Търсене и извличане на информация. Приложение на дълбоко машинно обучение

Стоян Михов

Лекция 4: Ранкирано търсене на документи. Езиков модел. Ентропия и перплексия.

План на лекцията

1. **Формалности за курса (5 мин)**
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Формалности

- Засега ще провеждаме занятията онлайн всяка сряда от 8:15 до 12:00 часа.
- Засега ще използваме платформата Google meet: meet.google.com/hue-frfx-axb
- Моля следете редовно обявите в Moodle за евентуални промени.
- Четвъртата лекция се базира на глави 6 и 12 от първия учебник и глава 9 от втория учебник.

План на лекцията

1. Формалности за курса (5 мин)
2. **Ранкирано търсене на информация (10 мин)**
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Ранкирано търсене на информация

- В големи колекции от документи често резултатът от булевото търсене връща хиляди документи, отговарящи на заявката, или нито един.
- За удовлетворяване на информационната потребност най-често е достатъчен само един или няколко от документите.
- Проблемът се състои в намирането и извеждането само на най-релевантните документи по отношение на информационната потребност.
- Задачата се свежда до извеждането само на първите k документа подредени по ранк, който следва да отразява релевантността по отношение на информационната потребност.
- Основната цел при ранкираното търсене е да се спести времето на потребителя при намирането на търсената информация.

Подход към задачата за ранкирано търсене

- Заявката е текст — въпрос, изречение или списък от ключови думи — свързани с информационната потребност.
- Извлича се списък от (всички) документи, които включват (един или повече) от <u>съществените</u> термове от заявката.
- За всеки от документите от извлечения списък се изчислява ранк спрямо заявката.
- Извеждат се най-високо ранкираните k документа от списъка.
- Ключовият проблем е реализирането на релевантна ранкираща функция.

Подходи за ранкиране

- **Базирано на зоните на документа:**
  - идея: ако повече термове от заявката се срещат в заглавието или резюмето на документа, то документът е по-релевантен (виж глава 6 от първия учебник).
- **Евристичен подход:**
  - ако в документа има повече броя срещания на термове от заявката и тези термове са по специфични, то документът е по-релевантен (ще разгледаме по-подробно).
- **Вероятностен модел за релевантността:**
  - (разглежда се в глава 11 от първия учебник).
- **Езиков модел:**
  - документите се ранкират по вероятността съответният езиков модел на документа да генерира заявката (ще разгледаме по-подробно).

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. **Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)**
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Недостатъци на документно представяне в $\{0,1\}^M$ и $\mathbb{N}^M$

- При използването на биномен или мултиномен документен модел векторите, съответстващи на документите, отразяват наличието или броя на срещанията на термове.
- Тези представяния водят до следните недостатъци:
  - Не се отчита специфичността на съответните термове. Стоп думи и термини се третират по еднакъв начин.
  - Броят на срещанията расте линейно (при мултиномен модел), докато човешките сензорни възприятия са логаритмични.
  - Бройките са абсолютни и не зависят от дължината на документите.

Тегло на срещанията

- Дефинираме $\mathrm{tf}_{t,d}$ (term frequency), като броя на срещанията на терма t в документа d.

- Ако даден терм от заявката се среща 10 пъти в документа, то това не означава, че документът е 10 пъти по релевантен. Затова дефинираме теглото на срещанията $\mathrm{w}_{t,d}$ логаритмично:

- $$\mathrm{w}_{t,d} = \begin{cases} 1 + \log \mathrm{tf}_{t,d} & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Тегло на терм

- По-редките термове са по-специфични и носят повече информация.
- Релевантно е колкото по рядко се среща даден терм, толкова по-високо тегло да има. Освен това е по-релевантно да се разглежда не броят на срещанията, а броят на документите, в които се среща даденият терм.
- Нека df_t (document frequency) е броят на документите, в които се среща термът t. Дефинираме обратната документна честота idf_t като:
- $$\mathrm{idf}_t = \log \frac{N}{\mathrm{df}_t}$$

Тегло $\mathrm{tf{\cdot}idf}$

- Дефинираме теглото $\mathrm{tf{\cdot}idf}$ като произведението на теглото на срещанията с теглото на терма:

$$\mathrm{tf{\cdot}idf}_{t,d} = \log(1 + \mathrm{tf}_{t,d}) \times \log \frac{N}{\mathrm{df}_t}$$

- Това е най-известното тегло за ранкиране в търсенето на информация. Често се изписва като $\mathrm{tf\text{-}idf}$ или $\mathrm{tf} \times \mathrm{idf}$.

