



Introduction to NLP

"Artificial Intelligence: A Modern Approach", Chapter 22

Outline

- Introduction
- Language Models
- Text classification
- Information Retrieval
- Information Extraction

Introduction

- Somewhere around 100,000 years ago, humans learned how to speak, and about 7,000 years ago learned to write
- When Alan Turing proposed his Test, he based it on language
- There are two main reasons why we want our computer agents to be able to process natural languages:
 - To communicate with humans
 - To acquire information from written language

Information Extraction

Subject: **curriculum meeting**
Date: January 15, 2012

Event: Curriculum mtg
Date: Jan-16-2012
Start: 10:00am
End: 11:30am
Where: Gates 159

Hi Dan, we've now scheduled the curriculum meeting.
It will be in Gates 159 tomorrow from 10:00-11:30.
-Chris

Create new Calendar entry

Sentiment Analysis



Attributes:

zoom



affordability



size and weight



flash



ease of use



Size and weight

- ✓ • nice and compact to carry!
- ✓ • since the camera is small and light, I won't need to carry around those heavy, bulky professional cameras either!
- ✗ • the camera feels flimsy, is plastic and very light in weight you have to be very delicate in the handling of this camera



Machine Translation

- Fully automatic

Enter Source Text:

这不过是一个时间的问题。

Translation from Stanford's *Phrasal*:

This is only a matter of time.

- Helping human translators

Enter Source Text:

تعرض الرئيس اللبناني اميل لحود لـ حملة عنيفة في مجلس النواب الذي انعقد امس في جلسة تشريعية عادية تحولت الي " محاكمة " لـ رئيس الجمهورية علي موقفه من المحكمة الدولية و " الملاحظات " التي ادلي بها حول هذا الموضوع .

Translate Clear

Enter Translation:

lebanese |

- president
- suffered
- exposed
- president emile
- before
- presented
- offer

Done!

Language Technology

mostly solved

Spam detection

Let's go to Agra!



Buy V1AGRA ...



Part-of-speech (POS) tagging

ADJ ADJ NOUN VERB ADV

Colorless green ideas sleep furiously.

Named entity recognition (NER)

PERSON ORG LOC

Einstein met with UN officials in Princeton

making good progress

Sentiment analysis

Best roast chicken in San Francisco!



The waiter ignored us for 20 minutes.



Coreference resolution

Carter told Mubarak he shouldn't run again.

Word sense disambiguation (WSD)

I need new batteries for my *mouse*.



Parsing

I can see Alcatraz from the window!

Machine translation (MT)

第13届上海国际电影节开幕...



The 13th Shanghai International Film Festival...

Information extraction (IE)

You're invited to our dinner party, Friday May 27 at 8:30



Party
May 27
add

still really hard

Question answering (QA)

Q. How effective is ibuprofen in reducing fever in patients with acute febrile illness?

Paraphrase

XYZ acquired ABC yesterday

ABC has been taken over by XYZ

Summarization

The Dow Jones is up

The S&P500 jumped

Housing prices rose



Economy is good

Dialog

Where is Citizen Kane playing in SF?

Castro Theatre at 7:30. Do you want a ticket?



Basic Text Processing (Normalization)

- Need to “normalize” terms
 - Information Retrieval: indexed text & query terms must have same form.
 - We want to match *U.S.A.* and *USA*
- We implicitly define equivalence classes of terms
- Alternative: asymmetric expansion:
 - Enter: *window* Search: *window, windows*
 - Enter: *windows* Search: *Windows, windows, window*
 - Enter: *Windows* Search: *Windows*
 - Potentially more powerful, but less efficient
- Case folding
 - Applications like IR: reduce all letters to lower case

Word tokenization

- How many words?
they lay back on the San Francisco grass and looked at the stars and their ...
- **Type**: an element of the vocabulary.
- **Token**: an instance of that type in running text.
- How many?
 - 15 tokens (or 14)
 - 13 types (or 12) (or 11?)

N = number of tokens

V = vocabulary = set of types

$|V|$ is the size of the vocabulary

Language Modeling

■ Probabilistic Language Models

- Assign a probability to a sentence

- Machine Translation:

$P(\text{high winds tonite}) > P(\text{large winds tonite})$

- Spell Correction

- The office is about fifteen **minuets** from my house

$P(\text{about fifteen minutes from}) > P(\text{about fifteen minuets from})$

- Speech Recognition

$P(\text{I saw a van}) \gg P(\text{eyes awe of an})$

- + Summarization, question-answering, etc.!!

Probabilistic Language Modeling

- **Goal**: Compute the probability of a sentence or sequence of words:
 - $P(W) = P(w_1, w_2, w_3, w_4, w_5, \dots, w_n)$
- Related task: probability of an upcoming word:
 - $P(w_5 | w_1, w_2, w_3, w_4)$
- A model that computes either of these:
 $P(W)$ or $P(w_n | w_1, w_2 \dots w_{n-1})$
is called a **language model**.

How to compute $P(W)$

- How to compute this joint probability:

$P(\text{its, water, is, so, transparent, that})$

- Intuition: let's rely on the **Chain Rule of Probability**
- Recall the definition of conditional probabilities

$$P(B|A) = P(A, B) / P(A) \implies P(A, B) = P(A)P(B|A)$$

- More variables:

$$P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, B, C)$$

- The Chain Rule in General

$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1, \dots, x_{n-1})$$

The Chain Rule applied to compute joint probability of words in sentence

$$P(w_1 w_2 \dots w_n) = \prod_i P(w_i | w_1 w_2 \dots w_{i-1})$$

$P(\text{"its water is so transparent"}) =$

$P(\text{its}) \times P(\text{water} | \text{its}) \times P(\text{is} | \text{its water})$

$\times P(\text{so} | \text{its water is}) \times P(\text{transparent} | \text{its water is so})$

How to estimate these probabilities

- Could we just count and divide?

