoulli:
  - fundamental event: *does the word w occur in the sample?*
  - observation is a set of binary events, one for each possible word



$$P(q_1 \ldots q_k \mid M) = \prod_{w \in q_1 \ldots q_k} P(w \mid M) \prod_{w \notin q_1 \ldots q_k} \left[1 - P(w \mid M)\right]$$

Multinomial or multiple-Bernoulli?

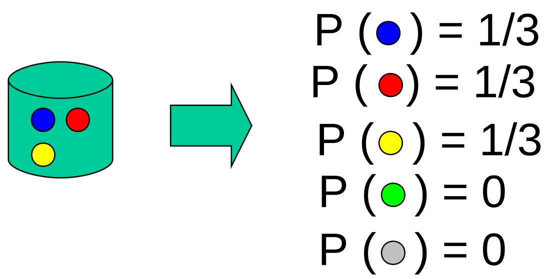
- Two models are fundamentally different
  - entirely different event spaces ("word" means different things)
  - both assume word independence (though it has different meanings)
  - have different estimation methods (though appear very similar)
- Multinomial
  - accounts for multiple word occurrences in the query (primitive)
  - well understood: lots of research in related fields (and now in IR)
  - possibility for integration with ASR/MT/NLP (same event space)
- Multiple-Bernoulli
  - arguably better suited to IR (directly checks presence of query terms)
  - provisions for explicit negation of query terms ("A but not B")
  - no issues with observation length

Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and feedback

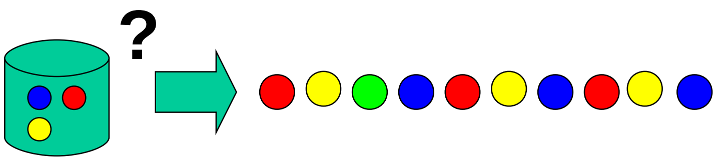
Estimation of Language Models

- Usually we don't know the model **M**
  - but have a sample of text representative of that model
  - estimate a language model from that sample
- Maximum likelihood estimator:
  - count relative frequency of each word



The Zero-frequency Problem

- Suppose some event (word) not in our sample D
  - model will assign zero probability to that event
  - and to any set of events involving the unseen event
- Happens very frequently with language (Zipf)
- It is incorrect to infer zero probabilities
  - especially when dealing with incomplete samples



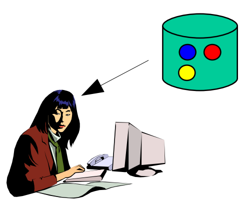
Counts vs. Probabilities

- Have a biased coin: P(“heads” = ***p***)
  - flip a coin several times → get sequence of heads / tails
  - try to recover ***p*** from these observations

●,●,●,●,○,○,●,●,●,●,●,●,○,●,●,●,●,●,●,…

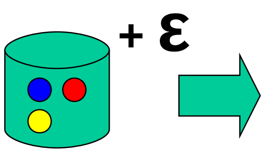
0 / 0, ***p*** = ???	4 / 5, ***p*** = 0.80
1 / 1, ***p*** = 1.00	17 / 20, ***p*** = 0.85
2 / 2, ***p*** = 1.00	72 / 100, ***p*** = 0.72

- Same problem with language-models (*n*-faced coins)
  - document is an observation (word counts)
  - “sampled” from urn with unknown frequencies
    - i.e. contents of the author's mind while writing



Simple Discounting Methods

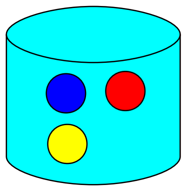
- Laplace correction:
  - add 1 to every count, normalize
  - problematic for large vocabularies
- Lindstone correction:
  - add a small constant ***ε*** to every count, re-normalize
- Absolute Discounting
  - subtract a constant ***ε***, re-distribute the probability mass



P (●) = (1 + ε) / (3+5ε)
(●) = (1 + ε) / (3+5ε)
(●) = (1 + ε) / (3+5ε)
(●) = (0 + ε) / (3+5ε)
(●) = (0 + ε) / (3+5ε)

Good-Turing Estimation

- Leave-one-out discounting
  - remove some word, compute $P(D|M_D)$
  - repeat for every word in the document
  - iteratively adjusting ***ε*** to maximize $P(D|M_D)$
    - increase if word occurs once, decrease if more than once



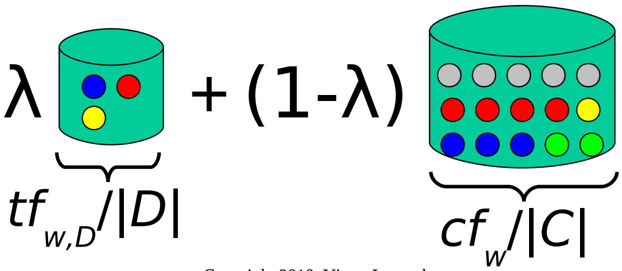
- Good-Turing estimate
  - derived from leave-one-out discounting, but closed-form
  - if a word occurred ***n*** times, its "adjusted" frequency is:

$$n^* = (n+1)\ E\{\#_{n+1}\} / E\{\#_n\}$$

  - probability of that word is: n^* / N^*
  - $E\{\#_n\}$ is the "expected" number of words with ***n*** occurrences
  - $E\{\#_n\}$ very unreliable for high values of ***n***

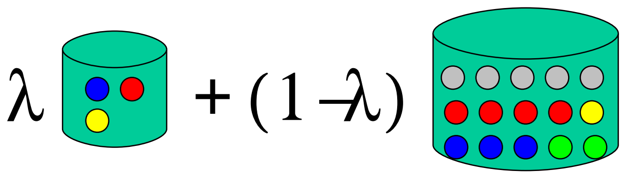
Interpolation Methods

- Problem with all discounting methods:
  - discounting treats unseen words equally (add or subtract $\boldsymbol{\varepsilon}$)
  - some words are more frequent than others
- Idea: use background probabilities
  - "interpolate" ML estimates with General English expectations
  - reflects expected frequency of words in "average" document
  - in IR applications, plays the role of IDF
- 2-state HMM analogy



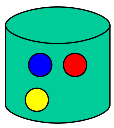
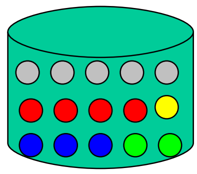
"Jelinek-Mercer" Smoothing

- Correctly setting **λ** is very important
- Start simple:
  - set **λ** to be a constant, independent of document, query
- Tune to optimize retrieval performance
  - optimal value of **λ** varies with different databases, queries, etc.