Документно представяне чрез вектори от тегла

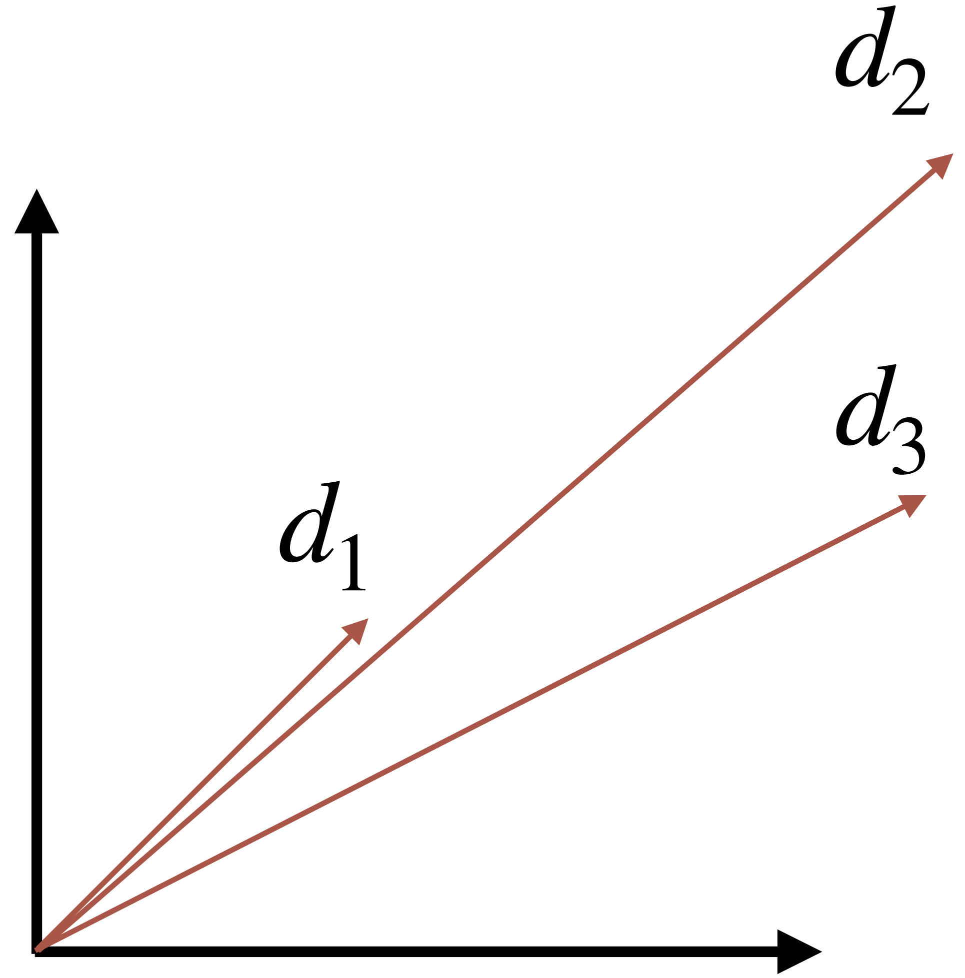
- На всеки документ съпоставяме вектор от $M = |V|$ тегла $\mathrm{tf{\cdot}idf}$. Тогава $d \in \mathbb{R}^{+M}$.
- Векторите са много разредени — повечето елементи са 0.
- Ключова идея:
  - Семантичната близост на два документа свеждаме до близост между съответните им вектори.
  - Ранкираме документите в зависимост от близостта им до вектора, представящ заявката.

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. **Ранкиране при документно представяне чрез вектори от тегла (10 мин)**
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Проблеми при Евклидово разстояние

- Разстоянието зависи от броя на думите в документите.
- Семантично по-релевантно е да се използва за близост ъгълът между документите
- Малкият ъгъл между два документа съответсва на близко честотно разпределение на съществените термове в двата документа.