$P(\text{the} | \text{its water is so transparent that})$

$$= \frac{\text{Count}(\text{its water is so transparent that the})}{\text{Count}(\text{its water is so transparent that})}$$

- **No!** Too many possible sentences!
- We'll **never** see enough data for estimating these

Markov Assumption



Andrei Markov

- **Simplifying assumption:**

$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{that})$

- **Or maybe**

$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{transparent that})$

Markov Assumption

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$



Andrei Markov

- In other words, we approximate each component in the product

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-k} \dots w_{i-1})$$

Simplest case: Unigram model

$$P(w_1 w_2 \dots w_n) \approx \prod_{i=1}^n P(w_i)$$

- Some automatically generated sentences from a unigram model

fifth, an, of, futures, the, an, incorporated, a,
a, the, inflation, most, dollars, quarter, in, is,
mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the

Bigram model

- Condition on the previous word:

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-1})$$

texaco, rose, one, in, this, issue, is, pursuing, growth,
in, a, boiler, house, said, mr., gurria, mexico, 's, motion,
control, proposal, without, permission, from, five, hundred,
fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached

this, would, be, a, record, november

N-gram models

- Trigram model

$$P(w_i | w_1, w_2, \dots, w_{i-1}) \approx P(w_i | w_{i-2}, w_{i-1})$$

- We can extend to 4-grams, 5-grams, ...
- In general this is an insufficient model of language
 - Because language has **long-distance dependencies**

"The computer which I had just put into the machine room on the fifth floor crashed."

- But we can often get away with N-gram models

Estimating N-gram probabilities

- The Maximum Likelihood Estimate

- Unigram

$$P(w_i) = \frac{\text{count}(w_i)}{M}$$

M → Number of the tokens in the corpus

- Bigram

$$P(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}w_i)}{\text{count}(w_{i-1})}$$

- Trigram

$$P(w_i|w_{i-2}w_{i-1}) = \frac{\text{count}(w_{i-2}w_{i-1}w_i)}{\text{count}(w_{i-2}w_{i-1})}$$

An example

<s> I am Sam </s>

<s> Sam I am </s>

<s> I do not like green eggs and ham </s>

$$P(I | < s >) = \frac{2}{3}$$

$$P(am | I) = \frac{2}{3}$$

$$P(Sam | am) = \frac{1}{2}$$

$$P(not | do) = \frac{1}{1} = 1$$

$$P(Sam | < s >) = \frac{1}{3}$$

$$P(do | I) = \frac{1}{3}$$

$$P(</s> | am) = \frac{1}{2}$$

...

More examples

- Berkeley Restaurant Project sentences
 - can you tell me about any good cantonese restaurants close by
 - mid priced thai food is what i'm looking for
 - tell me about chez panisse
 - can you give me a listing of the kinds of food that are available
 - i'm looking for a good place to eat breakfast
 - when is caffe venezia open during the day

Raw bigram counts

- Out of 9222 sentences

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Raw bigram probabilities

- Normalize by unigrams:

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

- Result:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Bigram estimates of sentence probabilities

$$\begin{aligned} P(\langle s \rangle \text{ I want english food } \langle /s \rangle) &= \\ &P(\text{I} | \langle s \rangle) \\ &\times P(\text{want} | \text{I}) \\ &\times P(\text{english} | \text{want}) \\ &\times P(\text{food} | \text{english}) \\ &\times P(\langle /s \rangle | \text{food}) \\ &= .000031 \end{aligned}$$

Practical Issues

- We do everything in log space
 - Avoid underflow
 - (also adding is faster than multiplying)

$$p_1 \times p_2 \times p_3 \times p_4 = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

Evaluation: How good is our model?

- Does our language model prefer good sentences to bad ones?
 - Assign higher probability to "real" or "frequently observed" sentences
 - Than "ungrammatical" or "rarely observed" sentences?
- We train parameters of our model on a **training set**.
- We test the model's performance on data we haven't seen.
 - A **test set** is an unseen dataset that is different from our training set, totally unused.
 - An **evaluation metric** tells us how well our model does on the test set.

Intuition of Perplexity

■ The Shannon Game:

- How well can we predict the next word?

I always order pizza with cheese and _____

The 33rd President of the US was _____

I saw a _____

- Unigrams are terrible at this game. (**Why?**)

■ A better model of a text

- is one which assigns a higher probability to the word that actually occurs

mushrooms 0.1

pepperoni 0.1

anchovies 0.01

....

fried rice 0.0001

....

and 1e-100

Perplexity

- The best language model is one that best predicts an unseen test set

- Gives the highest $P(\text{sentence})$

- **Perplexity** is **the probability of the test set**, normalized by the number of words:

- Minimizing perplexity is the same as maximizing probability

- **Lower perplexity = better model**

- Training 38 million words, test 1.5 million words, WSJ

$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}} \end{aligned}$$

$$\text{Chain rule: } \text{PP}(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_1 \dots w_{i-1})}}$$

$$\text{For bigrams: } \text{PP}(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_{i-1})}}$$

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

Zeros

- Training set:
 - ... denied the allegations
 - ... denied the reports
 - ... denied the claims
 - ... denied the request