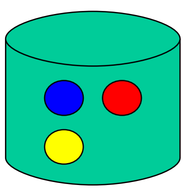
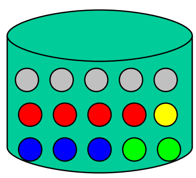
“Dirichlet” Smoothing

- Problem with Jelinek-Mercer:
  - longer documents provide better estimates
  - could get by with less smoothing
- Make smoothing depend on sample size
- Formal derivation from Bayesian (Dirichlet) prior on LMs
- Currently best out-of-the-box choice for short queries
  - parameter tuned to optimize MAP, needs some relevance judgments

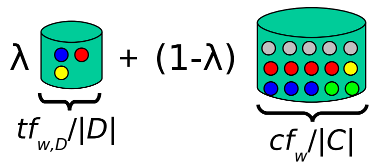
$\underbrace{N / (N + \mu)}_{\lambda}$  $+ \mu \underbrace{/ (N + \mu)}_{(1-\lambda)}$ 

Leave-one-out Smoothing

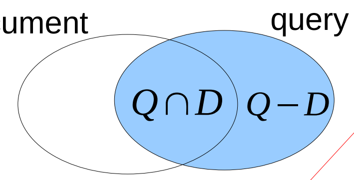
- Re-visit leave-one-out idea:
  - Randomly remove some word from the example
  - Compute the likelihood for the original example, based on **λ**
  - Repeat for every word in the sample
  - Adjust **λ** to maximize the likelihood
- Performs as well as well-tuned Dirichlet smoothing
  - does not require relevance judgments for tuning the parameter

λ  $+ (1 - \lambda)$ 

IDF-like role of smoothing



$$P(Q|D)=\prod_{w\in Q}P(w|D)$$



$$\prod_{w\in Q}\left[(1-\lambda)\frac{cf_w}{|C|}\right]\perp D$$

$$=\prod_{w\in Q\cap D}\left[\lambda\frac{tf_{w,D}}{|D|}+(1-\lambda)\frac{cf_w}{|C|}\right]\prod_{w\in Q-D}\left[(1-\lambda)\frac{cf_w}{|C|}\right]\frac{\prod_{w\in Q\cap D}\left[(1-\lambda)\frac{cf_w}{|C|}\right]}{\prod_{w\in Q\cap D}\left[(1-\lambda)\frac{cf_w}{|C|}\right]}$$

$$\stackrel{rank}{=}\prod_{w\in Q\cap D}\frac{\lambda\frac{tf_{w,D}}{|D|}+(1-\lambda)\frac{cf_w}{|C|}}{(1-\lambda)\frac{cf_w}{|C|}}=\prod_{w\in Q\cap D}\left[1+\frac{\lambda}{1-\lambda}\frac{tf_{w,D}}{|D|}\frac{|C|}{cf_w}\right]$$

LMs: an intuitive view

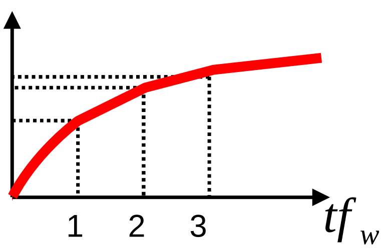
Common words less important

Repetitions of query words ➔ good

$$\log P(Q|D)=\sum_{w\in Q\cap D}\underbrace{\log\left(1+\frac{\lambda_D}{1-\lambda_D}\cdot\frac{tf_{w,D}}{|D|}\cdot\frac{|C|}{cf_w}\right)}$$

More words in common with the query ➔ good

Repetitions less important than different query words



Variations of the LM Framework

- Query-likelihood: $P(Q|M_D)$
  - probability of observing query from the document model $\boldsymbol{M_D}$
  - difficult to incorporate relevance feedback, expansion, operators
- Document-likelihood: $P(D|M_Q)$
  - estimate relevance model $\boldsymbol{M_q}$ using text in the query
  - compute likelihood of observing document as a random sample
  - strong connections to classical probabilistic models: $P(D|R)$
  - ability to incorporate relevance, interaction, query expansion
- Model comparison: $D\ (M_Q \,||\, M_D)$
  - estimate both document and query models
  - measure "divergence" between the two models
  - best of both worlds, but loses pure probabilistic interpretation

Language Models and PRP

- Relevance not explicitly part of LM approach
- [Lafferty & Zhai, 2003]: it's *implicitly* there:
  - PRP: $$P(R=1|D,Q) \overset{rank}{=} \frac{P(R=1|D,Q)}{P(R=0|D,Q)} = \frac{P(D,Q|R=1)\,P(R=1)}{P(D,Q|R=0)\,P(R=0)}$$
  - Bayes' rule, then chain rule: $$.= \frac{P(Q|D,R=1)\,P(D|R=1)\,P(R=1)}{P(Q|D,R=0)\,P(D|R=0)\,P(R=0)}$$
  - Bayes' rule again: $$.= \frac{P(Q|D,R=1)}{P(Q|D,R=0)} \cdot \frac{P(R=1|D)}{P(R=0|D)}$$
  - Assumption: $$.= \frac{P(Q|D,R=1)}{P(Q|R=0)} \cdot \frac{P(R=1|D)}{P(R=0|D)}$$
    - R=1: Q drawn from D (LM)
    - R=0: Q independent of D
    - odds ratio assumed to be 1

$$. \overset{rank}{=} P(Q|D) \cdot \frac{P(R=1|D)}{P(R=0|D)}$$

Summary: Language Modeling

- Formal mathematical model of retrieval
  - based on simple process: sampling query from a document urn
  - assumes word independence, higher-order LMs unsuccessful
  - **cleverly avoids pitfall of the classical probabilistic model**
- At a cost: no notion of relevance in the model
  - relevance feedback / query expansion unnatural
    - "augment the sample" rather than "re-estimate model"
  - can't accommodate phrases, passages, Boolean operators
  - extensions to LM overcome many of these problems
    - query feedback, risk minimization framework, LM+BeliefNet, MRF
- Active area of research

