دانشگاه صنعتی امیرکبیر
دانشکده مهندسی کامپیوتر و فناوری اطلاعات

پردازش زبان‌های طبیعی

«هوش مصنوعی: یک رهیافت نوین»، فصل ۲۲

ارائه‌دهنده: سیده فاطمه موسوی

نیم‌سال اول ۱۴۰۰-۱۳۹۹

عناوین

- مقدمه
- مدل‌های زبان
- دسته بندی متن
- بازیابی اطلاعات
- استخراج اطلاعات

مقدمه

- چرا می‌خواهیم عامل‌های کامپیوتری قادر به پردازش زبان‌های طبیعی باشند؟
  - ارتباط برقرار کردن با افراد
  - به‌دست آوردن اطلاعات ارزشمند از زبان نوشتاری
- عاملی که به دنبال کسب دانش است، باید تا جای ممکن زبان‌های مبهم و گسترده‌ای که انسان‌ها استفاده می‌کنند را متوجه شود.
  - در این فصل این مورد را از نقطه‌نظر وظایف جستجوی اطلاعات مانند دسته‌بندی متن و بازیابی اطلاعات بررسی می‌کنیم.

مدل‌های زبانی

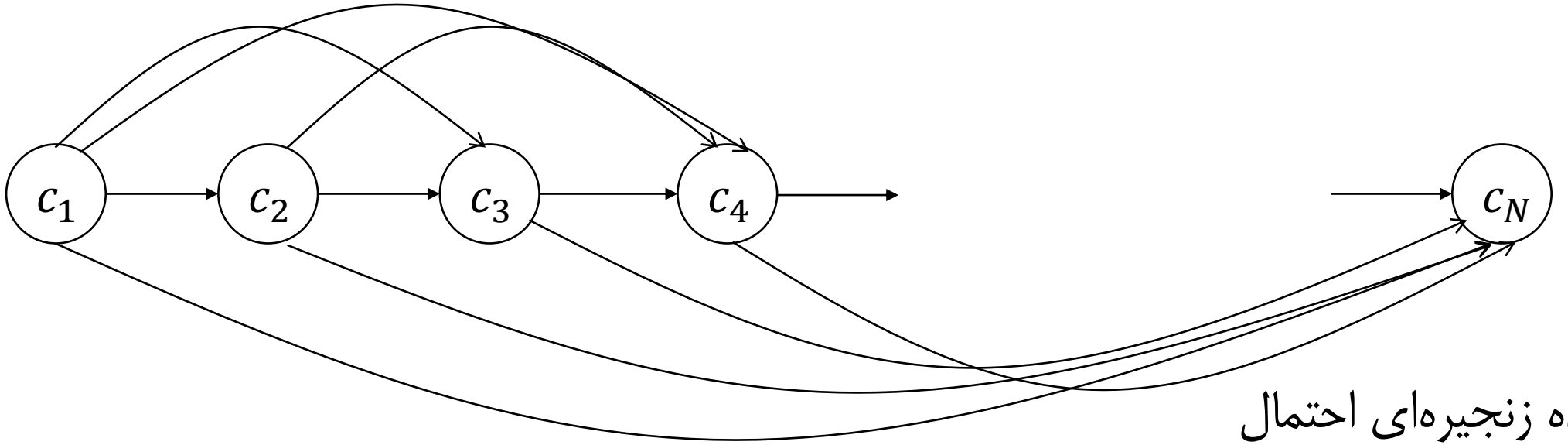
- زبان‌های رسمی (مانند پایتون، جاوا و ...)
  - به طور دقیق با مجموعه‌ای از قوانین (گرامر) مشخص می‌شوند. مثلا print(2+2) متعلق به زبان پایتون است اما print)2(2+ متعلق به این زبان نیست.
  - قوانینی دارند که معنا را به‌طور کامل مشخص می‌کند. مثلا معنای 2+2 عدد ۴ است.
- زبان‌های طبیعی (مانند انگلیسی و ...)
  - قواعد گرامری کامل و واضحی ندارند
    - * To be not invited is sad
  - به جای سوال در مورد تعلق یک جمله به یک زبان، می‌توان پرسید چقدر احتمال دارد آن جمله متعلق به آن زبان باشد.
  - ابهام در سطوح مختلف می‌تواند وجود داشته باشد
    - I saw the man with the telescope

مدل‌های زبانی

- تعریف یک مدل زبان طبیعی به‌صورت یک توزیع احتمالاتی
  - تخمین احتمال هر کلمه به شرط متن قبلی
  - P(phone | Please turn off your cell)
- n-gram: یک دنباله از n واحد (کاراکتر یا کلمه)
  - unigram، bigram و trigram برای n برابر با یک، دو و سه
- مدل n-gram توزیع احتمالاتی دنباله‌های n واحدی را به دست می‌آورد
  - با در نظر گرفتن فرض مارکوف، احتمال هر واحد تنها وابسته به n-1 واحد قبل است.
- Unigram: P(phone)
  Bigram: P(phone | cell)
  Trigram: P(phone | your cell)

مدل‌های زبانی

- $P(c_{1:N})$ احتمال دنباله Nتایی از واحدها $c_{1:N} = c_1 c_2 c_3 \dots c_N$

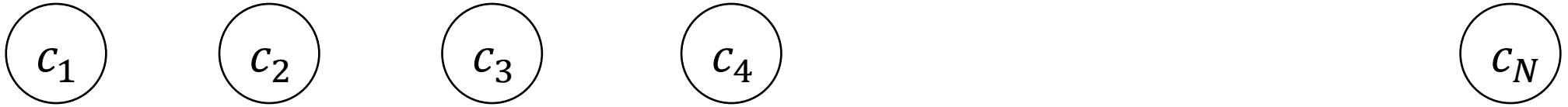


- قاعده زنجیره‌ای احتمال

$$P(c_{1:N}) = P(c_1)P(c_2|c_1)P(c_3|c_1c_2) \dots P(c_N|c_{1:N-1}) = \prod_{i=1}^{N} P(c_i|c_{1:i-1})$$