- Test set
 - ... denied the offer
 - ... denied the loan

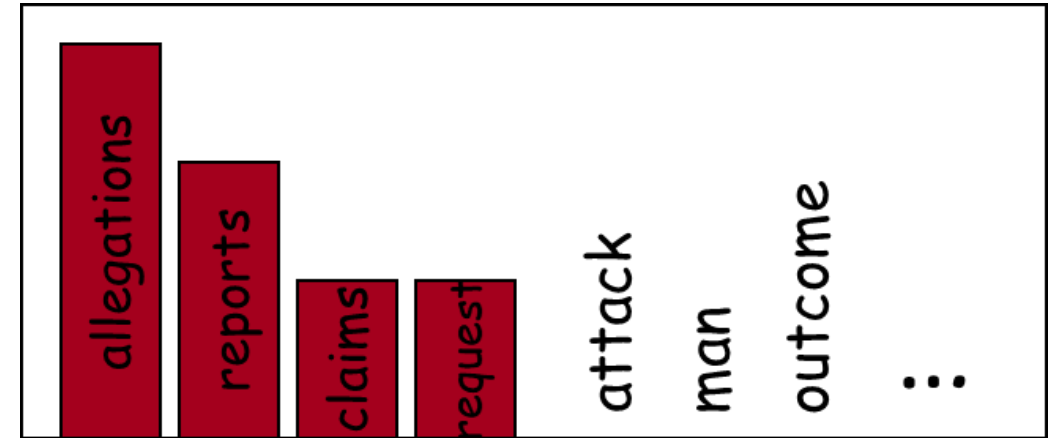
$$P(\text{offer} \mid \text{denied the}) = 0$$

The intuition of smoothing

- When we have sparse statistics:

$P(w \mid \text{denied the})$
3 allegations
2 reports
1 claims
1 request

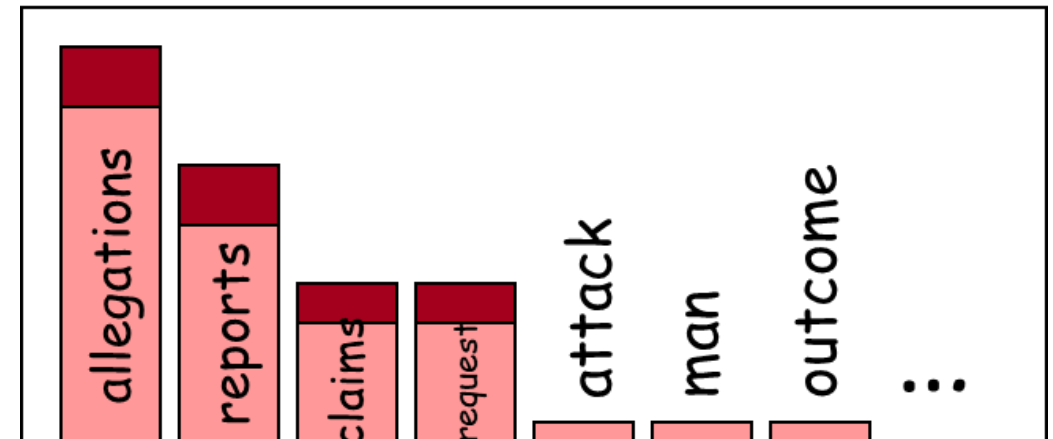
7 total



- Steal probability mass to generalize better

$P(w \mid \text{denied the})$
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other

7 total



Add-one estimation

- Also called Laplace smoothing
- Pretend we saw each word one more time than we did
- Just add one to all the counts!

- MLE estimate

$$P_{MLE}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

- Add-1 estimate:

$$P_{Add-1}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$$

words in the vocabulary
Or
types

Berkeley Restaurant Corpus: Laplace smoothed bigram counts

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

Laplace-smoothed bigrams

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Reconstituted counts

$$c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n) + 1] \times C(w_{n-1})}{C(w_{n-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Compare with raw bigram counts

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Backoff and Interpolation

- Sometimes it helps to **use less context**
 - Condition on less context for contexts you haven't learned much about
- **Backoff:**
 - Use trigram if you have good evidence,
 - Otherwise bigram, otherwise unigram
- **Interpolation:**
 - Mix unigram, bigram, trigram

$$\begin{aligned}\hat{P}(w_n|w_{n-1}w_{n-2}) = & \lambda_1 P(w_n|w_{n-1}w_{n-2}) \\ & + \lambda_2 P(w_n|w_{n-1}) \\ & + \lambda_3 P(w_n)\end{aligned}\quad \sum_i \lambda_i = 1$$

- Interpolation works better

The Task of Text Classification

- Is this spam?

Subject: Important notice!

From: Stanford University <newsforum@stanford.edu>

Date: October 28, 2011 12:34:16 PM PDT

To: undisclosed-recipients;;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

<http://www.123contactform.com/contact-form-StanfordNew1-236335.html>

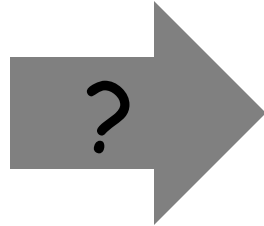
Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information about the new services.

© Stanford University. All Rights Reserved.

Male or female author?

1. By 1925 present-day Vietnam was divided into three parts under French colonial rule. The southern region embracing Saigon and the Mekong delta was the colony of Cochinchina; the central area with its imperial capital at Hue was the protectorate of Annam...
2. Clara never failed to be astonished by the extraordinary felicity of her own name. She found it hard to trust herself to the mercy of fate, which had managed over the years to convert her greatest shame into one of her greatest assets...

MEDLINE Article



- Antagonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- ...