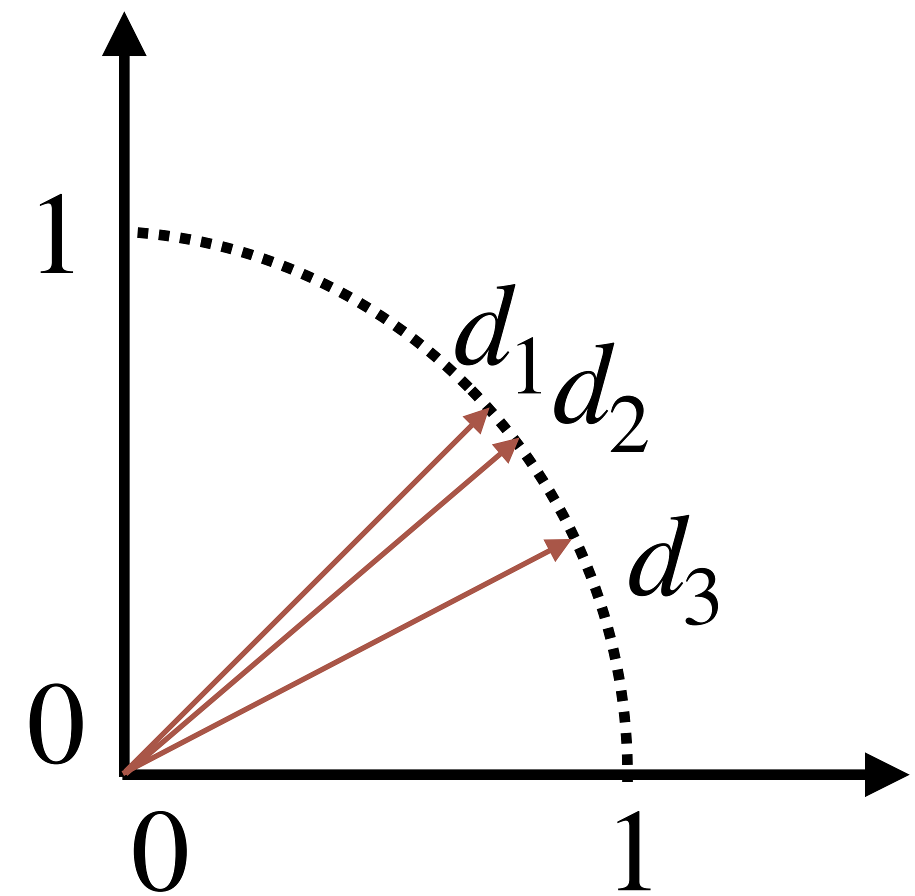
Косинусова близост

- В интервала $\left[0, \frac{\pi}{2}\right]$ функцията косинус е монотонно намаляваща
- Вместо ъгъла е изчислително по-удобно да намираме косинуса между векторите:

$$\cos(\vec{q}, \vec{d}) = \frac{\vec{q} \,.\, \vec{d}}{|\vec{q}||\vec{d}|} = \frac{\sum_{i=1}^{M} q_i d_i}{\sqrt{\sum_{i=1}^{M} q_i^2}\sqrt{\sum_{i=1}^{M} d_i^2}}$$

- Алтернативно, ако документите са нормализирани, така че всички вектори да са с дължина 1, то косинусът е равен на декартовото произведение между двата вектора:

$$\cos(\vec{q}, \vec{d}) = \sum_{i=1}^{M} q_i d_i, \text{ ако } |\vec{q}| = |\vec{d}| = 1$$



Алгоритъм за ранкирано търсене

```
CosineScore(q)
float Scores[N] = 0
Initialize Length[N]
for each query term t do
    calculate w_{t,q} and fetch postings list for t
    for each pair(d, tf_{t,d}) in postings list do
        Scores[d] += wf_{t,d} × w_{t,q}
Read the array Length[d]
for each d do
    Scores[d] = Scores[d]/Length[d]
return Top K components of Scores[]
```