مدل‌های زبانی

- تقریب unigram



$$P(c_{1:N}) = P(c_1)P(c_2)P(c_3)\dots P(c_N) = \prod_{i=1}^{N} P(c_i)$$

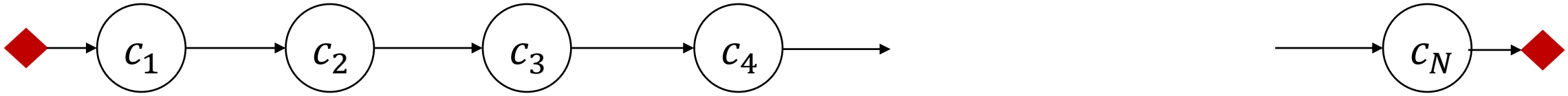
$$P(a) = \frac{7}{10} \qquad P(b) = \frac{3}{10}$$

$$P(baa) = \frac{2}{10} \times \frac{7}{10} \times \frac{7}{10}$$

<s> abaa </s>
<s> aaab </s>
<s> ab</s>

مدل‌های زبانی

- تقریب bigram



$$P(c_{1:N}) = P(c_1|<s>)P(c_2|c_1)P(c_3|c_2)\dots P(c_N|c_{N-1}) = \prod_{i=1}^{N} P(c_i|c_{i-1})$$

<s> abaa </s>
<s> aaab </s>
<s> ab</s>

$$P(a|<s>) = \frac{3}{3} = 1 \qquad P(b|<s>) = \frac{0}{3} = 0$$

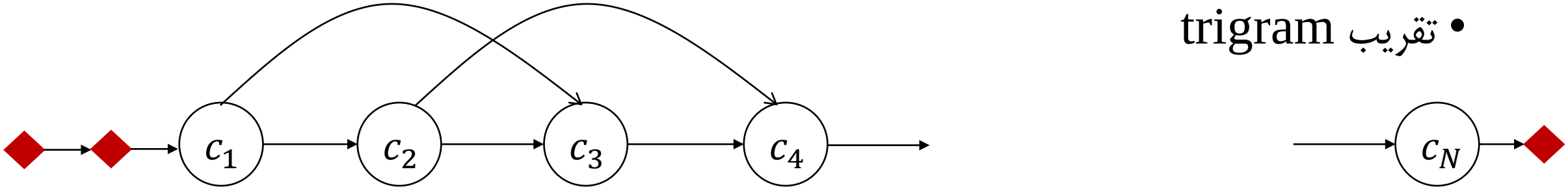
$$P(a|a) = \frac{3}{7} \qquad P(b|a) = \frac{3}{7} \qquad P(</s>|a) = \frac{1}{7}$$

$$P(a|b) = \frac{1}{3} \qquad P(b|b) = \frac{0}{3} \qquad P(</s>|b) = \frac{2}{3}$$

$$P(baa) = 0 \times \frac{1}{3} \times \frac{3}{7} \times \frac{1}{7}$$

مدل‌های زبانی

- تقریب trigram



$$P(c_{1:N}) = \prod_{i=1}^{N} P(c_i | c_{i-2:i-1})$$

<s> abaa </s>
<s> aaab </s>
<s> ab</s>

$$P(a| < s >) = \frac{3}{3} = 1 \qquad P(b| < s >) = \frac{0}{3} = 0$$

$$P(a| < s > a) = \frac{1}{3} \qquad P(b| < s > a) = \frac{2}{3} \qquad P(</s > | < s > a) = \frac{0}{3}$$

...

ساخت مدل زبانی

- استفاده از پیکره‌ها (corpus) برای یادگیری پارامترهای مدل
- احتمالات شرطی n-gram را می‌توان با استفاده از یک پیکره متون آموزشی براساس فرکانس نسبی دنباله واحدها تخمین زد.
- unigram: M تعداد کل واحدها در پیکره

$$P(c_i) = \frac{count(c_i)}{M}$$

- bigram

$$P(c_i|c_{i-1}) = \frac{count(c_{i-1}c_i)}{count(c_{i-1})}$$

- trigram

$$P(c_i|c_{i-2:i-1}) = \frac{count(c_{i-2}c_{i-1}c_i)}{count(c_{i-2}c_{i-1})}$$

ساخت مدل زبانی

- پیکره آموزشی:

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

- احتمالات محاسبه شده برای مدل bigram کلمه‌ای:

$$P(I|<s>) = \frac{2}{3} \qquad P(Sam|<s>) = \frac{1}{3}$$
$$P(am|I) = \frac{2}{3} \qquad P(do|I) = \frac{1}{3}$$
$$P(Sam|am) = \frac{1}{2} \qquad P(</s>|am) = \frac{1}{2}$$
$$P(not|do) = \frac{1}{1} = 1$$

...

- توجه کنید که مجموع احتمالاتی که سمت شرط آن‌ها یکی است باید برابر با یک شود.

ارزیابی مدل زبانی

<s> abaa </s>
<s> aaab </s>
<s> ab</s>

M_1: $P(a) = 0.5$, $P(b) = 0.5$
M_2: $P(a) = 0.7$, $P(b) = 0.3$

$< s > baaa </ s >$

- انتخاب یک مدل از میان مدل‌های مختلف یک زبان
  - استفاده از دو پیکره مجزای آموزشی و تست
  - پارامترهای مدل با استفاده از پیکره آموزشی تعیین می‌شوند
  - مدل بر روی پیکره تست ارزیابی می‌شود
- اگر متن پیکره تست با $c_{1:N}$ نشان داده شود، احتمال آن یعنی $P(c_{1:N})$ را با استفاده از مدل زبانی مدنظر محاسبه می‌شود.
- هرقدر این احتمال بیشتر باشد، مدل بهتری خواهیم داشت.
- تعریف معیار perplexity:

