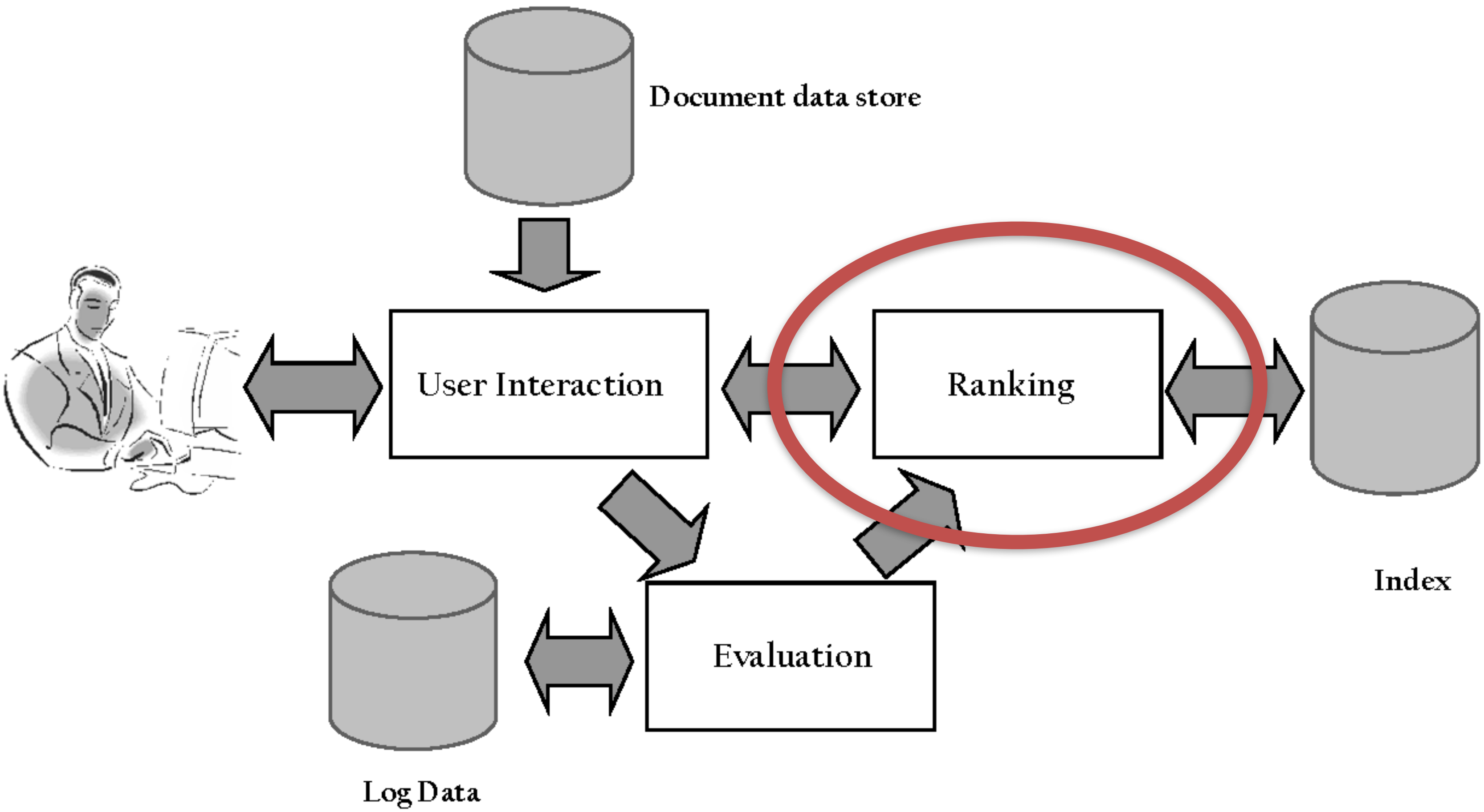
CS6200
Information Retrieval

Jesse Anderton
College of Computer and Information Science
Northeastern University

Query Process



Review: Ranking

- **Ranking** is the process of selecting *which documents* to show the user, and *in what order*
- Rankers are generally developed with a certain **retrieval model** in mind. The retrieval model provides base-line assumptions about what relevance means:
  - ➡ **Boolean Retrieval** models assume a document is entirely relevant or non-relevant, and compose queries using set operations (AND, OR, NOT, XOR, NOR, XNOR).
  - ➡ **Vector Space Models** treat a document or a query as a vector of weights for each vocabulary word, and find document vectors that best match the query's vector.
  - ➡ **Language Models** construct probabilistic models that could generate the text of a query or document, and compare the likelihood that a document and query were generated by the same model.
  - ➡ **Learning to Rank** trains a machine learning algorithm to predict the relevance score for a document based on some fixed set of document features.

Review: Vector Space Models

- **Vector Space Models** treat a document or a query as a vector of weights for each vocabulary word, and find document vectors that best match the query's vector.
- These models consider each term independently of the others, and so do not consider information about noun phrases ("White House") or other important linguistic constructs.
- The main differences between vector space models are in the particular *term weights* and *similarity functions* used.
- The term weight should generally be larger when the term contributes more to the theme of the document.
  - ➡ **TF-IDF** is a heuristic which combines document importance with corpus importance.
  - ➡ **BM25** is a Bayesian formalization of TF-IDF which also considers document length.
- The similarity function should be larger for documents that better satisfy a query's (hidden) information need.
  - ➡ **Cosine Similarity** compares the angles of the vectors while ignoring their magnitude. Matching many high-weight terms leads to a better score.

Language Models

Language Models | Topic Models | Relevance Models
Combining Evidence | Learning to Rank

Language Models

- **Language Models** construct probabilistic models that could generate the text of a query or document, and compare the likelihood that a document and query were generated by the same model.

- These models can handle more complicated linguistic properties, but often take a lot of data and time to train. Often, some training must happen at query time.

- A language model is a function which assigns a probability to a block of text. In IR, you can think of this as the probability that a document is relevant to a query.

  ➡ **Unigram Language Models** estimate the probability of a single word (a "unigram") appearing in a (relevant) document.

  ➡ **N-gram Language Models** assign probabilities to sequences of n words, and so can model phrases. The probability of observing a word depends on the words that came before it.

  ➡ Other language models can model different linguistic properties, such as parts of speech, topics, misspellings, etc.

Language Models in IR

- There are three common techniques for retrieval with language models:

  1. Fit a model to the query and estimate document likelihood:

$$d_1 \prec d_2 \implies Pr(d_1|q) > Pr(d_2|q)$$

  2. Fit a model to the document and estimate query likelihood:

$$d_1 \prec d_2 \implies Pr(q|d_1) > Pr(q|d_2)$$

  3. Jointly model query and document:

$$d_1 \prec d_2 \implies Pr(q, d_1) > Pr(q, d_2)$$

- You can also model topical relevance, as we will discuss later

Ranking by Query Likelihood

- Rank documents based on the likelihood that the model which produced the document could also generate the query.