Text Classification: definition

- Input:
 - a document d
 - a fixed set of classes $\mathcal{C} = \{c_1, c_2, \dots, c_J\}$
- Output: a predicted class $c \in \mathcal{C}$



Classification Methods:

Supervised Machine Learning

- **Input:**
 - A document d
 - A fixed set of classes $C = \{c_1, c_2, \dots, c_J\}$
 - A training set of m hand-labeled documents $(d_1, c_1), \dots, (d_m, c_m)$
- **Output:**
 - A learned classifier $\gamma: d \rightarrow c$

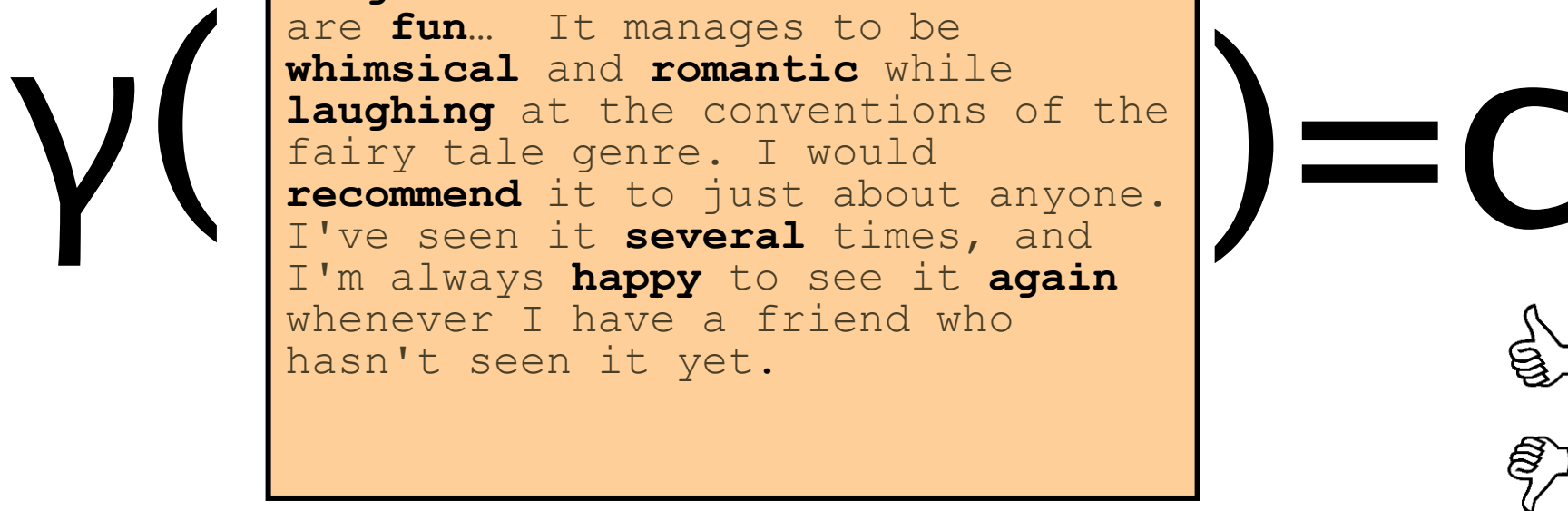
Naïve Bayes Intuition

- Simple ("naïve") classification method based on Bayes rule
- Relies on very simple representation of document
 - Bag of words

$$Y(\text{I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.}) = C$$


Naïve Bayes Intuition

- Simple ("naïve") classification method based on Bayes rule
- Relies on very simple representation of document
 - Bag of words



Naïve Bayes Intuition

- Simple ("naïve") classification method based on Bayes rule
- Relies on very simple representation of document
 - Bag of words

Y(

```
x love xxxxxxxxxxxxxxxx sweet
xxxxxxxx satirical xxxxxxxxxxx
xxxxxxxxxxxx great xxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxx fun xxxx
xxxxxxxxxxxxxxxx whimsical xxxx
romantic xxxx laughing
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxx recommend xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
several xxxxxxxxxxxxxxxxxxxxxxxx xxxxx
happy xxxxxxxx again
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
```

) = C





Naïve Bayes Intuition

- Simple (“naïve”) classification method based on Bayes rule
- Relies on very simple representation of document
 - Bag of words

$Y(\text{$

great	1
love	1
recommend	1
laugh	1
happy	1
...	...

$\text{)} = C$

Learning the Multinomial Naïve Bayes Model

- Simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

fraction of times word w_i appears
among all words in documents of topic c_j

$$C_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(w_i | c)$$

Information retrieval

- The task of finding documents that are relevant to a user's need for information.
 - Search engines on the World Wide Web
- An information retrieval (henceforth IR) system can be characterized by
 - A corpus of documents
 - Queries posed in a query language
 - A result set
 - A presentation of the result set

Information retrieval

- Boolean keyword model
 - Each word in the document collection is **treated as a Boolean feature**
 - The query language is the language of Boolean expressions over features
 - A document is relevant only if the expression evaluates to true
 - The query "information AND retrieval" is true for R&N: Chap. 22
 - Advantages
 - Simple to explain and implement
 - Disadvantages
 - The degree of relevance of a document is a single bit
 - Boolean expressions are unfamiliar to users who are not programmers or logicians
 - It can be hard to formulate an appropriate query, even for a skilled user

IR scoring functions

- Most IR systems use models based on the statistics of word counts
- BM25 scoring function
 - A scoring function takes a document and a query and returns a numeric score
 - The most relevant documents have the highest scores
 - The score is a linear weighted combination of scores for each of the words that make up the query

$$BM25(d_j, q_{1:N}) = \sum_{i=1}^N IDF(q_i) \cdot \frac{TF(q_i, d_j) \cdot (k + 1)}{TF(q_i, d_j) + k \cdot (1 - b + b \cdot \frac{|d_j|}{L})}$$

$$IDF(q_i) = \log \frac{N - DF(q_i) + 0.5}{DF(q_i) + 0.5}$$

$$L = \sum_i |d_i| / N \quad k = 2.0 \text{ and } b = 0.75$$

IR system evaluation

- Precision: % of selected items that are correct
- Recall: % of correct items that are selected

	Predicted Positives	Predicted Negatives
Positives	True Positives	False Negatives
Negatives	False Positives	True Negatives

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

$$\text{F1 score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

IR system evaluation

- Imagine that an IR system has returned a result set for a single query, for which we know which documents are and are not relevant, out of a corpus of 100 documents.