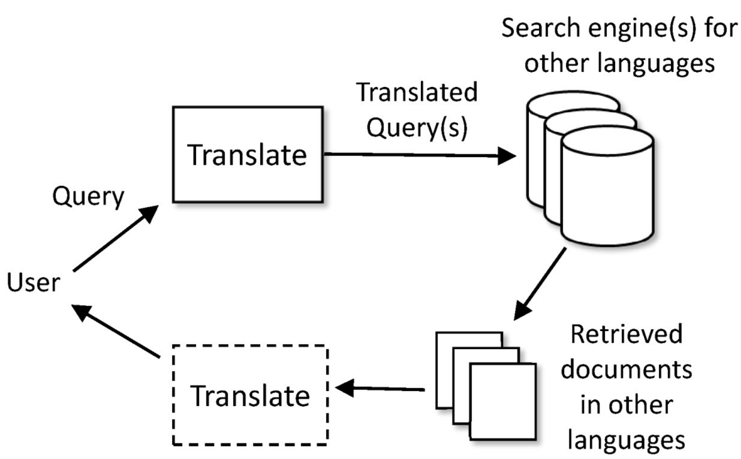
Outline

- Recap of probability theory
- Probability ranking principle
- Classical probabilistic model
  - Binary Independence Model
  - 2-Poisson model and BM25
  - feedback methods
- Language modeling approach
  - overview and design decisions
  - estimation techniques
  - synonymy and cross-language

Cross-language IR

- Cross-language Information Retrieval (CLIR)
  - accept queries / questions in one language (English)
  - find relevant information in a variety of other languages
- Why is this useful?
  - Ex1: research central banks' response to financial crisis
    - dozens of languages, would like to formulate a single query
    - can translate retrieved web-pages into English
  - Ex2: Topic Detection and Tracking (TDT)
    - identify new events (e.g. "5.9 earthquake in El-Salvador on Nov.15")
    - group together all stories discussing the event, regardless of language
    - note: no query to start with
- Good domain to show slightly advanced LMs

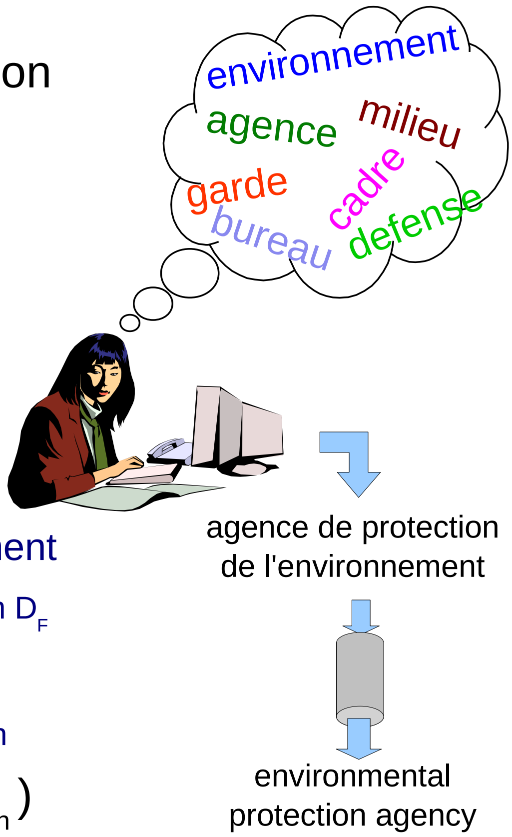
Typical CLIR architecture



Translating the queries

- Translating documents usually infeasible
- Query translation: ambiguous process
  - query as a sentence: may produce odd results
    - not a well-formed utterance, ok for “phrase” queries
  - word-for-word: multiple candidate translations
    - **environment** → environnement, milieu, atmosphere, cadre, conditions
    - **protection** → garde, protection, preservation, defense, racket
    - **agency** → agence, action, organisme, bureau
- How to combine translations?
  - single bag of words: bad idea
  - combinations / hypotheses
    - How many? How to assign weights?

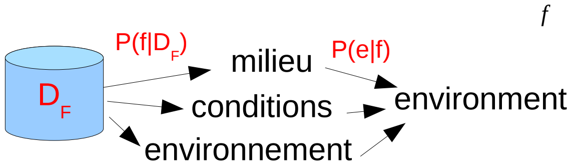
Language modeling view of CLIR



- Don't translate, model query generation
- Metaphor: user is really foreign
  - contemplates relevant documents
  - writes query in the foreign language
  - sends it over a noisy channel
    - query arrives “garbled” into English
- Using metaphor for retrieval:
  - language model for every foreign document
    - what foreign queries could be generated from D_F
  - translation model for the noisy channel
    - how foreign queries are “garbled” into English
- Rank documents by P $(Q_{English} \mid D_{Foreign})$

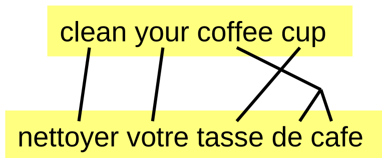
Language modeling approach

- Translation model: set of probabilities P(e|f)
  - probability that French word "f" translates to English word "e"
    - e.g. P("environment" | "milieu") = ¼, P("agency" | "agence") = ½, etc.
- Language model of a French document: $P(f|D_F)$
  - probability of observing "f": $P(\text{milieu}|D_F) = \frac{tf_{\text{milieu},D_F}}{|D_F|}$
- Combine into noisy-channel model:
  - prob. of sampling f and translating to e: $P(e,f|D_F) = P(e|f)\,P(f|D_F)$
  - many different foreign words can translate to e
  - total probability of observing e: $P(e|D_F) = \sum_f P(e|f)\,P(f|D_F)$



Translation probabilities

- How to estimate P(e|f)?
- f → e dictionary: assign equal likelihoods to all translations
  - agence → agency:1/5, bureau:1/5, branch:1/5, office:1/5, service:1/5
- e → f dictionary: use Bayes rule, collection frequency
  - agency → agence:¼, action:¼, organisme:¼, bureau:¼
  - P(agency|agence) = P(agence|agency) * P(agency) / P(agence)
- parallel corpus:
  - set of parallel sentences {E,F} such that E is a translation of F
  - simple co-occurrence: how many times e,f co-occur: $P(e|f) = \frac{|(E,F): e \in E \wedge f \in F|}{|F: f \in F|}$
  - IBM translation model 1:
    - alignment: links between English, French words
    - count how many times e,f are aligned
    - iterative (EM) solution