- Our real goal is to rank by some estimate of $Pr(d|q)$

- To find that, we can apply Bayes' Rule and get:

$$Pr(d|q) \overset{rank}{=} Pr(q|d)Pr(d)$$

- If we assume the prior is uniform (all documents equally likely) and use a unigram model, we get:

$$Pr(q|d)Pr(d) \approx Pr(q|d) = \prod_{q_i \in q} Pr(q_i|d)$$

Estimating Probabilities

- The obvious estimate for term probability is the *maximum likelihood estimate*:

$$Pr(q_i|d) = \frac{tf(q_i, d)}{|d|}$$

- This maximizes the probability of the document by assigning probability to its terms in proportion to their actual occurrence.

- The catch: if $tf(q_i, d) = 0$ for any query term, then

$$\prod_{q_i \in q} Pr(q_i|d) = 0$$

- This takes us back to Boolean Retrieval: missing one term is the same as missing all the terms.

Smoothing our Estimates

- We imagine our document is a sample drawn from a particular language model, and does not perfectly characterize the full sample space.

- Words missing from the document should not have zero probability, and estimates for words found in the document are probably a bit too high.

- *Smoothing* is a process which takes some excess probability from observed words and assigns it to unobserved words.

  ➡ The probability distribution becomes "smoother" – less "spiky."

  ➡ There are many different smoothing techniques.

  ➡ Note that this reduces the likelihood of the observed documents.

Generalized Smoothing

- Most smoothing techniques can be expressed as a linear combination of estimates from the corpus *c* and from a particular document *d*:

$$\hat{Pr}(q_i|d) = (1-\alpha)Pr(q_i|d) + \alpha Pr(q_i|c)$$

- Different smoothing techniques come from different ways of finding the parameter α.

Jelinek-Mercer Smoothing

- In *Jelinek-Mercer Smoothing*, we set α to some constant, λ

$$\hat{Pr}(q_i|d) = (1-\lambda)Pr(q_i|d) + \lambda Pr(q_i|c), \lambda \in [0,1]$$

- This makes our model probability:

$$\hat{Pr}(q_i|d) = (1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}$$

- A document's ranking score is:

$$\hat{Pr}(q|d) = \prod_{q_i \in q}\left((1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}\right)$$

$$\stackrel{rank}{=} \sum_{q_i \in q}\log\left((1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}\right)$$

This is close to TF-IDF!

$$
\begin{aligned}
\log Pr(q|d) &= \sum_{q_i \in q} \log\left((1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}\right) \\
&= \sum_{q_i : tf(q_i,d)>0} \log\left((1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}\right) + \sum_{q_i : tf(q_i,d)=0} \log\left(\lambda\frac{tf(q_i,c)}{|c|}\right) \\
&= \sum_{q_i : tf(q_i,d)>0} \log\left(\frac{(1-\lambda)\frac{tf(q_i,d)}{|d|} + \lambda\frac{tf(q_i,c)}{|c|}}{\lambda\frac{tf(q_i,c)}{|c|}}\right) + \sum_{q_i \in q} \log\left(\lambda\frac{tf(q_i,c)}{|c|}\right) \\
&\overset{rank}{=} \sum_{q_i : tf(q_i,d)>0} \log\left(\frac{(1-\lambda)\frac{tf(q_i,d)}{|d|}}{\lambda\frac{tf(q_i,c)}{|c|}} + 1\right)
\end{aligned}
$$

This ranking score is proportional to TF and inversely proportional to DF.

Dirichlet Smoothing

- In *Dirichlet Smoothing*, we set α based on document length:

$$\alpha = \frac{\mu}{|d| + \mu}$$

- This makes our model probability:

$$Pr(q_i|d) = \frac{tf(q_i, d) + \mu \frac{tf(q_i, c)}{|c|}}{|d| + \mu}$$

- A document's ranking score is:

$$\log Pr(q|d) = \sum_{q_i \in q} \log \frac{tf(q_i, d) + \mu \frac{tf(q_i, c)}{|c|}}{|d| + \mu}$$

Dirichlet Smoothing Example

- Consider the query "president lincoln."
- Suppose that, for some document:

$$tf(\text{“president”}, d) = 15; tf(\text{“president”}, c) = 160000$$

$$tf(\text{“lincoln”}, d) = 25; tf(\text{“lincoln”}, c) = 2400$$

$$|d| = 1800; |c| \approx 10^9$$

$$\mu = 2000$$

- Number of terms in the corpus is based on 2000 terms per document, on average, times 500,000 documents.

Dirichlet Smoothing Example

$$
\begin{aligned}
\log Pr(q|d) &= \sum_{q_i \in q} \log \frac{tf(q_i, d) + \mu \frac{tf(q_i, c)}{|c|}}{|d| + \mu} \\
&= \log \frac{15 + 2000((1.6 \times 10^5)/10^9)}{1800 + 2000} \\
&\quad + \log \frac{25 + 2000(2400/10^9)}{1800 + 2000} \\
&= \log \frac{15.32}{3800} + \log \frac{25.005}{3800} \\
&= -5.51 + -5.02 \\
&= -10.53
\end{aligned}
$$

Dirichlet Smoothing Example

Frequency of "president"	Frequency of "lincoln"	QL Score
15	25	-10.53
15	1	-13.75
15	0	-19.05
1	25	-12.99
0	25	-14.40

Topic Models

Language Models | **Topic Models** | Relevance Models
Combining Evidence | Learning to Rank

Topic Models

- A **topic** can be represented as a language model.
  - ➡ The probability of observing a word depends on the topic being discussed.
  - ➡ Words more strongly associated with a topic will have higher model probabilities.
- A topic model is commonly a multinomial distribution over the vocabulary, conditioned on the topic.
  - ➡ Often works well, but can’t (easily) handle ngrams.

Topic Models

- Interpreting topic models
    - ➡ Improved representation of documents: a document is a collection of topics rather than of words
    - ➡ Improved smoothing: a document becomes relevant to all words related to its topics, whether they appear in the document or not
- Approaches to modeling (latent) topics
    - ➡ Latent Semantic Indexing (LSI) – heuristic, based on decomposition of document term matrix
    - ➡ Probabilistic Latent Semantic Indexing (pLSI) – a probabilistic, generative model based on LSI
    - ➡ Latent Dirichlet Allocation (LDA) – an extension of pLSI which adds a Dirichlet prior to a document's topic distribution

Goals of Topic Modeling

Topic models are applied to manage the following linguistic behaviors:

Text Reuse

Jobless rate at 3-year low as payrolls surge

Recommend 1,328 people recommend this.