	In result set	Not in result set
Relevant	30	20
Not relevant	10	40

$$\text{precision} = \frac{30}{30 + 10} = 0.75 \quad \text{false positive rate} = 1 - 0.75 = 0.2$$

$$\text{recall} = \frac{30}{30 + 20} = 0.6 \quad \text{false negative rate} = 1 - 0.6 = 0.4$$

Example

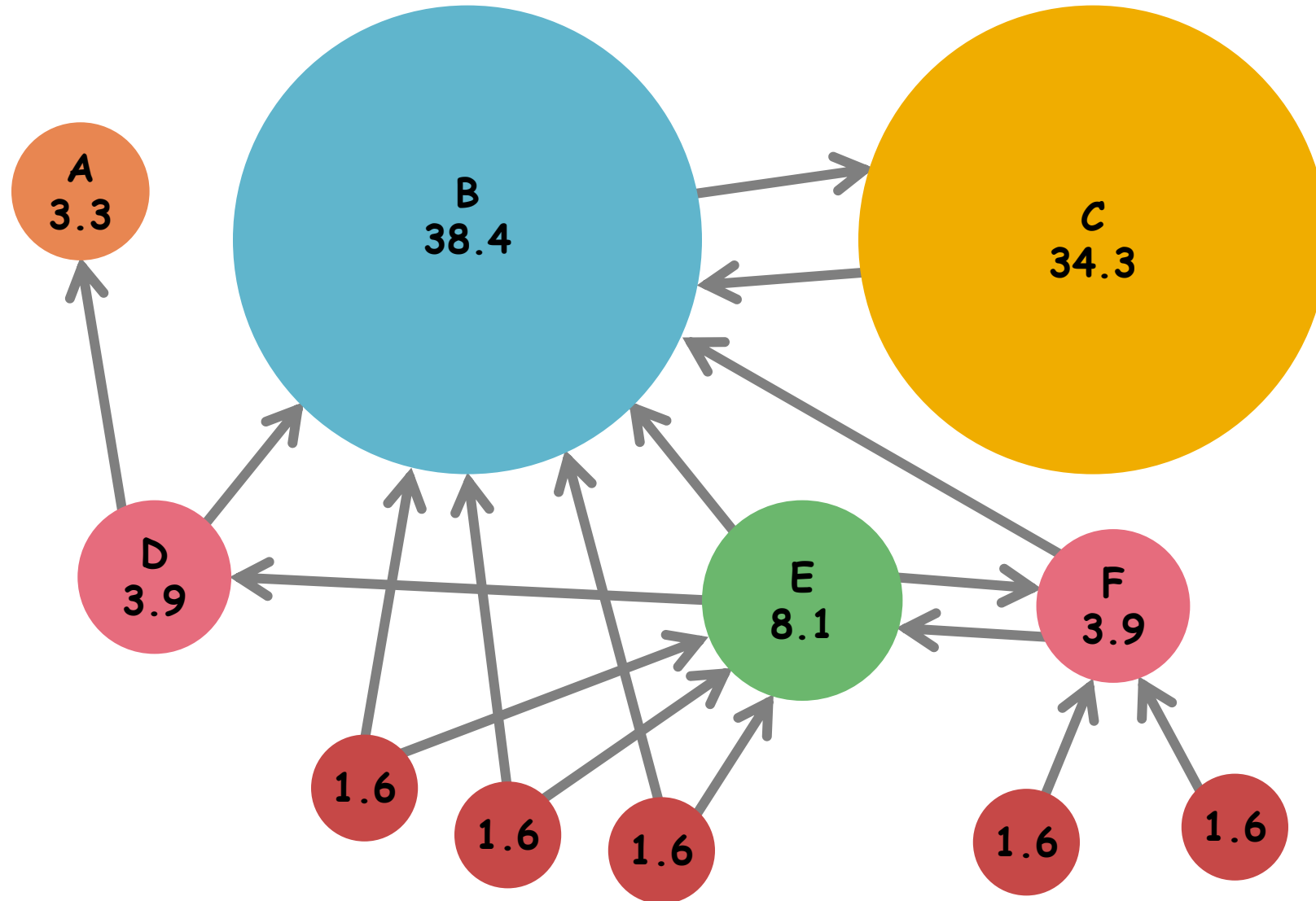
- Unigram Laplace smoothing
 - Tiny Corpus: $V = 4$

Word	True Ct	Unigram Prob	New Ct	Adjusted Prob
eat	10	.5	11	.46
British	4	.2	5	.21
food	6	.3	7	.29
happily	0	.0	1	.04
	20	1.0	~20	1.0