CLIR: putting it all together

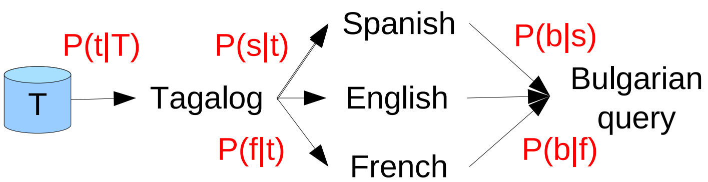
- Rank documents by

$$P(e_{1\ldots k}|D_F)=\prod_{i=1}^{k}\left(\underbrace{\lambda_D\sum_f \overbrace{P(e_i|f)}^{\text{channel}}\,\overbrace{P(f|D_F)}^{\text{source}}+\overbrace{(1-\lambda_D)P(e_i)}^{\text{smoothing}}}_{\text{probability of seeing query word } e_i \text{ during random sampling from } D_F}\right)$$

- Important issues:
  - translation probabilities ignore context
    - one solution: treat phrases as units, but there's a better way
  - vocabulary coverage extremely important
    - use as many dictionaries / lexicons / corpora as possible
  - morphological analysis crucial for Arabic, Slavic, etc.
  - no coverage for proper names → *transliterate:*
    - Qadafi, Kaddafi, Qathafi, Gadafi, Qaddafy, Quadhaffi, al-Qaddafi, ..

Triangulated translation

- Translation models need bilingual resources
  - dictionaries / parallel corpora
  - not available for every language pair (Bulgarian ↔ Tagalog)
- Idea: use resource-rich languages as interlingua:
  - map Tagalog → Spanish, then Spanish → Bulgarian
  - use multiple intermediate languages, assign weights
- Results slightly exceed direct bilingual resource



Summary: CLIR

- Queries in one language, documents in another
  - real task, at least for intelligence analysts
  - translate query to foreign language, retrieve, translate results
- Language modelling approach:
  - probabilistic way to deal with uncertainty in translations
  - effective: 75-95% of mono-lingual performance
  - translation probabilities: based on dictionary, parallel corpus
- Triangulated translation for resource-poor languages
- Translation model: very general idea
  - synonyms: English → English translation

Content-based Image Search

Image Annotation / Retrieval Task

- Given a collection of **un-labeled** images
  - **annotation:** assign relevant keywords to images
  - **retrieval:** find images relevant to a given query
- Learn to associate sets of words with pictures

{ tiger, grass, trees }

Annotation vs. Retrieval

- NOTE: related but not equivalent problems
  - can have good retrieval with bad annotation
    - half the words assigned to each image are wrong
    - 80% of queries (all but “city”) will have perfect precision

city, tiger

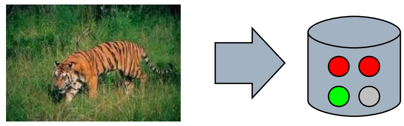
city, iguana

city, bear

city, zebra

Language-modeling Approach

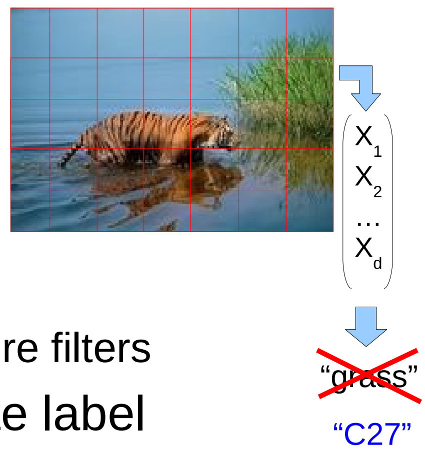
- Query is a bag of words: {tiger,grass,trees}
- Convert every image to a bag of word-like units