Tweet

Share

Share t

+1 49

Email

Print

Related Ne

Instant vi
nonfarm p
rose by 2
Fri, Feb 3 20

Snap ana
creation a
broadly
Fri, Feb 3 20

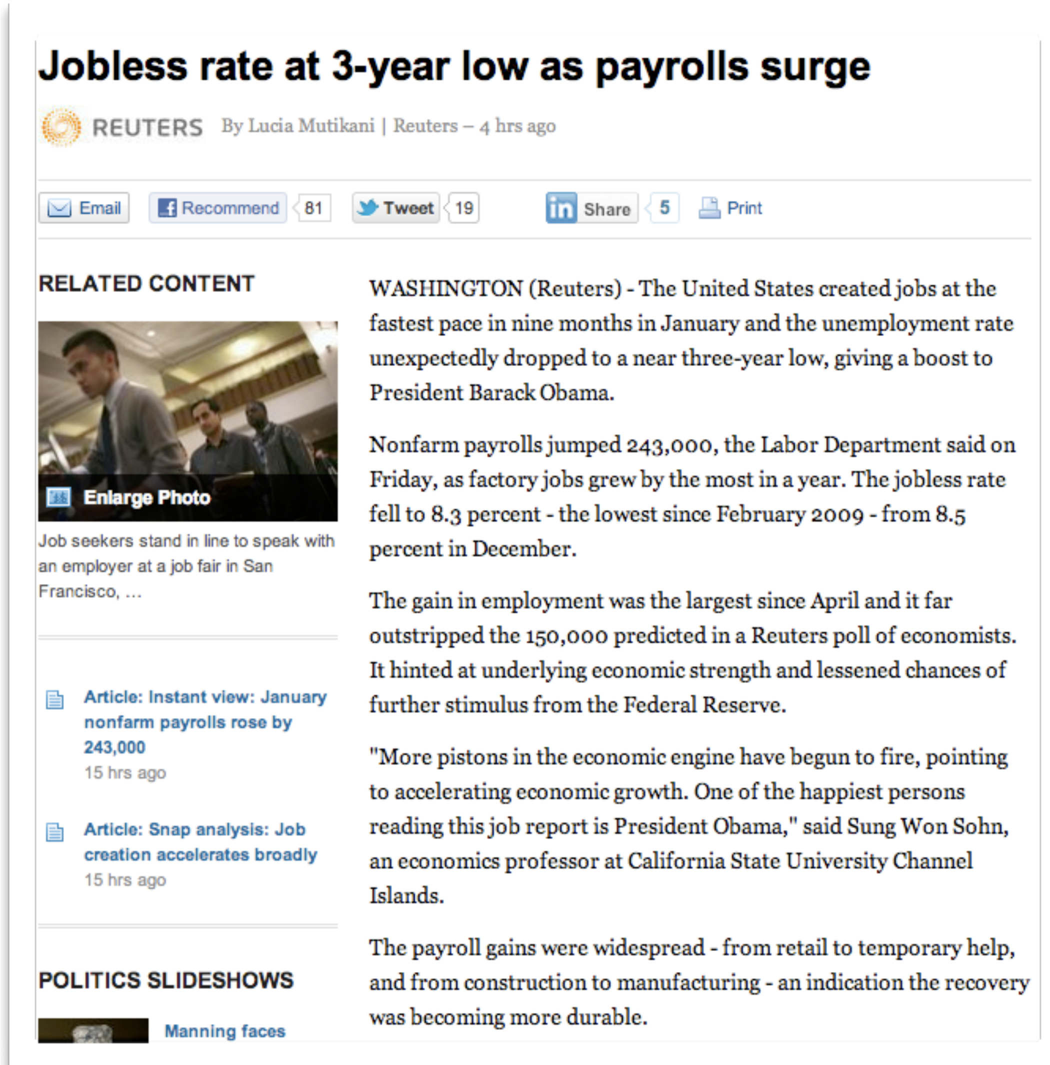
Analysis &

Only 45,000
created in J
TrimTabs

By Lucia Mutikani

WASHINGTON | Fri Feb 3, 2012 5:35pm EST

(Reuters) - The United States created jobs at the fastest pace in nine months in January and the unemployment rate unexpectedly dropped to a near three-year low, giving a boost to President Barack Obama.



Jobless rate at 3-year low as payrolls surge

REUTERS By Lucia Mutikani | Reuters – 4 hrs ago

Email Recommend 81 Tweet 19 Share 5 Print

RELATED CONTENT

Enlarge Photo

Job seekers stand in line to speak with an employer at a job fair in San Francisco, ...

Article: Instant view: January nonfarm payrolls rose by 243,000
15 hrs ago

Article: Snap analysis: Job creation accelerates broadly
15 hrs ago

POLITICS SLIDESHOWS

Manning faces

WASHINGTON (Reuters) - The United States created jobs at the fastest pace in nine months in January and the unemployment rate unexpectedly dropped to a near three-year low, giving a boost to President Barack Obama.

Nonfarm payrolls jumped 243,000, the Labor Department said on Friday, as factory jobs grew by the most in a year. The jobless rate fell to 8.3 percent - the lowest since February 2009 - from 8.5 percent in December.

The gain in employment was the largest since April and it far outstripped the 150,000 predicted in a Reuters poll of economists. It hinted at underlying economic strength and lessened chances of further stimulus from the Federal Reserve.

"More pistons in the economic engine have begun to fire, pointing to accelerating economic growth. One of the happiest persons reading this job report is President Obama," said Sung Won Sohn, an economics professor at California State University Channel Islands.

The payroll gains were widespread - from retail to temporary help, and from construction to manufacturing - an indication the recovery was becoming more durable.

Topical Similarity

Job Gains Reflect Hope a Recovery Is Blooming

Joan Barnett Lee/The Modesto Bee, via Associated Press

A job applicant received assistance at an employment fair in Modesto, Calif., this week.

By MOTOKO RICH
Published: February 3, 2012

The front wheels have lifted off the runway. Now, Americans are waiting to see if the economy can truly get aloft.

RECOMMEND
TWITTER
LINKEDIN
COMMENTS (576)
SIGN IN TO E-MAIL
PRINT
REPRINTS
SHARE

Multimedia

Jobs | Private | Rate

Change in jobs, in thousands

With the government reporting that the unemployment rate and the number of jobless fell in January to the lowest levels since early 2009, the recovery seems finally to be reaching American workers.

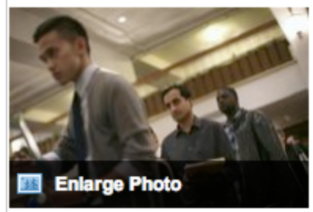
The Labor Department's latest

Jobless rate at 3-year low as payrolls surge

REUTERS By Lucia Mutikani | Reuters – 4 hrs ago

Email | Recommend 81 | Tweet 19 | Share 5 | Print

RELATED CONTENT



Enlarge Photo

Job seekers stand in line to speak with an employer at a job fair in San Francisco, ...