$$Perplexity(c_{1:N}) = P(c_{1:N})^{-\frac{1}{N}}$$

پیچیدگی مدل‌های زبانی

- یک مدل کاراکتری 3-gram از یک زبان با ۱۰۰ کاراکتر نیاز به محاسبه چند پارامتر دارد؟
  - یک میلیون پارامتر (۱۰۰*۱۰۰*۱۰۰)
- هرقدر n در مدل n-gram بیشتر باشد، تعداد پارامترها بیشتر شده و امکان عدم وجود برخی از دنباله‌های nتایی در پیکره بیشتر خواهد شد.
  - محاسبه پارامترها نیاز به یک پیکره غنی دارد تا اعدادی که به عنوان احتمال ارائه می‌شوند قابل اعتماد باشد.
  - حتی با وجود استفاده از یک پیکره طولانی، برخی از دنباله‌ها خیلی نامعمول هستند و امکان دادن مقدار صفر برای چنین دنباله‌هایی وجود دارد. درحالی که ممکن است این دنباله در جملاتی خارج از پیکره آموزشی مشاهده شوند.
    - برای مثال احتمال "<s>ht" صفر است (هیچ کلمه‌ای وجود ندارد که با ht شروع شده باشد.)
    - در این صورت احتمال جمله "The program issues an http request" صفر خواهد شد!

هموارسازی مدل‌های زبانی

- هموارسازی: تنظیم احتمال عبارات کم تکرار
  - قابلیت توسعه زبان به متون دیده نشده
- Pierre-Simon Laplace
  - اگر یک متغیر بولین تصادفی X در تمام n مشاهده تا کنون false باشد تخمین P(X=true) برای آن برابر با 1/(n+2) در نظر گرفته می‌شود.
  - فرض می‌شود با انجام دو trial بیشتر یکی ممکن است false باشد و دیگری true
- Backoff model
  - برای مثال در مورد trigram:

$$\widehat{P}(c_i|c_{i-2:i-1}) = \lambda_3 P(c_i|c_{i-2:i-1}) + \lambda_2 P(c_i|c_{i-1}) + \lambda_1 P(c_i)$$

$$\lambda_3 + \lambda_2 + \lambda_1 = 1$$

شناسایی زبان با مدل زبانی کاراکتری

- می‌خواهیم با دادن یک متن تعیین کنیم که به کدام زبان طبیعی نوشته شده است.
  - ساخت یک مدل زبانی (مثلا trigram) برای هر یک از زبان‌های مورد نظر
  - استفاده از قانون بیز برای انتخاب محتمل‌ترین زبان
  - چگونگی محاسبه احتمال $p(l)$

$$\begin{aligned}
\ell^* &= \underset{\ell}{\operatorname{argmax}}\ P(\ell \mid c_{1:N}) \\
&= \underset{\ell}{\operatorname{argmax}}\ P(\ell) P(c_{1:N} \mid \ell) \\
&= \underset{\ell}{\operatorname{argmax}}\ P(\ell) \prod_{i=1}^{N} P(c_i \mid c_{i-2:i-1}, \ell)
\end{aligned}$$

$$P(l|c_{1:N}) = \frac{P(l)P(c_{1:N}|l)}{P(c_{1:N})}$$

مدل زبانی کلمه‌ای

- استفاده از کلمات به جای کاراکترها
  - تعداد زیاد مجموعه واژگان نسبت به کاراکترها
- پس از آموزش مدل‌های مختلف با متن کتاب هوش مصنوعی عبارات زیر تولید شده است:

- *Unigram:* logical are as are confusion a may right tries agent goal the was . . .
- *Bigram:* systems are very similar computational approach would be represented . . .
- *Trigram:* planning and scheduling are integrated the success of naive bayes model is . . .

مدل زبانی کلمه‌ای

- احتمال مشاهده کلمه جدید برای مجموعه خارج از مجموعه واژگان (out of vocabulary)
- راه‌حل:
  - <UNK> را به عنوان نماد کلمات ناشناخته وارد مجموعه واژگان می‌کنیم.
  - در فاز آموزش، هر بار کلمه‌ای ظاهر شود که تا قبل از این مشاهده نشده بود به جای آن <UNK> قرار داده و آن کلمه را به مجموعه واژگان اضافه می‌کنیم. رخدادهای بعدی این کلمه بدون تغییر باقی خواهد ماند.
  - سپس تعداد رخداد هر یک از n-gram ها به طور معمول محاسبه شده و با <UNK> به عنوان یک کلمه رفتار می شود.
  - در فاز تست، در صورت مشاهده یک کلمه ناشناخته احتمال مربوط به آن را استفاده می‌کنیم.

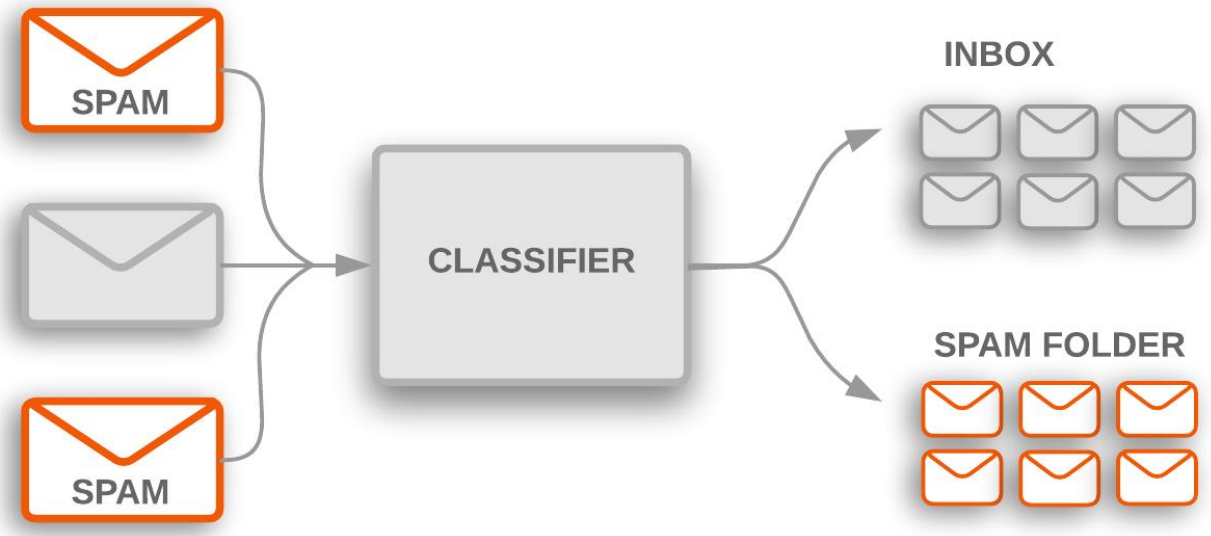
This is the first class about AI. I love AI.