Забележки по ефективността

```
CosineScore(q)
float Scores[N] = 0
Initialize Length[N]
for each query term t do
    calculate w_{t,q} and fetch postings list for t
    for each pair(d, tf_{t,d}) in postings list do
        Scores[d] += wf_{t,d} × w_{t,q}
Read the array Length[d]
for each d do
    Scores[d] = Scores[d]/Length[d]
return Top K components of Scores[]
```

- Може да съхраняваме нормализирани тегла и да отпадне делението на дължината.
- Вместо тегла (с плаваща запетая) може да съхраняваме брой срещания, за да пестим памет.
- Обратната документна честота можем да съхраняваме в речника за термовете.
- Най-високо ранкираните K документа можем да получим ефективно с приоритетна опашка (разглежда се в курса БАСД).

Варианти на теглото tf·idf

term frequency		document frequency		normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log\frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0,\log\frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

ltc

- Заявката и документите може да имат различни схеми за претегляне
- Нотацията SMART — ddd.qqq — например ltc.lnn

Пример за евристично ранкиране

Терм	Заявка				Документ		Произведение
	tf	df	idf	$w_{t,q}$	tf	$w_{t,d}$	
автомобил	0	5000	2.3	0	1	0.41	0
добра	1	50000	1.3	1.3	0	0	0
застраховка	1	1000	3.0	3.0	2	0.82	2.46
кола	1	10000	2.0	2.0	1	0.41	0.82

- Пример за ранкиране при заявка:
  “добра застраховка кола” в колекция от N=1000000 документа.
- Използва се SMART схема **nnc.btn**
- При даден документ с две срещания на термовете “застраховка” и по едно срещане на “кола” и “ автомобил” се получава ранк:
  $$\text{score}(q,d) = 0 \times 0.41 + 1.3 \times 0 + 3.0 \times 0.82 + 2.0 \times 0.41 = 3.28$$

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf}\cdot\mathrm{idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. **Езикови модели (10 мин)**
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Моделиране на езика

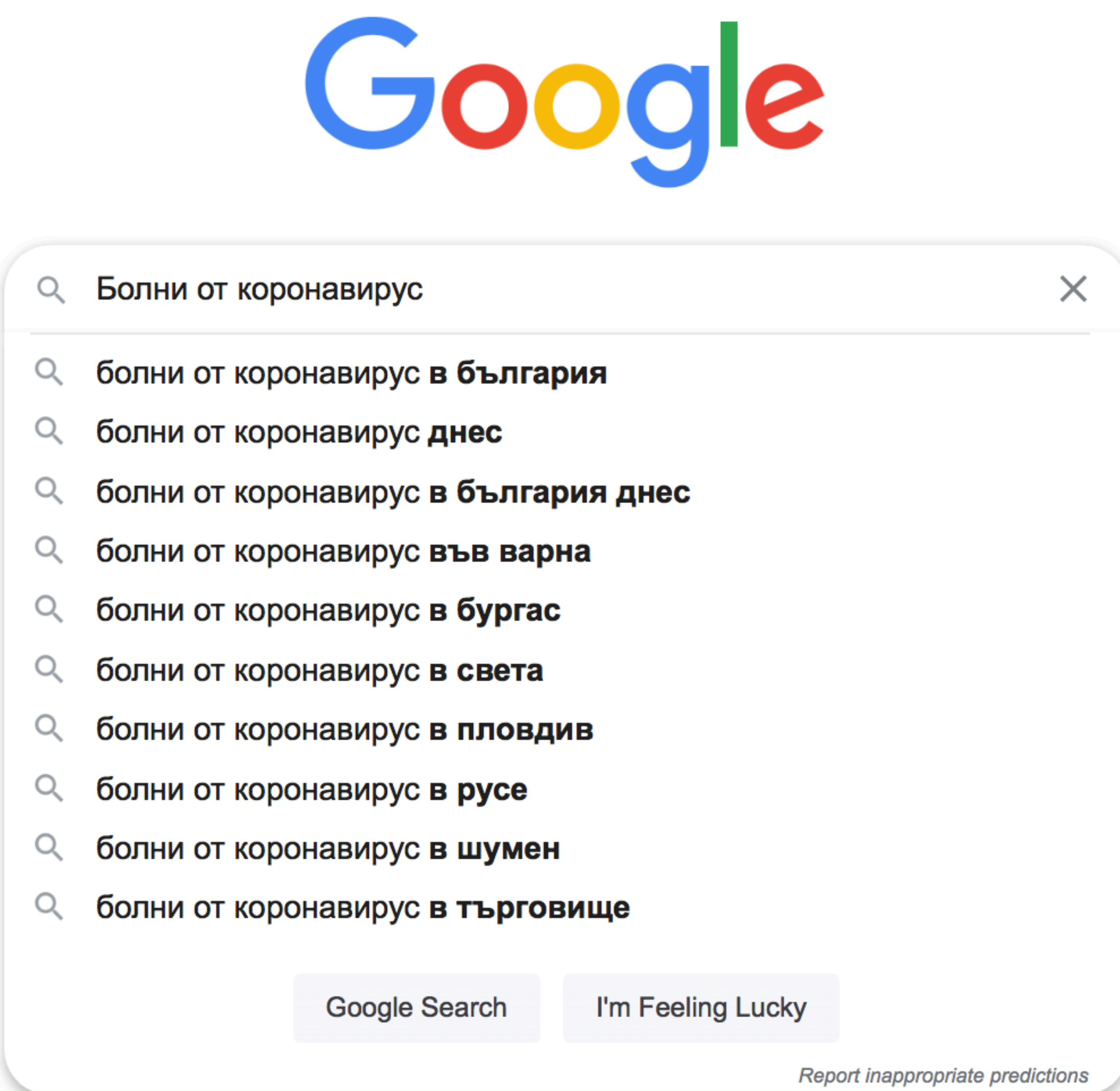
- Моделиране на езика е задачата да се определи вероятност на изреченията от езика, която да отразява вероятността да наблюдаваме даденото изречение.