PageRank

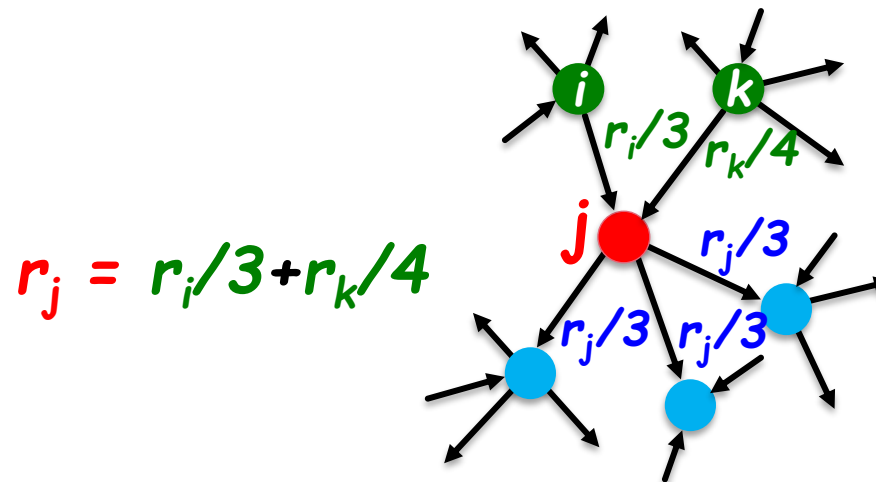
- **PageRank** was one of the two original ideas that set Google's search apart from other Web search engines when it was introduced in 1997
- If the query is [IBM], how do we make sure that IBM's home page, **ibm.com**, is the first result, even if another page mentions the term "IBM" more frequently?
- The idea is that ibm.com has many **in-links** (links to the page), so it should be ranked higher
 - each in-link is a vote for the quality of the linked-to page

PageRank



Simple Recursive Formulation

- Each link's vote is proportional to the **importance** of its source page
- If page **j** with importance r_j has n out-links, each link gets r_j / n votes
- Page **j**'s own importance is the sum of the votes on its in-links

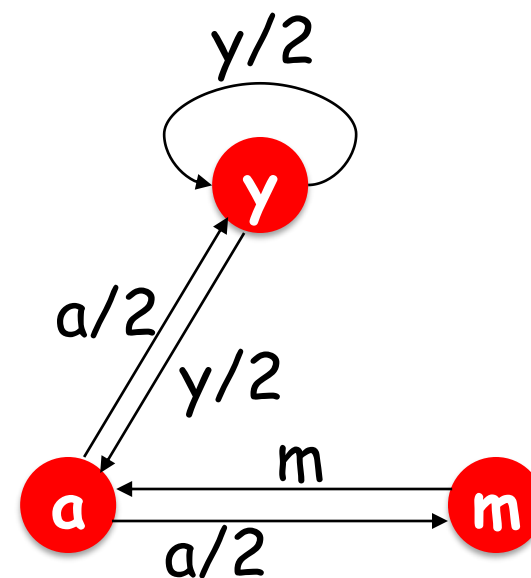


PageRank: The "Flow" Model

- A "vote" from an important page is worth more
- A page is important if it is pointed to by other important pages
- Define a "rank" r_j for page j

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

d_i ... out-degree of node i



"Flow" equations:

$$r_y = r_y / 2 + r_a / 2$$

$$r_a = r_y / 2 + r_m$$

$$r_m = r_a / 2$$

Solving the Flow Equations

- 3 equations, 3 unknowns,
no constants

- No unique solution
- All solutions equivalent modulo the scale factor

- Additional constraint forces uniqueness:

- $r_y + r_a + r_m = 1$
- **Solution:** $r_y = \frac{2}{5}, r_a = \frac{2}{5}, r_m = \frac{1}{5}$

Flow equations:

$$r_y = r_y / 2 + r_a / 2$$

$$r_a = r_y / 2 + r_m$$

$$r_m = r_a / 2$$

PageRank: Matrix Formulation

- **Stochastic adjacency matrix M**

- Let page i has d_i out-links
- If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
 - M is a **column stochastic matrix**
 - Columns sum to 1

- **Rank vector r** : vector with an entry per page

- r_i is the importance score of page i
- $\sum_i r_i = 1$

- **The flow equations can be written**

$$r = M \cdot r$$

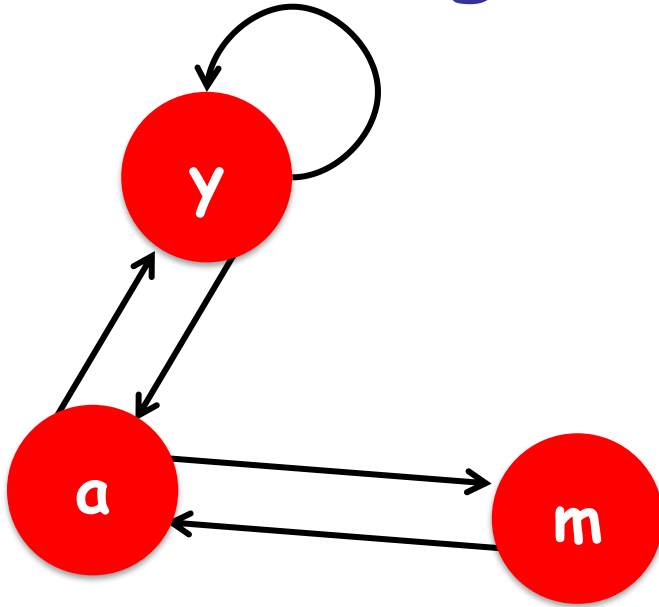
PageRank: Example

- Given a web graph with n nodes, where the nodes are pages and edges are hyperlinks
- **Power iteration:** a simple iterative scheme
 - Suppose there are N web pages
 - Initialize: $\mathbf{r}^{(0)} = [1/N, \dots, 1/N]^T$
 - Iterate: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
 - Stop when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$