<UNK>

دسته‌بندی متن

- تعیین دسته متن داده شده از میان مجموعه‌ای از دسته‌های از پیش تعریف شده
  - براساس مدل زبانی ایجاد شده از هر یک از دسته‌ها

$$\underset{c \in \{spam, ham\}}{\operatorname{argmax}} \; P(c \mid message) = \underset{c \in \{spam, ham\}}{\operatorname{argmax}} \; P(message \mid c)\, P(c)$$



دسته‌بندی متن

- براساس بازنمایی سند با استفاده ازمجموعه‌ای از ویژگی‌ها
  - نیازمند تعریف و انتخاب ویژگی
  - برای مثال تعداد دفعات تکرار هر کلمه در هر پیام، زمان ارسال پیام، طول پیام و ...
  - زیاد بودن تعداد ویژگی‌ها و درنتیجه طولانی بودن بردار ساخته شده برای پیام‌ها
  - کدام یک از ویژگی‌ها بهتر هستند؟
    - تعداد تکرار کلمه the یا تعداد تکرار کلمه free
- استفاده از الگوریتم‌های دسته‌بندی بر روی بردارهای به‌دست آمده
  - مانند شبکه عصبی، درخت تصمیم و ...

بازیابی اطلاعات

- یافتن اسناد مرتبط با نیاز اطلاعاتی کاربر
  - موتورهای جستجوی وب
- اجزا یک سیستم بازیابی اطلاعات
  - پیکره‌ای از اسناد متنی
  - درخواست‌های جستجو (query)
    - [AI book], [“AI book”]
    - [AI AND book]
    - [AI book site:www.aaai.org]
- مجموعه نتایج: مجموعه‌ای از اسناد که مرتبط با query هستند.
- ارائه نتایج: مثلا یک لیست رتبه‌بندی شده از اسناد
- موتور جستجو

بازیابی اطلاعات

- یک موتور جستجوی ساده: اجرای یک مدل بولین برای بازیابی اطلاعات
  - با هر کلمه در مجموعه اسناد به صورت یک ویژگی بولین رفتار می‌شود. اگر آن کلمه در query اتفاق بیفتند true و درغیر اینصورت false خواهد بود.
  - زبان query نیز به صورت عبارات بولین است. سندی مرتبط است که query برای آن true ارزیابی گردد.
  - ویژگی‌های این مدل
    - ساده برای پیاده‌سازی
    - درجه ارتباط یک سند با درخواست به‌صورت صفر و یک
    - آشنایی با زبان منطق و نوشتن یک درخواست مناسب
      - [information AND retrieval AND models AND optimization]
      - [information OR retrieval OR models OR optimization]

توابع رتبه‌بندی IR

- رتبه‌بندی اسناد براساس میزان شباهت آن‌ها به درخواست کاربر
- سه فاکتور بر روی وزن یک ترم در query تاثیرگذار است:
  - تعداد تکرار آن ترم در سند (Term frequency = TF)
  - عکس فرکانس سند در مورد یک ترم (Inverse Document Frequency = IDF)
    - برای query مانند [farming in Iran] تعداد اسنادی که کلمه "in" در آن‌ها ظاهر شده است بیشتر از تعداد اسنادی است که کلمه "farming" ظاهر شده است. پس کلمه "in" اهمیت چندانی نسبت به کلمه "farming" برای این query ندارد.
  - طول سند
    - یک سند با میلیون‌ها کلمه احتمالا تمام کلمات query را ذکر کند اما ممکن است واقعا در مورد query نباشد. یک سند کوتاه که تمام کلمات را ذکر کند کاندید بهتری است.

توابع رتبه‌بندی IR

- تابع رتبه BM25

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

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

$$L = \sum_i |d_i| / N$$

$$k = 2.0 \text{ and } b = 0.75$$

ارزیابی سیستم بازیابی اطلاعات

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

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

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

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

ارزیابی سیستم بازیابی اطلاعات

- یک query به سیستم داده می‌شود و مجموعه نتایج آن توسط انسان رتبه‌بندی می‌شود.

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

$$\text{precision} = \frac{30}{30+10} = 0.75$$

$$false\ positive\ rate\ = 1 - 0.75 = 0.2$$

$$\text{recall} = \frac{30}{30+20} = 0.6$$

$$false\ negative\ rate\ = 1 - 0.6 = 0.4$$

ارزیابی سیستم بازیابی اطلاعات

- نتیجه recall و precision برای دو سیستم زیر چه خواهد بود؟
  - سیستم تمام اسناد موجود را به عنوان نتیجه برگرداند.

	In result set	Not in result set
Relevant	50	0
Not relevant	50	0

- سیستم تنها یکی از اسناد مرتبط را برگرداند.