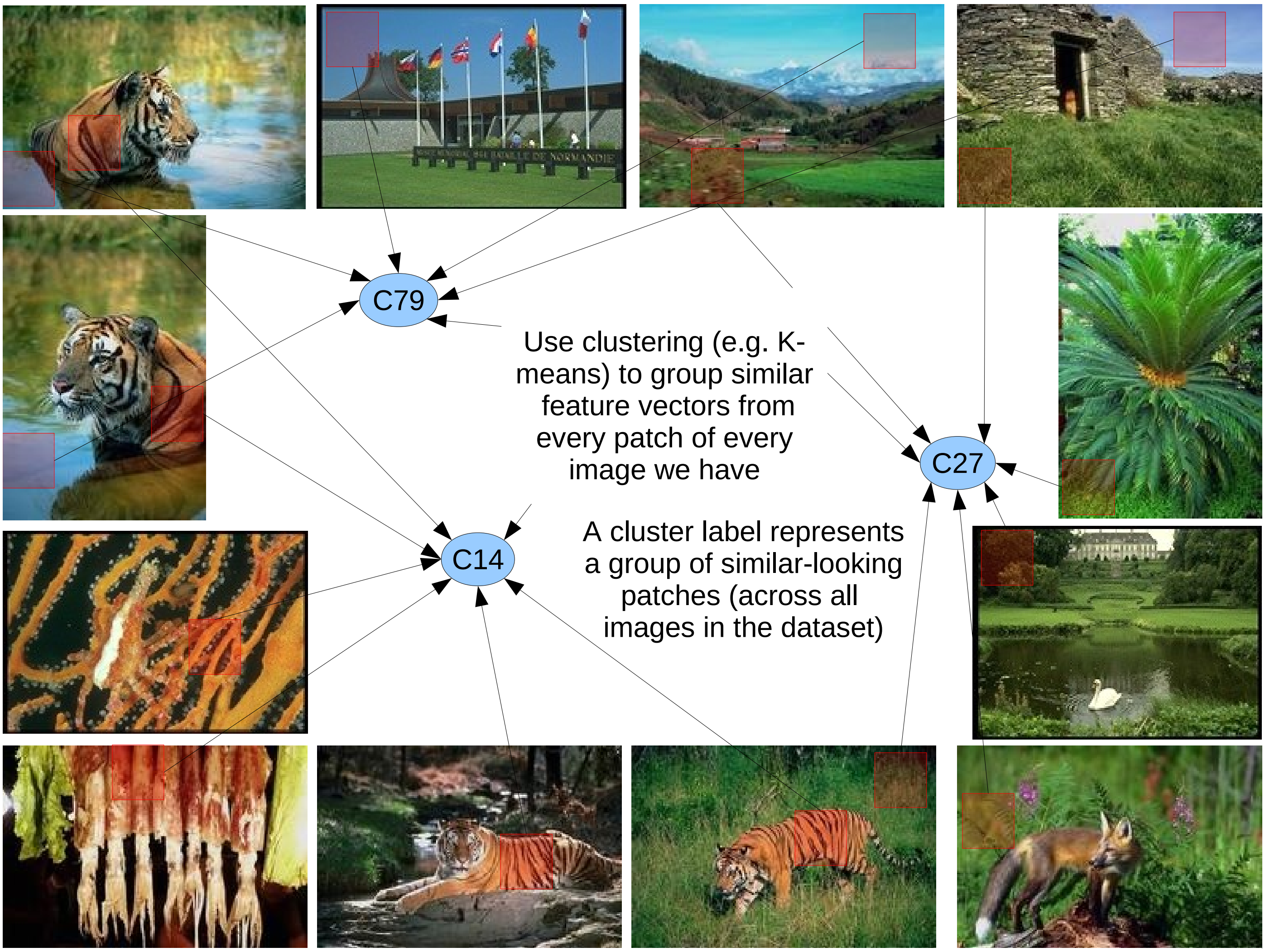


- Reduces to cross-language retrieval problem
  - given a query in English: “tiger grass”
  - match documents written in foreign (visual) “words”
- Main issues:
  - how do we define / compute these visual words?
  - is the cross-language retrieval model sufficient?

Converting Image to “Words”

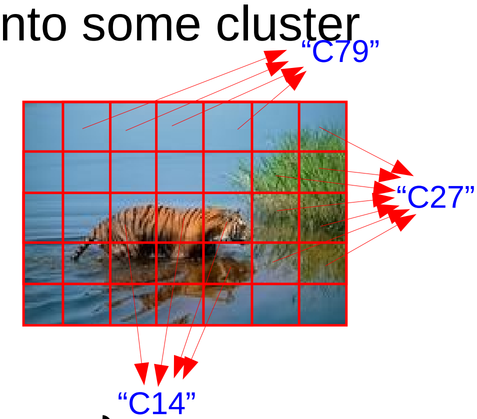
- Convert into a set of discrete “features”
  - break image into a set of patches
    - “grassy”, “watery”, “tigery” patches
    - captures different objects in image
  - extract features for each patch
    - reflect visual appearance of a patch
    - relative position, color histogram, texture filters
  - replace feature vector with a discrete label
    - meaningful label (e.g. “grass”) needs human annotations
    - clustering: group feature vector with other, similar vectors





Cluster Numbers as Visual "Words"

- After clustering:
  - every patch of every image falls into some cluster
    - all similar-looking patches fall into the same cluster
    - cluster id says something about patches that fall into it
      - "C27" → green, vertically-textured



- Use cluster ids as "words"
  - D={ 4 x "C14", 7 x "C27", 24 x "C79", 0 x everything else }
  - similar to controlled vocabulary / category codes
    - discrete, content-bearing, Zipfian distribution
    - sometimes called "vis-terms" or "visual words"

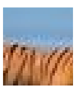
Retrieving Images

- Converted:  ⇨ {4 x "C14", 7 x "C27", 24 x "C79"}
- Want to query with *"tiger"*, not *"C14"*
  - use LMs to "translate" English queries into vis-terms
  - rank images by probability they "generate" query:

$$P(e_{1\ldots k}|I)=\prod_{i=1}^{k}\left(\underbrace{\lambda_I\sum_{v}\overbrace{P(e_i|v)}^{\text{translate to English}}\;\overbrace{P(v|I)}^{\text{draw a visterm}}}_{\text{probability that one of the visterms present in the image "translates" to query word } e_i}+\overbrace{(1-\lambda_I)P(e_i)}^{\text{Smoothing (IDF)}}\right)$$

  - need two components:
    - *P(v|I)* … document model based on counts of vis-terms
    - *P(q|v)* … model for associating words *q* with visterms *v*

Translating Visterms to Words

- No dictionaries
- Parallel corpora (manually-tagged images)
  - e.g.: Corel, Pascal VOC, TRECVid, LabelMe
  - pre-process → get paired sets: $\{\underbrace{v_1\ldots v_n}_{\text{visterms}},\ \underbrace{e_1\ldots e_m}_{\text{tags}}\}$
  - extract translation pairs P(e|v)
    - co-occurrence model (direct count): $P(e|v)=\frac{|I:e\in E_I, v\in V_I|}{|I:v\in V_I|}$
      - problem: will associate "tiger" with C14 **and** C27 **and** C79
    - IBM translation model 1
      - uses EM to align "tiger" → C14, "grass" → C27, etc.
  - problem: visterms don't map to words 1-1
    -  in isolation does not "translate" to anything

Set-to-Set Translation

- Visterms ↔ tags is a set-to-set mapping
  - don't try to break it into pairs: model holistically
  - joint probability of a set of tags w. a set of visterms
    - cross-media relevance model:

$$P(\underbrace{e_1 \ldots e_m}_{\text{query or candidate label}}, \underbrace{v_1 \ldots v_n}_{\text{unlabeled testing image}}) = \underbrace{\sum_{E,V}}_{\text{training images}} \underbrace{\prod_i P(e_i|E)}_{e_1 \ldots e_m \text{ observed in a training image}} \cdot \underbrace{\prod_j P(v_j|V)}_{v_1 \ldots v_n \text{ observed in the same training image}} \cdot \underbrace{P(E,V)}_{\text{prior}}$$

    - note: can't just count: $\{e_1 \ldots e_m, v_1 \ldots v_n\}$
- Annotate with set of tags: $\arg\max_{e_1 \ldots e_m} P(e_1 \ldots e_m, v_1 \ldots v_n)$
- Rank images by: $P(e_1 \ldots e_m | v_1 \ldots v_n) = \dfrac{P(e_1 \ldots e_m, v_1 \ldots v_n)}{P(v_1 \ldots v_n)}$

Summary: CBIR

- Task: associate image content with keywords
  - **annotate** new images with tags automatically
  - **retrieve** unlabeled images using keyword queries
- Convert image to vis-terms
  - segment into patches, group into clusters
  - cluster id = “word” reflecting visual appearance
- Use language models as in CLIR: $P(e_{1..m}|v_{1..n})$
  - translation pairs *P(e|v)* … co-occurrence, EM
  - better way: joint probabilities (relevance model)