$|\mathbf{x}|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the L_1 norm

Can use any other vector norm, e.g., Euclidean

PageRank: Example



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	1
m	0	$\frac{1}{2}$	0

$$r = M \cdot r$$

$$r_y = r_y / 2 + r_a / 2$$

$$r_a = r_y / 2 + r_m$$

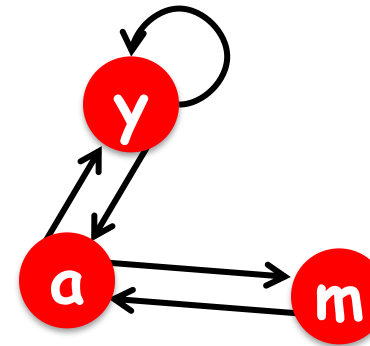
$$r_m = r_a / 2$$

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}$$

PageRank: Example

■ Power iteration:

- Initialize: $\mathbf{r}^{(0)} = [1/N, \dots, 1/N]^T$
- Iterate: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
- Stop when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	1
m	0	$\frac{1}{2}$	0

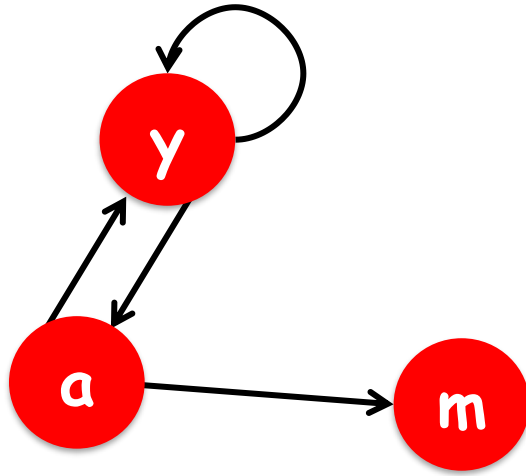
$$\begin{aligned}
 r_y &= r_y / 2 + r_a / 2 \\
 r_a &= r_y / 2 + r_m \\
 r_m &= r_a / 2
 \end{aligned}$$

$$\begin{array}{l}
 r_y \\
 r_a \\
 r_m
 \end{array}
 \begin{array}{|c|}
 \hline 1/3 \\
 \hline 1/3 \\
 \hline 1/3 \\
 \hline
 \end{array}
 =
 \begin{array}{|c|}
 \hline 1/3 \\
 \hline 3/6 \\
 \hline 1/6 \\
 \hline
 \end{array}$$

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{array}{cccc} 1/3 & 1/3 & 5/12 & 9/24 \\ 1/3 & 3/6 & 1/3 & 11/24 \dots \\ 1/3 & 1/6 & 3/12 & 1/6 \end{array} \quad \begin{array}{c} 6/15 \\ 6/15 \\ 3/15 \end{array}$$

Iteration 0, 1, 2, ...

PageRank



	y	a	m
y	$\frac{1}{2}$	$\frac{1}{2}$	0
a	$\frac{1}{2}$	0	0
m	0	$\frac{1}{2}$	0

$$r_y = r_y / 2 + r_a / 2$$

$$r_a = r_y / 2$$

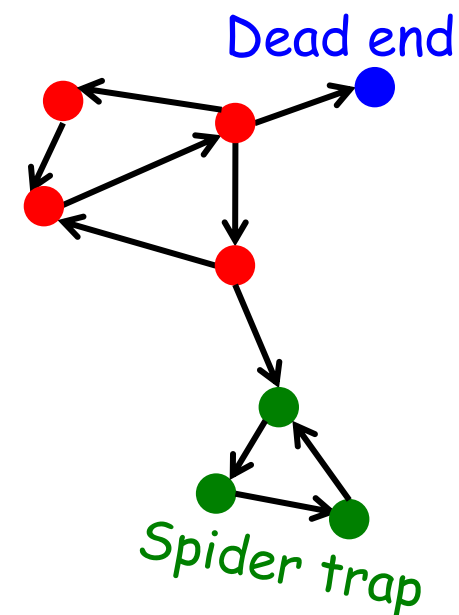
$$r_m = r_a / 2$$

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \dots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{pmatrix}$$

PageRank: Problems

2 problems:

- (1) Some pages are **dead ends** (have no out-links)
 - Random walk has "nowhere" to go to
 - Such pages cause importance to "leak out"
- (2) **Spider traps:** (all out-links are within the group)
 - Random walked gets "stuck" in a trap
 - And eventually spider traps absorb all importance



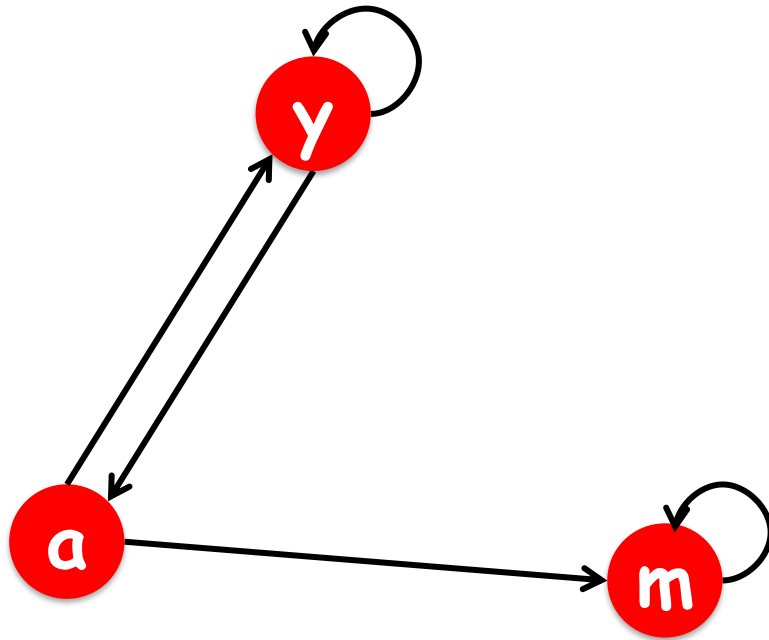
Solution: Teleports!