	In result set	Not in result set
Relevant	1	49
Not relevant	0	50

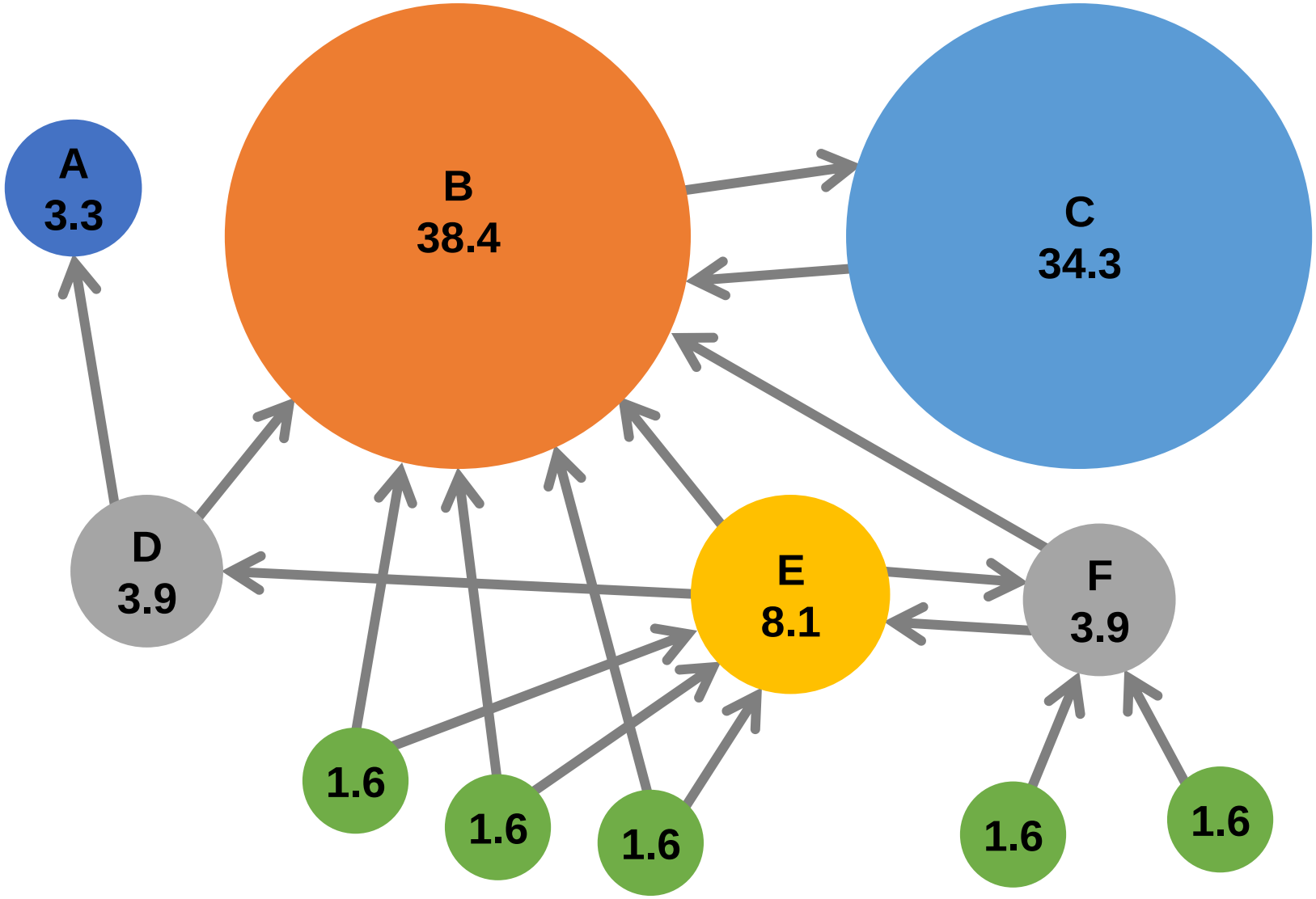
بهبودهای بازیابی اطلاعات

- بهبود الگوریتم امتیازدهی
  - تغییر نحوه تاثیر طول سند در امتیاز
  - در نظر گرفتن ارتباط میان کلمات
    - تبدیل حروف کوچک و بزرگ (case folding)
    - ریشه یابی کلمات (stemming)
    - کلمات هم معنی (synonyms)
  - استفاده از داده های جانبی (meta-data)
    - ارتباطات میان صفحات وب

رتبه صفحه (Page Rank)

- اگر درخواست شما [IBM] باشد مطمئنا انتظار دارید اولین نتیجه نمایش داده شده صفحه ibm.com باشد، حتی اگر تعداد تکرار کلمه IBM در صفحات دیگر بیشتر باشد.
  - یعنی علاوه بر حضور لغت در صفحه، میزان اطمینانی که به آن صفحه داریم نیز اهمیت دارد.
  - هر قدر لینک‌های ورودی به یک صفحه بیشتر باشد، آن صفحه معتبرتر است.
    - افزایش رتبه یک صفحه با ایجاد یک شبکه از صفحات که به این صفحه لینک می‌دهند!!
  - باید وزن لینک‌هایی که از صفحات معتبرتر می‌آیند بیشتر باشد.

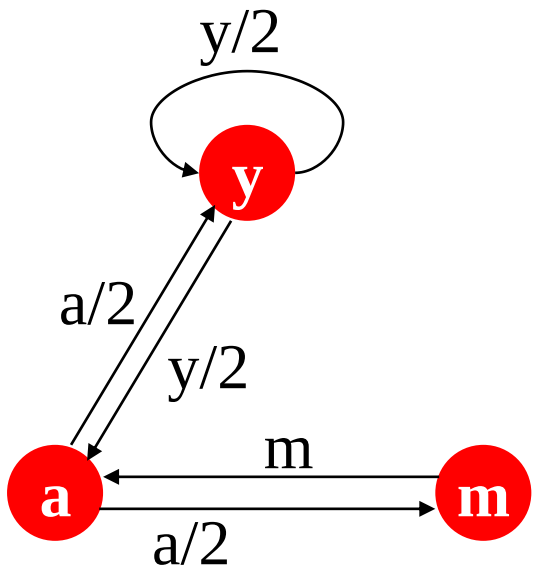
رتبه صفحه (Page Rank)



رتبه صفحه (Page Rank)

- رتبه r_j برای صفحه j

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

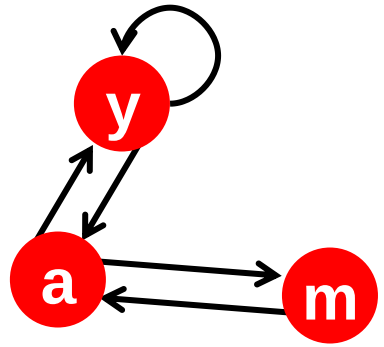


$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$
$$r_y + r_a + r_m = 1$$

- تفسیر رتبه صفحه با قدم زن تصادفی
  - فرض کنید یک قدم‌زن تصادفی بر روی گراف قرار دادیم
  - اگر در زمان t روی نود i باشد در زمان t+1 یکی از لینک‌های خروجی i را به‌طور تصادفی یکنواخت انتخاب می‌کند و نودی مانند j می‌رسد. این فرایند به طور نامتناهی تکرار می‌شود.