Practical Suggestions

Use a Toolkit

- Lemur (C++): www.lemurproject.org
  - use the Indri engine
- Terrier (Java): www.terrier.org
- Zettair (C): www.seg.rmit.edu.au/zettair
- Parallel (experimental):
  - Galago (Java): www.galagosearch.org
    - uses TupleFlow, used in Croft's new textbook
  - Ivory (Java): www.umiacs.umd.edu/~jimmylin/ivory
    - uses Hadoop/Cloud, new project
- Lucene, Xapian, etc.: production, not research

Compute Everything in Log-Space

- IR models have lots of variables (words, docs)
  - independence => products of 1000s of probabilities
    - probabilities are very small numbers (must add up to 1)
  - easy to "overflow" floating point precision:
    - smallest non-zero value: 10^{-38}(single), 10^{-308} (double)
    - overflows after ~1000 words, storing lots of zeroes
  - ratios won't save you:

$$\frac{P(\vec{d}|R=1)}{P(\vec{d}|R=0)} = \underbrace{\prod_v}_{>10^3} \underbrace{\frac{P(d_v|R=1)}{P(d_v|R=0)}}_{<1 \text{ for most } v}$$

- Take log of everything
  - turns 10^{-38} into -38

$$\log P(\vec{d}|R=1) = \log \prod_v P(d_v|R=1) = \sum_v \log P(d_v|R=1)$$

Log-sum-exp Trick

- Your model has a product inside a summation
  - applies to most mixture models
  - how to compute in log-space?

$$\begin{aligned}
\log\left[\sum_a \prod_b P_{a,b}\right] &= \log\left[\sum_a \exp\left(\log \prod_b P_{a,b}\right)\right] \\
&= \log\left[\sum_a \exp\left(\sum_b \log P_{a,b}\right)\right] \\
&= \log\left[\sum_a \exp\left(\sum_b \log P_{a,b} + A - A\right)\right] \\
&= \log\left[\sum_a \exp\left(\sum_b \log P_{a,b} + A\right) e^{-A}\right] \\
&= \log\left[\sum_a \exp\left(\sum_b \log P_{a,b} + A\right)\right] - A
\end{aligned}$$

A … sufficiently large constant

$A = -max_a\left[\sum_b \log P_{a,b}\right]$ preserves top "ranks"

$A = -\frac{1}{n_a}\sum_a \sum_b \log P_{a,b}$ preserves "average"

Compute over the D-Q Overlap

- Models often involve entire doc /qry/vocabulary
  - BIR: $P(\vec{d}|R=1) = \prod_v P(d_v|R=1) = \underbrace{\prod_{v\in d} r_v \prod_{v\notin d}(1-r_v)}_{\text{all words in vocabulary}}$
  - LM: $P(q|d) = \underbrace{\prod_{v\in q} P(v|d)}_{\text{words in qry}}$
- Very expensive to compute for every document
- Doesn't fit the way most toolkits work
  - don't call *Similarity(Q,D)* for every *D* in the corpus
  - retrieval scores computed from inverted indices
  - will pass only terms that occur **both** in *D* and in *Q*

Retrieval with Inverted Indices

- Initialize array to hold all partial scores
- For each query term
  - fetch inverted list from disk
  - update partial score of each document
- Extract result set (non-zero)

Computing over D-Q Overlap

- Re-work to go over overlapping terms

$$
\begin{aligned}
P(\vec{d}|R=1) &= \prod_{v\in d} r_v \times \prod_{v\notin d} (1-r_v) \times \overbrace{\prod_{v\in d} \frac{1-r_v}{1-r_v}}^{=1} \\
&= \prod_{v\in d} r_v \times \prod_{v} (1-r_v) \;/\; \prod_{v\in d} (1-r_v) \\
&= \prod_{v\in d} \frac{r_v}{1-r_v} \times \prod_{v} (1-r_v)
\end{aligned}
$$

- Log:

$$
\begin{aligned}
\log P(\vec{d}|R=1) &= \sum_{v\in d} \log \frac{r_v}{1-r_v} + \sum_{v} \log(1-r_v) \\
&= \vec{d}\cdot\vec{\rho} + \underbrace{\sum_{v} \log(1-r_v)}_{\text{constant: doesn't depend on } d\text{, doesn't affect ranking}}
\end{aligned}
$$

  - inner product of doc and model
  - constants can be pre-computed

dot product

document

"model" vector $\rho_v = \log \frac{r_v}{1-r_v}$

- Note for MapReduce:
  - some constants will be hard to fit into framework

Inconsistent Assumptions

- Common situation: estimate P(*R*|*D*,*Q*)
  - want to condition *R* on two sources of evidence (*D*,*Q*)
    - *R*,*D*,*Q*: relevance/doc/query, from/to/title, video/speech/tag...
  - don't want to condition on "complex" events (*D*,*Q*)
  - assume independence whenever convenient

$$P(R|D,Q) = \frac{P(D,Q|R)\,P(R)}{P(D,Q)}$$ apply Bayes' rule

$$= \frac{P(D|R)\cdot P(Q|R)}{P(D)\cdot P(Q)}\cdot P(R)$$ assume D and Q are independent

$$= \frac{P(R|D)}{P(R)}\cdot\frac{P(R|Q)}{P(R)}\cdot P(R)$$ Bayes' again: $P(D|R) = \frac{P(R|D)\,P(D)}{P(R)}$

$$= \frac{P(R|D)\,P(R|Q)}{P(R)}$$ easy to work with

Data Inconsistency

- Case: users pick doc, query, judge relevance:
  - *d* … one of 10 possible documents
  - *q* ... one of 10 possible queries
  - relevance observed in 10% of all trials
  - … but in 50% of trials with *d*, or with *q*

$$P(q)=P(d)=P(r)=0.1$$

$$P(r|q)=P(r|d)=0.5$$

- Joint events under assumptions: $Q \perp D$ and $Q \perp D|R$

P(*d,q* picked and judged relevant): $P(d,q,r)=P(d,q|r)P(r)$

$$\overset{Q\perp D|R}{=} P(d|r)P(q|r)P(r)$$

$$=\frac{P(r|d)P(d)}{P(r)}\frac{P(r|q)P(q)}{P(r)}P(r)$$

$$=0.5\times0.1\times0.5\times0.1/0.1$$

$$=\boxed{0.025}$$

P(*d,q* picked in the same trial):

$$P(d,q)\overset{Q\perp D}{=}P(d)P(q)$$

$$=0.1\times0.1$$

$$=\boxed{0.010}$$

absurdity

$$\boxed{P(d,q)<P(d,q,r)}$$

Did we pick bad estimates?