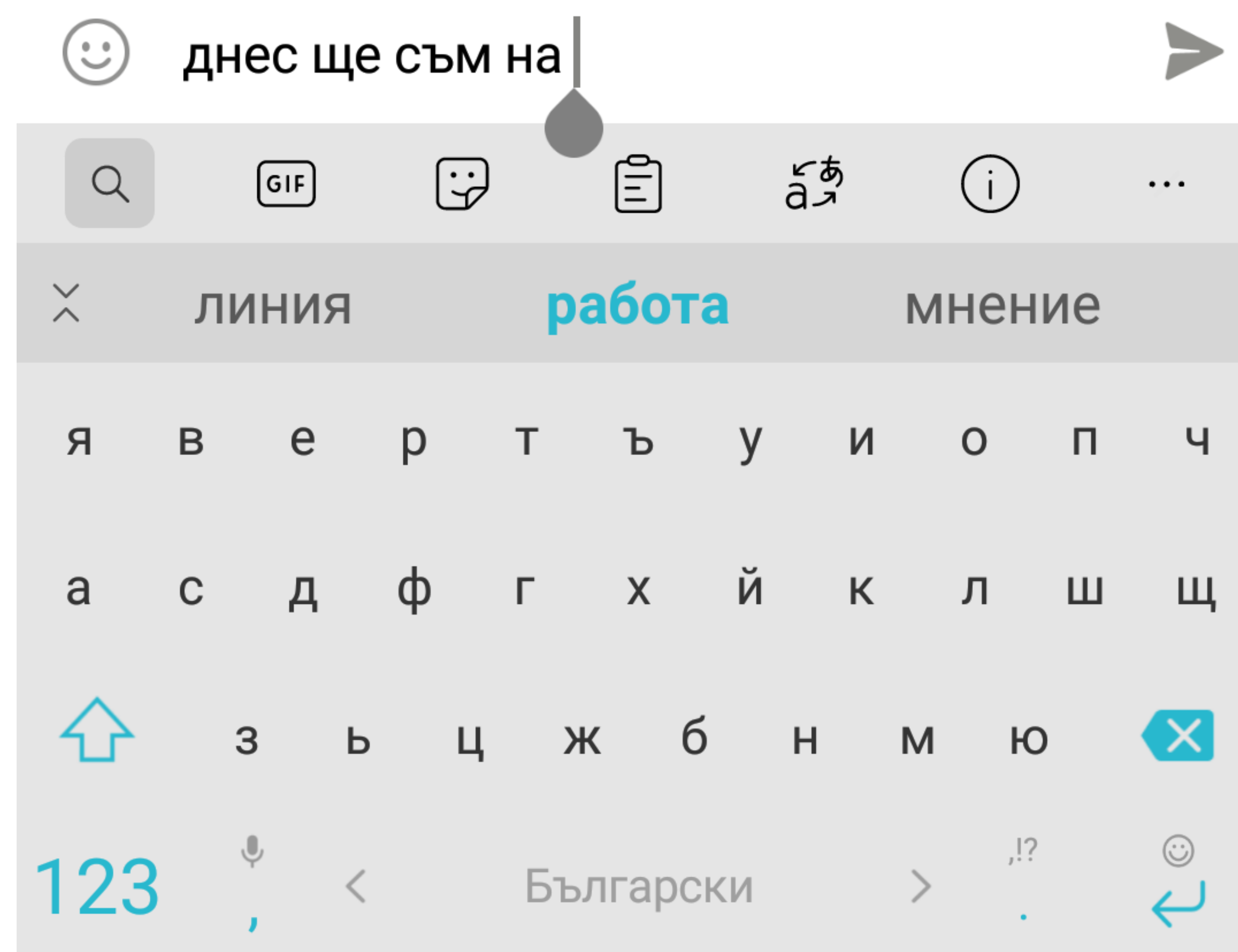
    - Например, каква е вероятността да наблюдаваме изречението “*Черното куче подгони котката*”?