رتبه صفحه (Page Rank)

	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

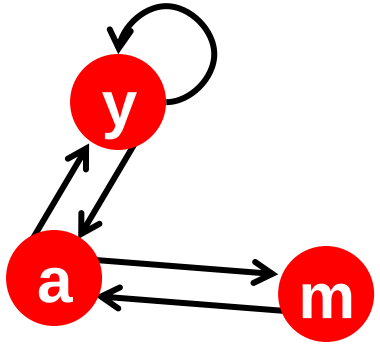


r_y	1/3
r_a	1/3
r_m	1/3

- ماتریس همسایگی M
  - فرض کنید صفحه i دارای d_i لینک خروجی باشد
  - If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
  - جمع هر یک از ستون‌های این ماتریس یک است.
- بردار رتبه r
  - r_i رتبه اهمیت صفحه i است.
  - $\sum_i r_i = 1$
- دستگاه معادلات را می‌توان به صورت زیر نوشت: $r = M \cdot r$

رتبه صفحه (Page Rank)

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



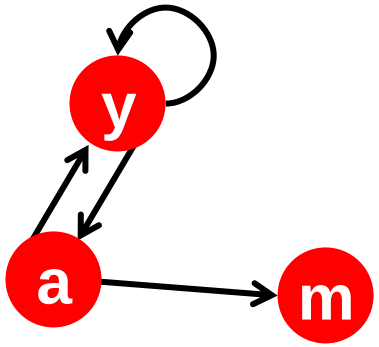
$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 3/6 \\ 1/6 \end{pmatrix}$$

$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \begin{pmatrix} 1/3 \\ 3/6 \\ 1/6 \end{pmatrix} \begin{pmatrix} 5/12 \\ 1/3 \\ 3/12 \end{pmatrix} \begin{pmatrix} 9/24 \\ 11/24 \\ 1/6 \end{pmatrix} \ldots \begin{pmatrix} 6/15 \\ 6/15 \\ 3/15 \end{pmatrix}$$

رتبه صفحه (Page Rank)



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

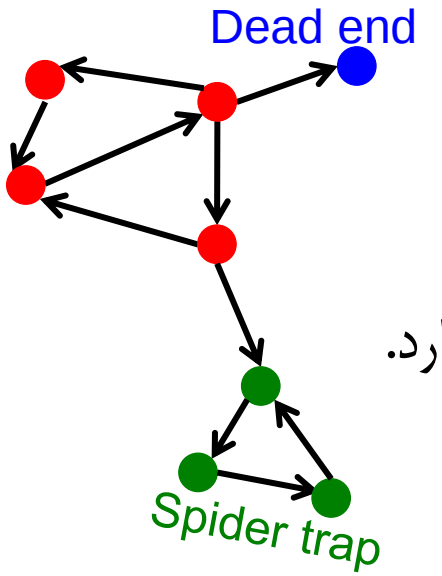
$r_y = r_y/2 + r_a/2$

$r_a = r_y/2$

$r_m = r_a/2$

r_y		1/3	2/6	3/12	5/24		0
r_a	=	1/3	1/6	2/12	3/24	…	0
r_m		1/3	1/6	1/12	2/24		0

رتبه صفحه (Page Rank)

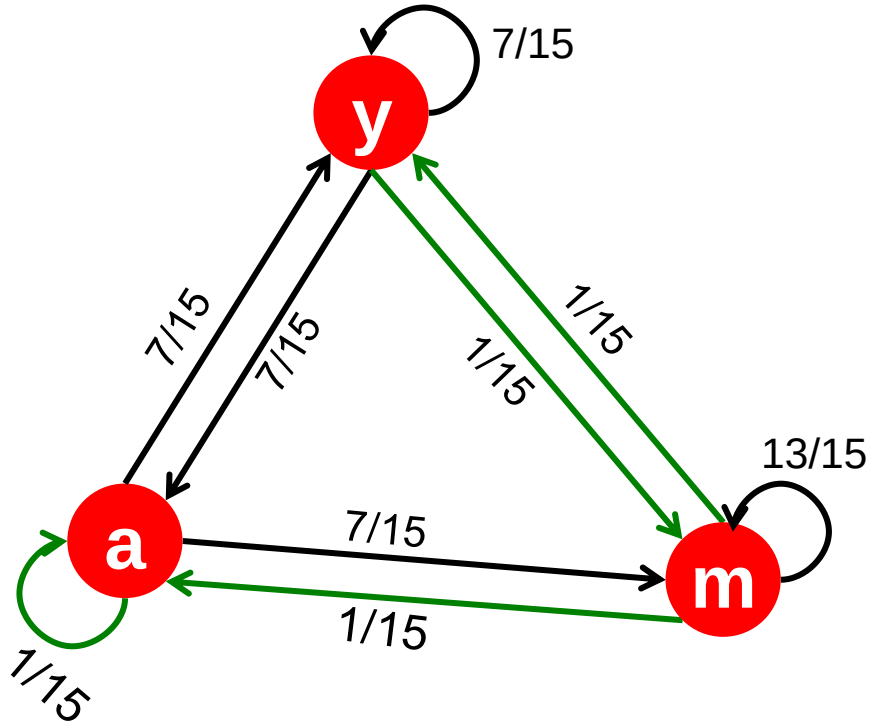


- دو مشکل رتبه صفحه:
  - dead end: قدم زن تصادفی هیچ جایی برای رفتن ندارد.
    - بعد از مدتی اهمیت تمامی صفحات صفر می‌شود.
  - spider tap: قدم زن در یک زیرگرافی گیر می‌کند که هیچ لینک خروجی به بیرون ندارد.
    - بعد از مدتی زیرگراف کل اهمیت را جذب خود می‌کند.
- راه‌حل: استفاده از teleport
  - قدم‌زن با احتمال β یک لینک را به طور تصادفی انتخاب کند
  - با احتمال $1-\beta$ به یک صفحه به‌طور تصادفی پرش کند.