Inconsistent Assumptions

- Are the estimates to blame?
  - did we just pick inconsistent P(q), P(d), P(r)?
    - no, this is achievable in practice

- Assumed both $Q \perp D$ and $Q \perp D$ given R
  - this means either R is independent of D

    … or R is independent of Q
  - probably not what you intended
    - defeats the purpose of using both *Q,D* as evidence

Proof: Q⊥D and Q⊥D|R => Q⊥R or D⊥R

Let Q … query
D … document
R … relevance (r or n)

Define: $r_d = P(d|R=r),\ r_q = P(q|R=r),\ p_r = P(R=r)$
$n_d = P(d|R=n),\ n_q = P(q|R=n),\ p_n = P(R=n)$

Assume: Q ⊥ D conditioned on R and Q ⊥ D

$$P(d|r)P(q|r)p_r + P(d|n)P(q|n)p_n = P(d,q) = P(d)P(q)$$

$$r_d r_q p_r + n_d n_q p_n = (r_d p_r + n_d p_n)(r_q p_r + n_q p_n)$$

$$r_d r_q p_r + n_d n_q p_n = r_d r_q p_r^2 + n_d n_q p_n^2 + n_d r_q p_n p_r + r_d n_q p_r p_n$$

$$r_d r_q p_r(1-p_r) + n_d n_q p_n \underbrace{(1-p_n)}_{=p_r} = (n_d r_q + r_d n_q) p_n p_r$$

$$r_d r_q + n_d n_q = n_d r_q + r_d n_q$$

absurdity

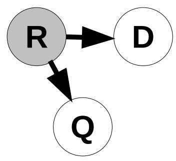
Holds for any values of D,Q
Can extend beyond binary R

$$r_d(r_q - n_q) = n_d(r_q - n_q)$$

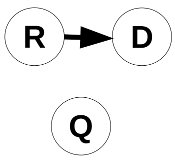
$r_q = n_q \Rightarrow R \perp Q$
or
$r_d = n_d \Rightarrow R \perp D$

Proof (easy version)

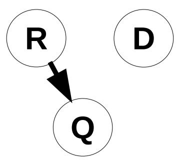
- Assumed both Q ⊥ D and Q ⊥ D given R
- Look at dependence diagrams:
  - conditional independence:
    P(Q,D|R) = P(Q|R) P(D|R)
    path between Q,D goes through R



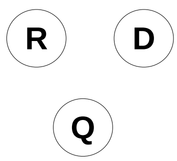
  - mutual independence:
    P(Q,D) = P(Q) P(D)
    no path between Q,D



or



or



    - either no path between R,Q
      … or no path between R,D
- Assuming both breaks dependence on R

What does it mean?

- Can't assume independence left and right
  - make sure assumptions don't contradict each other
- Isn't independence false anyway?
- Yes, but there's a difference:
  - false assumptions:
    - your model poorly fits observed data
  - inconsistent assumptions:
    - you don't have a model at all
      - model violates axioms of probability theory

Checking your Model

- Inconsistency is just one of modelling errors
- Which independence assumptions made
  - do they contradict each other?
- What event spaces are you using
  - are they compatible?
  - what are the possible values of each RV?
- Does the model respect probability axioms?
  - do the marginals add up to 1 over all words / docs?
  - if can't figure out – likely to be a problem

Summary: Practical Suggestions

- Use a toolkit
- Compute everything in log-space
- Log-sum-exp trick
- Compute over document-query overlap
- Check for inconsistencies in the model

References

Probabilistic Relevance Model

[1] S. Robertson. *The probability ranking principle in IR*. Journal of Documentation, 33(4):294-303, 1977.

[2] K. Sparck Jones, S. Walker and S. E. Robertson. *A probabilistic model of information retrieval: development and comparative experiments*. Information Processing and Management, pages 779-840, 2000.

[3] W. B. Croft and D. Harper. *Using probabilistic models of information retrieval without relevance information*. Journal of Documentation, 35(4):285-295, 1979.

[4] V. Lavrenko and W. B. Croft. *Relevance-based language models*. In Proc. 24th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 120-127, 2001.

[5] J. Teevan and D. R. Karger. *Empirical development of an exponential probabilistic model for text retrieval: using textual analysis to build a better model*. In SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, pages 18-25, New York, NY, USA, 2003. ACM.

[6] S. Robertson. *Understanding inverse document frequency: on theoretical arguments for IDF*. Journal of Documentation, 60(5):503-520, 2004.

[7] S. Robertson and S. Walker. *Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval*. In Proc. 17th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 232-241, 1994.

[8] S. E. Robertson and S. Walker. *On relevance weights with little relevance information*. In Proc. 20th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 16-24, 1997.

[9] S. E. Robertson and K. Sparck Jones. *Relevance weighting of search terms*. Journal of the American a Society for Information Science, 27(3):129-146, 1976.

[10] C. J. van Rijsbergen. *A theoretical basis for the use of cooccurrence data in information retrieva*l. Journal of Documentation, 33(2):106-119, 1977.

[11] W. S. Cooper. *Some inconsistencies and misnomers in probabilistic information retrieval*. In Proc. 14th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 57-61, 1991.

[12] P. Domingos and M. Pazzani. *Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier*. Machine Learning, pages 105-112, Morgan Kaufmann, 1996

[13] V. Lavrenko. *A Generative Theory of Relevance*. Pages 23-26, Springer-Verlag, 2009

References

Language Modeling Approach

[14] J. Ponte and W. B. Croft. *A language modeling approach to information retrieval*. In Proc. 21st Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 275-281, 1998.

[15] D. H. Miller, T. Leek, and R. Schwartz. *A hidden markov model information retrieval system*. In Proc. 22nd Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 214-221, 1999.

[16] C. Zhai and J. Lafferty. *A study of smoothing methods for language models applied to ad hoc information retrieval*. In Proc. 24th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 334-342, 2001.

[17] D. Hiemstra. *Using language models for information retrieval*. PhD thesis, University of Twente, Amsterdam, NL, 2001.

[18] J. Lafferty and C. Zhai. *Probabilistic relevance models based on document and query generation*, pages 1-10. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[19] J. Lafferty and C. Zhai. *Document language models, query models, and risk minimization for information retrieval*. In Proc. 24th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 111-119, 2001.