- **At each time step, the random surfer has two options**
 - With prob. β , follow a link at random
 - With prob. $1-\beta$, jump to some random page
 - Common values for β are in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**

$$r_j = \sum_{i \rightarrow j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N} \quad A = \beta M + (1 - \beta) \left[\frac{1}{N} \right]_{N \times N}$$

$$\mathbf{r} = \mathbf{A} \cdot \mathbf{r}$$

Random Teleports ($\beta = 0.8$)



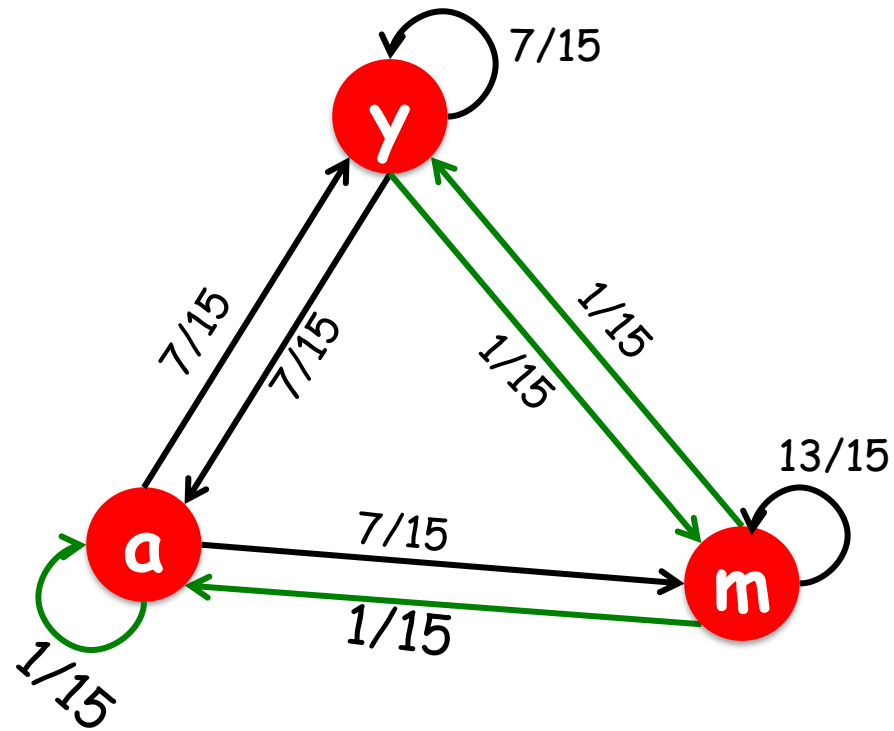
M

1/2	1/2	0
1/2	0	0
0	1/2	1

$[1/N]_{N \times N}$

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix}$$

Random Teleports ($\beta = 0.8$)



$$0.8 \begin{matrix} & \text{y} & \text{a} & \text{m} \\ \begin{matrix} \text{y} \\ \text{a} \\ \text{m} \end{matrix} & \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \end{matrix} + 0.2 \begin{matrix} & \text{y} & \text{a} & \text{m} \\ \begin{matrix} \text{y} \\ \text{a} \\ \text{m} \end{matrix} & \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \end{matrix}$$

$$\begin{matrix} & \text{y} & \text{a} & \text{m} \\ \begin{matrix} \text{y} \\ \text{a} \\ \text{m} \end{matrix} & \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix} \end{matrix}$$

A

$$\begin{pmatrix} \text{y} \\ \text{a} \\ \text{m} \end{pmatrix} = \begin{matrix} & 1/3 & 0.33 & 0.24 & 0.26 & & 7/33 \\ & 1/3 & 0.20 & 0.20 & 0.18 & \dots & 5/33 \\ & 1/3 & 0.46 & 0.52 & 0.56 & & 21/33 \end{matrix}$$

The HITS algorithm

- Also known as "Hubs and Authorities" or HITS
- HITS differs from PageRank in several ways
 - It is **a query-dependent measure**: it rates pages with respect to a query
 - It must be computed anew for each query
- HITS first finds a set of pages that are relevant to the query
- Each page in this set **is considered an authority** on the query **to the degree that other pages** in the relevant set **point to it**
- A page is considered **a hub** to the **degree that it points to other authoritative** pages in the relevant set

The HITS algorithm

function HITS(*query*) **returns** *pages* with hub and authority numbers

pages \leftarrow EXPAND-PAGES(RELEVANT-PAGES(*query*))

for each *p* **in** *pages* **do**

p.AUTHORITY \leftarrow 1

p.HUB \leftarrow 1

repeat until convergence **do**

for each *p* **in** *pages* **do**

p.AUTHORITY $\leftarrow \sum_i \text{INLINK}_i(p).\text{HUB}$

p.HUB $\leftarrow \sum_i \text{OUTLINK}_i(p).\text{AUTHORITY}$

 NORMALIZE(*pages*)

return *pages*