Article: Instant view: January nonfarm payrolls rose by 243,000
15 hrs ago

Article: Snap analysis: Job creation accelerates broadly
15 hrs ago

POLITICS SLIDESHOWS

Manning faces

WASHINGTON (Reuters) - The United States created jobs at the fastest pace in nine months in January and the unemployment rate unexpectedly dropped to a near three-year low, giving a boost to President Barack Obama.

Nonfarm payrolls jumped 243,000, the Labor Department said on Friday, as factory jobs grew by the most in a year. The jobless rate fell to 8.3 percent - the lowest since February 2009 - from 8.5 percent in December.

The gain in employment was the largest since April and it far outstripped the 150,000 predicted in a Reuters poll of economists. It hinted at underlying economic strength and lessened chances of further stimulus from the Federal Reserve.

"More pistons in the economic engine have begun to fire, pointing to accelerating economic growth. One of the happiest persons reading this job report is President Obama," said Sung Won Sohn, an economics professor at California State University Channel Islands.

The payroll gains were widespread - from retail to temporary help, and from construction to manufacturing - an indication the recovery was becoming more durable.

Parallel Bitext

Genehmigung des Protokolls
Das Protokoll der Sitzung vom Donnerstag, den 28. März 1996 wurde verteilt.
Gibt es Einwände?
Die Punkte 3 und 4 widersprechen sich jetzt, obwohl es bei der Abstimmung anders aussah.
Das muß ich erst einmal klären, Frau Oomen-Ruijten.

Approval of the minutes
The minutes of the sitting of Thursday, 28 March 1996 have been distributed.
Are there any comments?
Points 3 and 4 now contradict one another whereas the voting showed otherwise.
I will have to look into that, Mrs Oomen-Ruijten.

Koehn (2005): European Parliament corpus

Multilingual Topic Similarity

Abraham Lincoln

From Wikipedia, the free encyclopedia

This article is about the American president. For other uses, see Abraham Lincoln (disambiguation).

Abraham Lincoln /ˈeɪbrəhæm ˈlɪŋkən/ (February 12, 1809 – April 15, 1865) was the 16th President of the United States, serving from March 1861 until his assassination in April 1865. He successfully led his country through a great constitutional, military and moral crisis – the American Civil War – preserving the Union, while ending slavery, and promoting economic and financial modernization. Reared in a poor family on the western frontier, Lincoln was mostly self-educated. He became a country lawyer, an Illinois state legislator, and a one-term member of the United States House of Representatives, but failed in two attempts to be elected to the United States Senate.

Abraham Lincoln

Abraham Lincoln [ˈeɪbrəhæm ˈliŋkən] (* 12. Februar 1809 bei Hodgenville, Hardin County, heute: LaRue County, Kentucky; † 15. April 1865 in Washington, D.C.) amtierte von 1861 bis 1865 als 16. Präsident der Vereinigten Staaten von Amerika. Er war der erste aus den Reihen der Republikanischen Partei und der erste, der einem Attentat zum Opfer fiel. 1860 gewählt, gelang ihm 1864 die Wiederwahl.

Seine Präsidentschaft gilt als eine der bedeutendsten in der Geschichte der Vereinigten Staaten: Die Wahl des Sklavereigegners veranlasste zunächst sieben, später weitere vier der sklavenhaltenden Südstaaten zur Sezession. Lincoln führte die verbliebenen Nordstaaten durch den daraus entstandenen Bürgerkrieg, setzte die Wiederherstellung der Union durch und betrieb erfolgreich die Abschaffung der Sklaverei in den Vereinigten Staaten. Unter seiner Regierung schlugen die USA den Weg zum zentral regierten, modernen Industriestaat ein und schufen so die Basis für ihren Aufstieg zur Weltmacht im 20. Jahrhundert.

How do we represent topics?

- Bag of words? Ngrams?

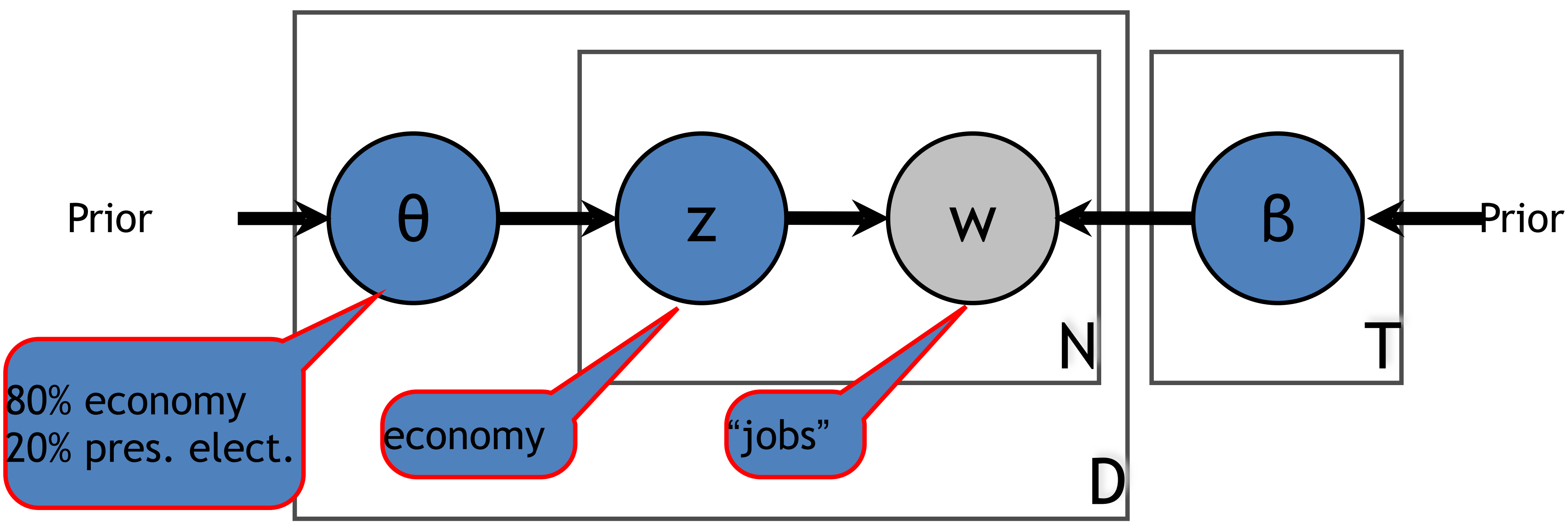
  ➡ Problem: there is a lot of vocabulary mismatch for a topic within a language (jobless vs. unemployed)

  ➡ The problem is even worse between languages. Do we need to translate everything to English first?

- Topic modeling represents documents as probability distributions over hidden (“latent”) topics.