[20] C. Zhai and J. Lafferty. *Two-stage language models for information retrieval*. In Proc. 25th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 49-56, 2002.

[21] D. Metzler, V. Lavrenko, and W. B. Croft. *Formal multiple bernoulli models for language modeling*. In Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 540-541, 2004.

[22] H. Zaragoza, D. Hiemstra, and M. Tipping. *Bayesian extension to the language model for ad-hoc information retrieva*l. In the proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 4-9, Toronto, Canada, July 2003.

[23] T. Tao, X. Wang, Q. Mei, and C. Zhai. *Language model information retrieval with document expansion*. In Proc. of HLT/NAACL, pages 407-414, 2006.

[24] C. Zhai and J. Lafferty. *Model-based feedback in the language modeling approach to information retrieva*l. In Proc. 10th Intl. Conf. on Information and Knowledge Management, pages 403-410, 2001.

[25] K. Collins-Thompson and J. Callan. *Query expansion using random walk models*. In Proc. 14th Intl. Conf. on Information and Knowledge Management, pages 704-711, 2005.

References

Language Models (cont)

[27] J. Gao, J. Nie, G. Wu, and G. Cao. *Dependence language model for information retrieval*. In Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 170-177, 2004.

[28] D. Metzler and W. B. Croft. *A markov random field model for term dependencies*. In Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 472-479, 2005.

[30] O. Kurland and L. Lee. *Corpus structure, language models, and ad hoc information retrieval*. In Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 194-201, 2004.

[31] X. Liu and W. B. Croft. *Cluster-based retrieval using language models*. In Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 186-193, 2004.

Other Probabilistic Models

[32] N. Fuhr. *Probabilistic models in information retrieval*. The Computer Journal, 35(3):243-255, 1992.

[33] G. Amati and C. J. van Rijsbergen. *Probabilistic models of information retrieval based on measuring the divergence from randomness*. ACM Transactions on Information Systems, 20(4):357-389, 2002.

[34] F. Gey. *Inferring probability of relevance using the method of logistic regression*. In Proc. 17th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1994.

[35] T. Hofmann. *Probabilistic latent semantic indexing*. In Proc. 22nd Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 50-57, 1999.

[36] X. Wei and W. B. Croft. *LDA-based document models for ad-hoc retrieval.* In Proc. 29th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 178-185, 2006.

[37] Q. Mei, H. Fang, and C. Zhai. *A study of poisson query generation model for information retrieval.* In Proc. 30th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 319-326, New York, NY, USA, 2007. ACM.

[39] R. Nallapati. *Discriminative models for information retrieval.* In Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 64-71, 2004.

[40] H. Turtle and W. B. Croft. *Evaluation of an inference network-based retrieval model.* ACM Transactions on Information Systems, 9(3):187-222, 1991.

[41] C. T. Yu, C. Buckley, K. Lam, and G. Salton. *A generalized term dependence model in information retrieval.* Technical report, Cornell University, 1983.

References

Cross-language Retrieval

[42] A. Berger and J. Lafferty. *Information retrieval as statistical translation*. In Proc. 22nd Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 222-229, 1999.

[43] J. Xu, R. Weischedel, and C. Nguyen. *Evaluating a probabilistic model for cross-lingual information retrieval*. In Proceedings of the Twenty-Fourth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages 105-110.

[44] V. Lavrenko, M. Choquette, and W. Croft. *Cross-lingual relevance models*. In Proceedings of the Twenty-Fifth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, SIGIR'02, pages 175-182.

[45] L. A. Ballesteros and W. B. Croft. *Resolving ambiguity for cross-language retrieval*. In Proceedings of the Twenty-First Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages 64-71.

[46] T. Gollins and M. Sanderson. *Improving cross language information retrieval with triangulated translation*. In Proceedings of the Twenty-Fourth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages 90-95.

[47] D. Hiemstra and F. de Jong. *Disambiguation strategies for cross-language information retrieval.* In Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libaries, ECDL'99, pages 274-293.

[48] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. *A statistical approach to machine translation.* Computational Linguistics, 16(2):79-85, 1990.

References

Interaction and Feedback

[49] C. Buckley and G. Salton. *Optimization of relevance feedback weights*. In Proc. 18th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 351-357, 1995.

[50] D. Harper and C. J. van Rijsbergen. *An evaluation of feedback in document retrieval using co-occurrence data*. Journal of Documentation, 34(3):189-216, 1978.

[51] D. J. Harper. *Relevance feedback in document retrieval systems*. PhD thesis, University of Cambridge, UK, February 1980. 4

[52] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. *Accurately interpreting clickthrough data as implicit feedback*. In Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 154-161, 2005.

[53] X. Shen and C. Zhai. *Active feedback in ad hoc information retrieval*. In Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pages 59-66, 2005.

[54] R. Nallapati, W. Croft, and J. Allan. *Relevant query feedback in statistical language modeling*. In Proceedings of CIKM 2003 conference, pages 560-563, 2003.

References

Multimedia Retrieval

[55] P. Duygulu, K. Barnard, N. Freitas, and D. Forsyth. *Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary*. In Proceedings of the Seventh European Conference on Computer Vision, pages 97-112, 2002.

[56] J. Jeon, V. Lavrenko, and R. Manmatha. *Automatic image annotation and retrieval using cross-media relevance models.* In In the proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 119-126, Toronto, Canada, August 2003.

[57] V. Lavrenko, R. Manmatha, and J. Jeon. *A model for learning the semantics of pictures.* In Proceedings of the Seventeenth Annual Conference on Neural Information Processing Systems (NIPS), Vancouver, Canada, December 2003.

[58] K. Barnard, P. Duygulu, N. Freitas, D. Forsyth, D. Blei, and M. Jordan. *Matching words and pictures.* Journal of Machine Learning Research, 3:1107-1135, 2003.

[59] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik. *Blobworld: A system for region-based image indexing and retrieval*. In Proceedings of the Third International Conference on Visual Information Systems, pages 509-516, 1999.

[60] D. Metzler and R. Manmatha. *An inference network approach to image retrieval.* In Proc. 3rd Intl. Conf. on Image and Video Retrieval, pages 42-50, 2004.

[61] Y. Mori, H. Takahashi, and R. Oka. *Image-to-word transformation based on dividing and vector quantizing images with words*. In Proceedings of the First International Workshop on Multimedia Intelligent Storage and Retrieval Management MISRM'99, 1999.