$$r_j = (1-\beta)\frac{1}{N} + \sum_{i \to j} \beta \frac{r_i}{d_i}$$

$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N} \qquad \boldsymbol{r} = \boldsymbol{A}\cdot\boldsymbol{r}$$

رتبه صفحه (Page Rank)



M **$[1/N]_{NxN}$**

$$0.8\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$\begin{array}{c} \\ y \\ a \\ m \end{array}\begin{array}{c} \begin{array}{ccc} y & a & m \end{array} \\ \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix} \end{array}$$

A

$$\begin{array}{c} \\ y \\ a \\ m \end{array}\begin{array}{c} = \\ \\ \\ \end{array}\begin{bmatrix} 1/3 & 0.33 & 0.24 & 0.26 \\ 1/3 & 0.20 & 0.20 & 0.18 \\ 1/3 & 0.46 & 0.52 & 0.56 \end{bmatrix} \ldots \begin{array}{c} 7/33 \\ 5/33 \\ 21/33 \end{array}$$

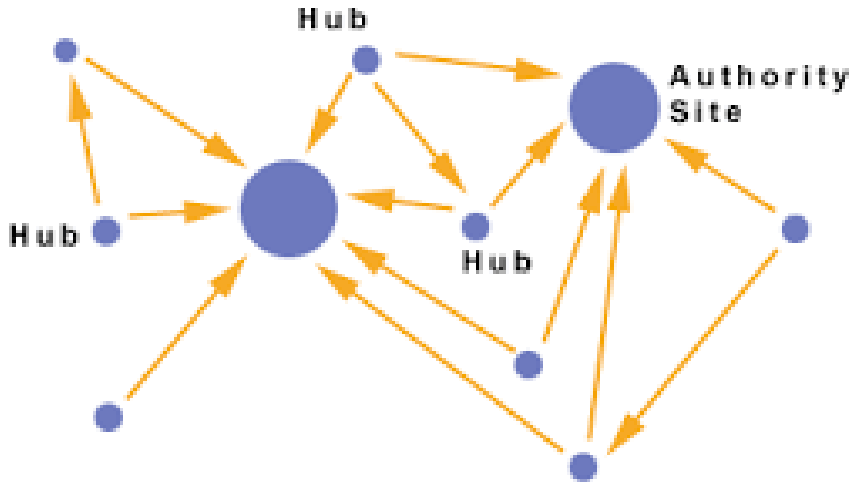
الگوریتم HITS

- یک الگوریتم وابسته به query است.
  - صفحات را وابسته به query داده شده رتبه‌بندی می‌کند. بنابراین برای هر query جدید دوباره باید محاسبه شود.
  - با فرض داشتن یک query، این الگوریتم ابتدا مجموعه صفحات مرتبط را پیدا می‌کند. این کار با اشتراک گرفتن میان hit list های کلمات query و اضافه کردن صفحاتی که به آن‌ها لینک داده‌اند یا از آن‌ها لینک گرفته‌اند انجام می‌شود.
- برای هر صفحه دو معیار محاسبه می‌شود:
  - authority: چقدر صفحات موجود در مجموعه به این صفحه ارجاع می‌دهند.
  - hub: چقدر این صفحه به صفحات معتبر موجود در مجموعه ارجاع می‌دهد.

الگوریتم HITS

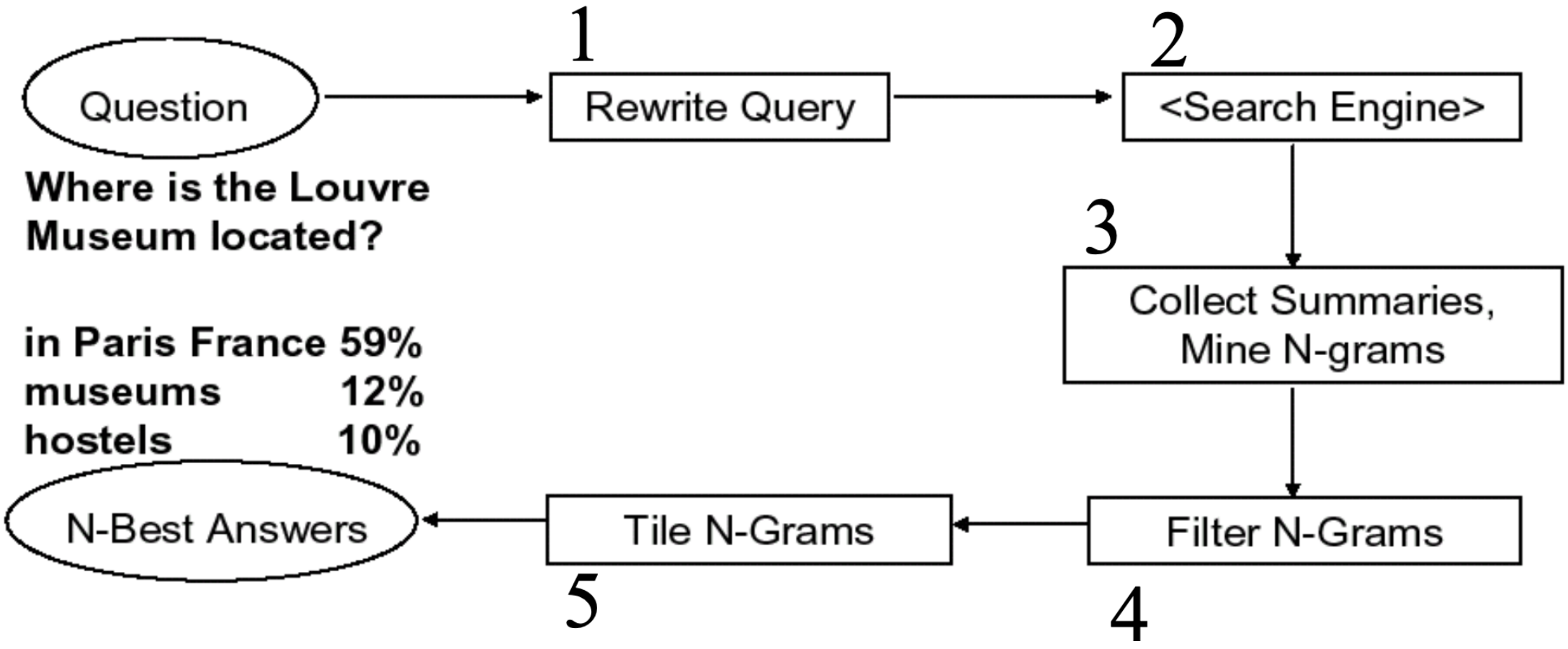
function HITS(*query*) **returns** *pages* with hub and authority numbers