Modeling Text with Topics

- Most modern topic models extend Latent Dirichlet Allocation (Blei, Ng, Jordan 2003)
- The corpus is presumed to contain T topics
- Each topic is a probability distribution over the entire vocabulary
- For D documents, each with N_D words:



Top Words By Topic

Topics →

1	2	3	4	5	6	7	8
DISEASE	WATER	MIND	STORY	FIELD	SCIENCE	BALL	JOB
BACTERIA	FISH	WORLD	STORIES	MAGNETIC	STUDY	GAME	WORK
DISEASES	SEA	DREAM	TELL	MAGNET	SCIENTISTS	TEAM	JOBS
GERMS	SWIM	DREAMS	CHARACTER	WIRE	SCIENTIFIC	FOOTBALL	CAREER
FEVER	SWIMMING	THOUGHT	CHARACTERS	NEEDLE	KNOWLEDGE	BASEBALL	EXPERIENCE
CAUSE	POOL	IMAGINATION	AUTHOR	CURRENT	WORK	PLAYERS	EMPLOYMENT
CAUSED	LIKE	MOMENT	READ	COIL	RESEARCH	PLAY	OPPORTUNITIES
SPREAD	SHELL	THOUGHTS	TOLD	POLES	CHEMISTRY	FIELD	WORKING
VIRUSES	SHARK	OWN	SETTING	IRON	TECHNOLOGY	PLAYER	TRAINING
INFECTION	TANK	REAL	TALES	COMPASS	MANY	BASKETBALL	SKILLS
VIRUS	SHELLS	LIFE	PLOT	LINES	MATHEMATICS	COACH	CAREERS
MICROORGANISMS	SHARKS	IMAGINE	TELLING	CORE	BIOLOGY	PLAYED	POSITIONS
PERSON	DIVING	SENSE	SHORT	ELECTRIC	FIELD	PLAYING	FIND
INFECTIOUS	DOLPHINS	CONSCIOUSNESS	FICTION	DIRECTION	PHYSICS	HIT	POSITION
COMMON	SWAM	STRANGE	ACTION	FORCE	LABORATORY	TENNIS	FIELD
CAUSING	LONG	FEELING	TRUE	MAGNETS	STUDIES	TEAMS	OCCUPATIONS
SMALLPOX	SEAL	WHOLE	EVENTS	BE	WORLD	GAMES	REQUIRE
BODY	DIVE	BEING	TELLS	MAGNETISM	SCIENTIST	SPORTS	OPPORTUNITY
INFECTIONS	DOLPHIN	MIGHT	TALE	POLE	STUDYING	BAT	EARN
CERTAIN	UNDERWATER	HOPE	NOVEL	INDUCED	SCIENCES	TERRY	ABLE

Griffiths et al.

Top Words By Topic

Topics →

1	2	3	4	5	6	7	8
DISEASE	WATER	MIND	STORY	**FIELD**	SCIENCE	BALL	JOB
BACTERIA	FISH	WORLD	STORIES	MAGNETIC	STUDY	GAME	WORK
DISEASES	SEA	DREAM	TELL	MAGNET	SCIENTISTS	TEAM	JOBS
GERMS	SWIM	DREAMS	CHARACTER	WIRE	SCIENTIFIC	FOOTBALL	CAREER
FEVER	SWIMMING	THOUGHT	CHARACTERS	NEEDLE	KNOWLEDGE	BASEBALL	EXPERIENCE
CAUSE	POOL	IMAGINATION	AUTHOR	CURRENT	WORK	PLAYERS	EMPLOYMENT
CAUSED	LIKE	MOMENT	READ	COIL	RESEARCH	PLAY	OPPORTUNITIES
SPREAD	SHELL	THOUGHTS	TOLD	POLES	CHEMISTRY	**FIELD**	WORKING
VIRUSES	SHARK	OWN	SETTING	IRON	TECHNOLOGY	PLAYER	TRAINING
INFECTION	TANK	REAL	TALES	COMPASS	MANY	BASKETBALL	SKILLS
VIRUS	SHELLS	LIFE	PLOT	LINES	MATHEMATICS	COACH	CAREERS
MICROORGANISMS	SHARKS	IMAGINE	TELLING	CORE	BIOLOGY	PLAYED	POSITIONS
PERSON	DIVING	SENSE	SHORT	ELECTRIC	**FIELD**	PLAYING	FIND
INFECTIOUS	DOLPHINS	CONSCIOUSNESS	FICTION	DIRECTION	PHYSICS	HIT	POSITION
COMMON	SWAM	STRANGE	ACTION	FORCE	LABORATORY	TENNIS	**FIELD**
CAUSING	LONG	FEELING	TRUE	MAGNETS	STUDIES	TEAMS	OCCUPATIONS
SMALLPOX	SEAL	WHOLE	EVENTS	BE	WORLD	GAMES	REQUIRE
BODY	DIVE	BEING	TELLS	MAGNETISM	SCIENTIST	SPORTS	OPPORTUNITY
INFECTIONS	DOLPHIN	MIGHT	TALE	POLE	STUDYING	BAT	EARN
CERTAIN	UNDERWATER	HOPE	NOVEL	INDUCED	SCIENCES	TERRY	ABLE

Griffiths et al.

LDA

A document is modeled as being generated from a mixture of topics:

1. For each document D, pick a multinomial distribution θ_D from a Dirichlet distribution with parameter α ,
2. For each word position in document D,
   (a) pick a topic z from the multinomial distribution θ_D ,
   (b) Choose a word w from $P(w|z, \beta)$, a multinomial probability conditioned on the topic z with parameter β.

LDA

- Gives language model probabilities

$$Pr_{lda}(w|D) = Pr(w|\theta_D, \beta) = \sum_z Pr(w|z, \beta)Pr(z|\theta_D)$$

- Can be used to smooth the document representation by mixing them with the query likelihood probability, as follows:

$$Pr(w|D) = \lambda \left(\frac{tf(w, D) + \mu \frac{tf(w,C)}{|C|}}{|D| + \mu} \right) + (1 - \lambda)Pr_{lda}(w|D)$$

LDA

- If the LDA probabilities are used directly as the document representation, the effectiveness will be significantly reduced because the features are *too smoothed*
  - ➡ In a typical TREC experiment, only 400 topics are used for the entire collection
  - ➡ Generating LDA topics and fitting them to documents is expensive
- However, when used for smoothing the ranking effectiveness is improved

LDA Example

- If the LDA probabilities are used directly as the document representation, the effectiveness will be significantly reduced because the features are *too smoothed*
  - ➡ In a typical TREC experiment, only 400 topics are used for the entire collection
  - ➡ Generating LDA topics and fitting them to documents is expensive
- However, when used for smoothing the ranking effectiveness is improved

LDA Example

Top words from 4 LDA topics from a TREC news corpus:

Arts	*Budgets*	*Children*	*Education*
new	million	children	school
film	tax	women	students
show	program	people	schools
music	budget	child	education
movie	billion	years	teachers
play	federal	families	high
musical	year	work	public
best	spending	parents	teacher
actor	new	says	bennett
first	state	family	manigat
york	plan	welfare	namphy
opera	money	men	state
theater	programs	percent	president
actress	government	care	elementary
love	congress	life	haiti

Relevance Models

Relevance Models

- A **relevance model** is a language model representing the user's information need
  - ➡ The query and the relevant documents are considered samples from this model
- The probability of generating the text in a document given a relevance model is denoted $Pr(D|R)$
  - ➡ This is a *document likelihood* model
  - ➡ Less effective than *query likelihood* due to difficulties comparing across documents of different lengths

Pseudo-Relevance Feedback

- Fit a relevance model to a query and the top-ranked documents
- Then rank documents by the similarity between their document models and the relevance model
- The two models can be compared using **Kullback-Leibler divergence** (KL-divergence), an information theoretic measure which gives the difference between two probability distributions

KL-Divergence

- Given a *true* probability distribution P, how close is some *approximation* Q of that distribution?

$$KL(P\|Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

  ➡ This is not symmetric! $KL(P\|Q) \neq KL(Q\|P)$

- For pseudo-relevance feedback:
  ➡ P is the relevance model R
  ➡ Q is the document's distribution
  ➡ We rank documents by their (negative) KL-divergence

$$\sum_{w \in V} P(w|R) \log P(w|D) - \sum_{w \in V} P(w|R) \log P(w|R)$$

KL-Divergence

- If we use a maximum likelihood unigram language model for the relevance model, the ranking score is:

$$\sum_{w \in V} \frac{tf(w, Q)}{|Q|} \log P(w|D)$$

- This is rank-equivalent to the query likelihood score.
- The query likelihood model is a special case of retrieval based on a relevance model.

Estimating the Relevance Model

- The probability of pulling word w out of the "bucket" representing the relevance model depends on the n query words we have just pulled out:

$$Pr(w|R) \approx Pr(w|q_1, \ldots, q_n)$$

- By definition,

$$Pr(w|R) \approx \frac{Pr(w, q_1, \ldots, q_n)}{Pr(q_1, \ldots, q_n)}$$

Estimating the Relevance Model

- The joint probability is:

$$Pr(w, q_1, \ldots, q_n) = \sum_{D \in \mathcal{C}} Pr(D) Pr(w, q_1, \ldots, q_n | D)$$

- If we assume:

$$Pr(w, q_1, \ldots, q_n | D) = Pr(w|D) \prod_{q_i \in Q} Pr(q_i|D)$$

- That gives:

$$Pr(w, q_1, \ldots, q_n) = \sum_{D \in \mathcal{C}} \left(Pr(D) Pr(w|D) \prod_{q_i \in Q} Pr(q_i|D) \right)$$

Interpreting the Relevance Model

- $Pr(D)$ is usually assumed to be uniform
- $Pr(w, q_1, \ldots, q_n)$ is a weighted average of the language model probabilities for w in a set of documents
  - ➡ The weights are the query likelihood scores for those documents
- This gives a formal model for pseudo-relevance feedback
- This also gives a query expansion technique

Pseudo-Feedback Algorithm

1. Rank documents using the query likelihood score for query Q.
2. Select some number of the top-ranked documents to be the set $\mathcal{C}$.
3. Calculate the relevance model probabilities $P(w|R)$. $P(q_1 \ldots q_n)$ is used as a normalizing constant and is calculated as

$$P(q_1 \ldots q_n) = \sum_{w \in V} P(w, q_1 \ldots q_n)$$

4. Rank documents again using the KL-divergence score

$$\sum_{w} P(w|R) \log P(w|D)$$

Example from 10 Docs

president lincoln	*abraham lincoln*	*fishing*	*tropical fish*
lincoln	lincoln	fish	fish
president	america	farm	tropic
room	president	salmon	japan
bedroom	faith	new	aquarium
house	guest	wild	water
white	abraham	water	species
america	new	caught	aquatic
guest	room	catch	fair
serve	christian	tag	china
bed	history	time	coral
washington	public	eat	source
old	bedroom	raise	tank
office	war	city	reef
war	politics	people	animal
long	old	fishermen	tarpon
abraham	national	boat	fishery

Example from Top 50 Docs

president lincoln	*abraham lincoln*	*fishing*	*tropical fish*
lincoln	lincoln	fish	fish
president	president	water	tropic
america	america	catch	water
new	abraham	reef	storm
national	war	fishermen	species
great	man	river	boat
white	civil	new	sea
war	new	year	river
washington	history	time	country
clinton	two	bass	tuna
house	room	boat	world
history	booth	world	million
time	time	farm	state
center	politics	angle	time
kennedy	public	fly	japan
room	guest	trout	mile

Combining Evidence

Language Models | Topic Models | Relevance Models
Combining Evidence | Learning to Rank

Combining Evidence

- No single ranking score has been found which produces satisfactory performance for all queries.

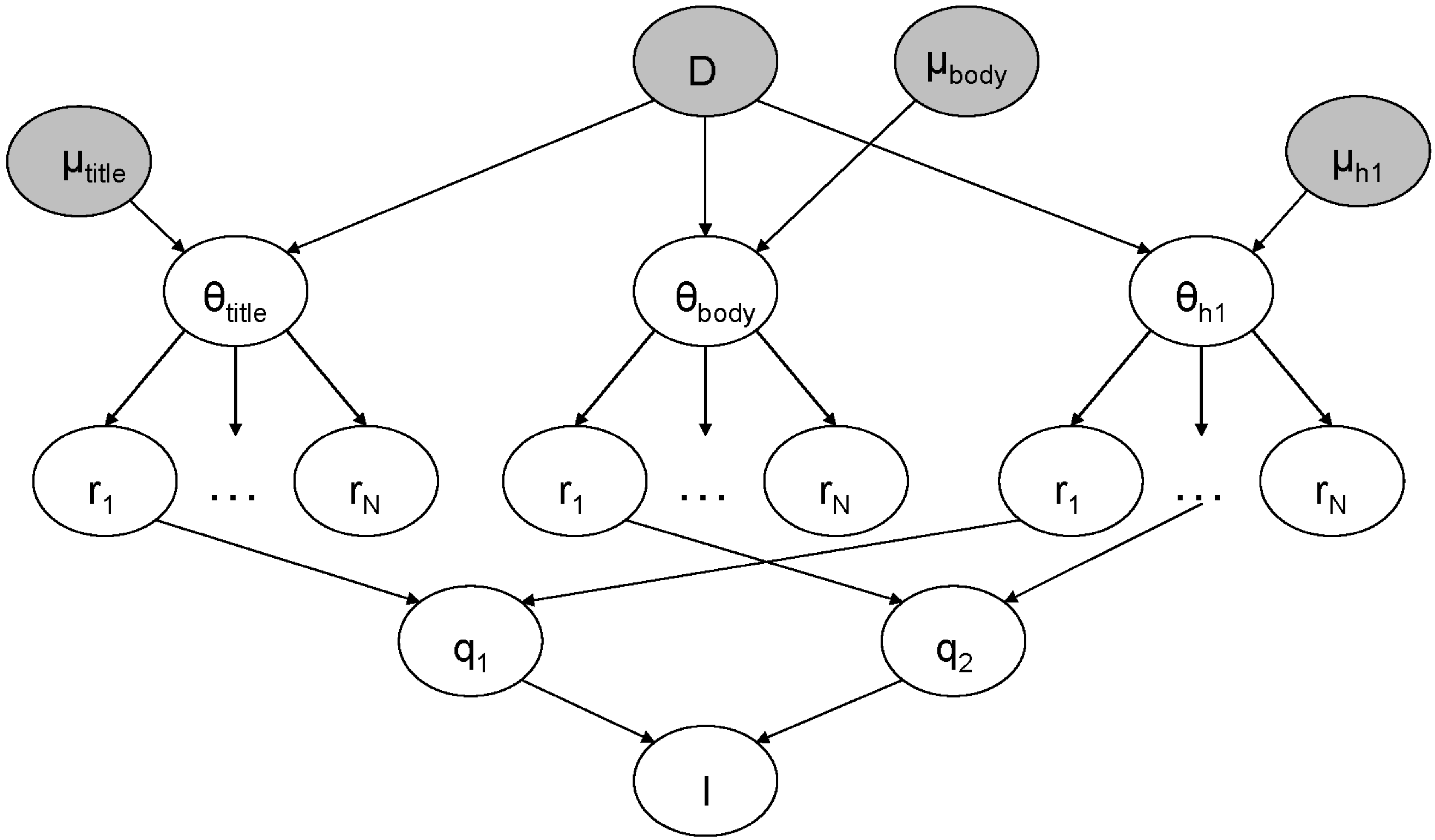
- Effective retrieval requires combining many pieces of evidence about a document's potential relevance.

  - ➡ We have focused so far on simple word-based evidence

  - ➡ There are many other types: document structure, PageRank, metadata, even scores from multiple relevance models

- An **inference network** is one approach for combining this evidence, based on Bayesian networks (aka Bayes Nets)

Inference Network



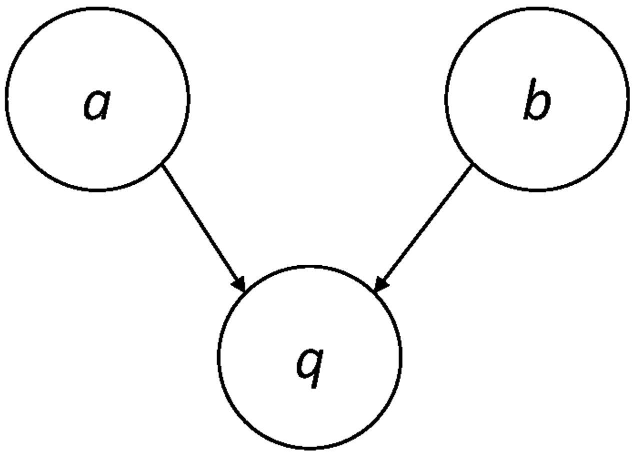
Inference Network

- A document node (D) represents the random event that a document is observed
- Representation nodes (r_i) are document features (evidence)
  - ➡ The probabilities associated with those features are based on language models θ estimated using parameters μ
  - ➡ We train one language model for each significant document feature/structure
  - ➡ The r_i nodes can represent proximity features or other types of evidence (e.g. date)

Inference Network

- Query nodes (q_i) are used to combine evidence from representation nodes and other query nodes.
    - ➡ They represent the occurrence of more complex evidence and document features.
    - ➡ A number of combination operators are available.
- The information need node (I) is a special query node that combines all of the evidence from the other query nodes.
    - ➡ The network computes $Pr(I|D, \mu)$

Example: AND Combination



a and *b* are *parent* nodes for q

$P(q = \text{TRUE} \mid a, b)$	a	b
0	FALSE	FALSE
0	FALSE	TRUE
0	TRUE	FALSE
1	TRUE	TRUE

Example: AND Combination

- Combination operators must compute all possible states of all their parents.
- Some combinations can be computed efficiently.

$$
\begin{aligned}
bel_{and}(q) &= p_{00}P(a = \text{FALSE})P(b = \text{FALSE}) \\
&\quad + p_{01}P(a = \text{FALSE})P(b = \text{TRUE}) \\
&\quad + p_{10}P(a = \text{TRUE})P(b = \text{FALSE}) \\
&\quad + p_{11}P(a = \text{TRUE})P(b = \text{TRUE}) \\
&= 0 \cdot (1 - p_a)(1 - p_b) + 0 \cdot (1 - p_a)p_b + 0 \cdot p_a(1 - p_b) + 1 \cdot p_a p_b \\
&= p_a p_b
\end{aligned}
$$

Inference Network Operators

$$
\begin{aligned}
bel_{not}(q) &= 1 - p_1 \\
bel_{or}(q) &= 1 - \prod_{i}^{n}(1 - p_i) \\
bel_{and}(q) &= \prod_{i}^{n} p_i \\
bel_{wand}(q) &= \prod_{i}^{n} p_i^{wt_i} \\
bel_{max}(q) &= max\{p_1, p_2, \ldots, p_n\} \\
bel_{sum}(q) &= \frac{\sum_i^n p_i}{n} \\
bel_{wsum}(q) &= \frac{\sum_i^n wt_i p_i}{\sum_i^n wt_i}
\end{aligned}
$$

Web Search

- The most important, but not the only, search application
- Has major differences as compared with research applications, such as TREC news:
  - ➡ Collection size
  - ➡ Connections between documents
  - ➡ Range of document types
  - ➡ The importance of spam
  - ➡ Query volume
  - ➡ Range of query types

Search Taxonomy

- **Informational** Queries
  - ➡ Finding information about some topic which may be found on one or more web pages
  - ➡ Topical search
- **Navigational** (“Page Finding”) Queries
  - ➡ Finding a particular web page that the user has either seen before, or assumes to exist
- **Transactional** (“e-commerce”) Queries
  - ➡ Finding a site where a task such as shopping or downloading music can be performed

Web Search

- For effective navigational and transactional search, need to combine features that reflect *user relevance*.

- Commercial web search engines combine evidence from *hundreds* of features to generate a ranking score for each web page.