pages ← EXPAND-PAGES(RELEVANT-PAGES(*query*))
for each *p* **in** *pages* **do**
  p.AUTHORITY ← 1
  p.HUB ← 1
repeat until convergence **do**
  for each *p* **in** *pages* **do**
    p.AUTHORITY ← $\sum_i$ INLINK$_i$(*p*).HUB
    p.HUB ← $\sum_i$ OUTLINK$_i$(*p*).AUTHORITY
  NORMALIZE(*pages*)
return *pages*



سامانه های پاسخگویی به پرسش

- درخواست یک سوال است و پاسخ یک عبارت، جمله یا متن کوتاه.



سامانه های پاسخگویی به پرسش

- گام اول: بازنویسی query
- شهود: سوال کاربر معمولا از نظر گرامری بسیار شبیه به جملات پاسخ است.

- Where is the Louvre Museum located?
- The Louvre Museum is located in ***Paris***

- Who created the character of Scrooge?
- ***Charles Dickens*** created the character of Scrooge.

سامانه های پاسخگویی به پرسش

- سوال‌ها به چند دسته تقسیم می‌شوند.

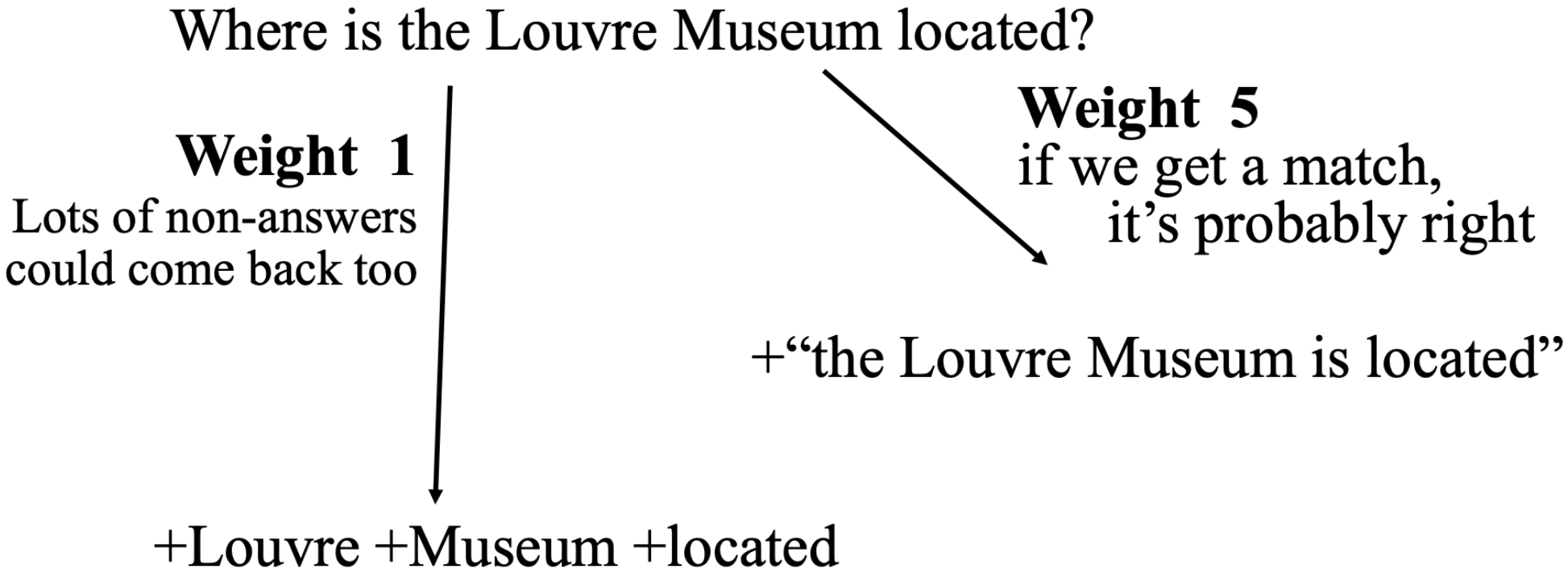
- **Who** is/was/are/were...?
  When is/did/will/are/were ...?
  Where is/are/were ...?

- قوانین تبدیل وابسته به دسته

- eg “For Where questions, move ‘is’ to all possible locations”
- “Where is the Louvre Museum located”
- “is the Louvre Museum located”
- “the is Louvre Museum located”
- “the Louvre is Museum located”
- “the Louvre Museum is located”
- “the Louvre Museum located is”

سامانه های پاسخگویی به پرسش

- وزن‌دهی به queryهای بازنویسی‌شده
- برخی از آن‌ها قابل اعتمادتر هستند.



سامانه های پاسخگویی به پرسش

- گام دوم: موتور جستجوی query
  - تمام query ها را به یک موتور جستجوی وب می‌فرستیم.
  - N پاسخ اول آن‌ها را بازیابی می‌کنیم.
    - نیازی به بازیابی کل سند نیست بخشی از هر سند که مربوط به سوال است مورد استفاده قرار می‌گیرد.
- گام سوم: استخراج n-gramها
  - شمارش تمام n-gramها در همه نتایج برگشتی از موتور جستجو
  - تعداد هر یک از n-gramها در وزن queryای که از آن حاصل شده است، ضرب می‌شود.
- گام چهارم: فیلتر کردن n-gramها
  - نوع داده پاسخ مورد انتظار
    - **When** was the French Revolution? → DATE or TIME

سامانه های پاسخگویی به پرسش

- گام پنجم: ادغام پاسخ‌ها