  ➡ Page content, page metadata, anchor text, links (e.g. PageRank), and user behavior (click logs)

  ➡ Page metadata – e.g. “age,” how often it is updated, the URL of the page, the domain name of its site, and the amount of text content

Search Engine Optimization

- **SEO**: Understanding the relative importance of the many features used in search and how they can be manipulated to obtain better search rankings for a web page

    ➡ e.g., improve the text used in the title tag, improve the text in heading tags, make sure that the domain name and URL contain important keywords, and try to improve the anchor text and link structure

    ➡ Some of these techniques are regarded as not appropriate by search engine companies

Web Search

- In TREC evaluations, the most effective features for navigational search are:
  - ➡ Text in the title, body, and heading (h1, h2, h3, and h4), the anchor text of all links pointing to the document, the PageRank number, and the in-link count
- Given the size of Web, many pages will contain all query terms
  - ➡ Ranking algorithms focus on discriminating between these pages
  - ➡ Word proximity is important

Term Proximity

- Many models have been developed
- N-grams are commonly used in commercial web search
- Dependence model based on inference net has been effective in TREC - e.g.

```
#weight(
        0.8 #combine(embryonic stem cells)
        0.1 #combine( #od:1(stem cells) #od:1(embryonic stem)
                      #od:1(embryonic stem cells))
        0.1 #combine( #uw:8(stem cells) #uw:8(embryonic cells)
                      #uw:8(embryonic stem) #uw:12(embryonic stem cells)))
```

Example Web Query

```
#weight(
        0.1 #weight( 0.6 #prior(pagerank) 0.4 #prior(inlinks))
        1.0 #weight(
            0.9 #combine(
                #weight( 1.0 pet.(anchor) 1.0 pet.(title)
                         3.0 pet.(body) 1.0 pet.(heading))
                #weight( 1.0 therapy.(anchor) 1.0 therapy.(title)
                         3.0 therapy.(body) 1.0 therapy.(heading)))
            0.1 #weight(
                1.0 #od:1(pet therapy).(anchor) 1.0 #od:1(pet therapy).(title)
                3.0 #od:1(pet therapy).(body) 1.0 #od:1(pet therapy).(heading))
            0.1 #weight(
                1.0 #uw:8(pet therapy).(anchor) 1.0 #uw:8(pet therapy).(title)
                3.0 #uw:8(pet therapy).(body) 1.0 #uw:8(pet therapy).(heading)))
        )
```

Learning to Rank

Machine Learning and IR

- Considerable interaction between these fields
  - ➡ Rocchio algorithm (60s) is a simple learning approach
  - ➡ 80s, 90s: learning ranking algorithms based on user feedback
  - ➡ 2000s: text categorization
- Limited mainly by the amount of training data
- Web query logs have generated new wave of research
  - ➡ e.g., “Learning to Rank”

Generative vs. Discriminative

- All of the probabilistic retrieval models presented so far fall into the category of generative models
    - ➡ A **generative model** assumes that documents were generated from some underlying model (in this case, usually a multinomial distribution) and uses training data to estimate the parameters of the model
    - ➡ The probability of belonging to a class (i.e. the relevant documents for a query) is then estimated using Bayes' Rule and the document model

Generative vs. Discriminative

- A **discriminative model** estimates the probability of belonging to a class *directly* from the observed features of the document based on the training data

- Generative models perform well with low numbers of training examples

- Discriminative models usually have the advantage given enough training data

  ➡ Can also easily incorporate many features

Discriminative Models for IR

- Discriminative models can be trained using explicit relevance judgments or click data in query logs

- There is a large class of algorithms called *learning to rank*

  ➡ Learns weights on a linear (or non-linear) combination of features that is used to rank documents

  ➡ Finds the best weights to optimize some chosen performance metric

Ranking SVM

- The training data is:

$$(q_1, r_1), (q_2, r_2), \ldots, (q_n, r_n)$$

  - ➡ r_i is partial rank information: If document d_a should be ranked higher than d_b, then $(d_a, d_b) \in r_i$
  - ➡ This partial rank information generally comes from relevance judgments (allows multiple levels of relevance) or click data
  - ➡ If d_1, d_2 and d_3 are the documents in the first, second and third rank of the search output, but only d_3 was clicked: → (d_3, d_1) and (d_3, d_2) will be in the desired ranking for this query

Ranking SVM

- Learning a linear ranking function $\vec{w} \cdot \vec{d_a}$
  - ➡ w is a weight vector that is adjusted by learning
  - ➡ d_a is the vector representation of the features of a document
  - ➡ *non-linear* functions are also used
- Weights represent the relative importance of features
  - ➡ These are learned using training data
  - ➡ e.g.,

$$\vec{w} \cdot \vec{d_a} = (2,1,2) \cdot (2,4,1) = 2 \cdot 2 + 1 \cdot 4 + 2 \cdot 1 = 10$$

Ranking SVM

- The goal is to learn weights that satisfy as many of the following conditions as possible:

$$\begin{aligned}
\forall(d_i, d_j) \in r_1 \quad &: \quad \vec{w}.\vec{d_i} > \vec{w}.\vec{d_j} \\
&\quad \ldots \\
\forall(d_i, d_j) \in r_n \quad &: \quad \vec{w}.\vec{d_i} > \vec{w}.\vec{d_j}
\end{aligned}$$

- This can be formulated as an *optimization problem*, and a standard optimization tool can solve it.

Ranking SVM

$$
\begin{aligned}
minimize: \quad & \frac{1}{2}\vec{w}.\vec{w} + C\sum \xi_{i,j,k} \\
subject\ to: \quad & \\
\forall(d_i, d_j) \in r_1 \quad : \quad & \vec{w}.\vec{d_i} > \vec{w}.\vec{d_j} + 1 - \xi_{i,j,1} \\
& \ldots \\
\forall(d_i, d_j) \in r_n \quad : \quad & \vec{w}.\vec{d_i} > \vec{w}.\vec{d_j} + 1 - \xi_{i,j,n} \\
& \forall i \forall j \forall k : \xi_i, j, k \geq 0
\end{aligned}
$$

- ξ, known as a slack variable, allows for misclassification of difficult or noisy training examples, and C is a parameter that is used to prevent overfitting

Ranking SVM

- Software is available to do optimization
- Each pair of documents in our training data can be represented by the vector:

$$(\vec{d_i} - \vec{d_j})$$

- The score for this pair is:

$$\vec{w} \cdot (\vec{d_i} - \vec{d_j})$$

- A SVM classifier will find a w that makes the smallest score as large as possible
    - ➡ Makes the differences in scores as large as possible for the pairs of documents that are hardest to rank

Summary

- The best retrieval model depends on the application and the data available

- An evaluation corpus (or test collection), training data, and user data are all critical resources

- Open source search engines can be used to find effective ranking algorithms

    ➡ The Galago query language makes this particularly easy

- Language resources (e.g., a thesaurus) can make a big difference