- Еквивалентна задача е за дадено начало на изречение да се даде вероятност на всяка дума от езика да следва даденото начало.

    - Например, каква е вероятността да наблюдаваме думата “*самолет*” след начало “*Черното куче подгони*”?

Приложения на езиков модел





Еквивалентност

- Означаваме:
  $\Pr[x_1x_2\ldots x_n] := \Pr[X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n] = \Pr[X_1 = x_1 \cap X_2 = x_2 \cap \ldots \cap X_n = x_n]$ като вероятностното пространство Ω са всички крайни последователности от думи над речник V, такива че завършват със специален символ за край на последователност "$" и не съдържат друго срещане на символа "$". Случайната величина X_i отразява i-тата дума в изречението — $X_i = x_i$ означава, че случайната величина X_i приема стойност x_i (приемаме, че x_i е индексът на съответната дума в речника).

- От Верижното правило следва:

- $\Pr[x_1x_2\ldots x_n] = \Pr[x_1]\Pr[x_2 \mid x_1]\Pr[x_3 \mid x_1x_2]\ldots\Pr[x_n \mid x_1x_2\ldots x_{n-1}]$

- От друга страна, от дефиницията за условна вероятност следва:

- $\Pr[x_n \mid x_1x_2\ldots x_{n-1}] = \dfrac{\Pr[x_1x_2\ldots x_n]}{\Pr[x_1x_2\ldots x_{n-1}]}$

- По-нататък под езиков модел ще разбираме система (функция, алгоритъм, метод), която за всяка начална последователност от думи $x_1x_2\ldots x_{n-1}$ ни връща вероятностно разпределение дефинирано върху думите от речника V, отразяващо условната вероятност $\Pr[x \mid x_1x_2\ldots x_{n-1}]$ за всяко $x \in V$.

Свойства на езиковите модели

- За всяка начална последователност от думи $x_1x_2\ldots x_{n-1}$, такава че $\$ \notin \{x_1, x_2, \ldots, x_{n-1}\}$, е в сила $\sum_{x\in V} \Pr[x \,|\, x_1x_2\ldots x_{n-1}] = 1$.
- Винаги предполагаме, че $\$ \in V$.
- В такъв случай трябва де е в сила:

$$\sum_{\mathbf{x}\in\Omega} \Pr[\mathbf{x}] = 1$$

- **Задача**:
  а) Да се докаже, че ако за всяко $n \in \mathbb{N}^+$ и за всяка подпоследователност от думи $x_1x_2\ldots x_{n-1}$, такава че $\$ \notin \{x_1, x_2, \ldots, x_{n-1}\}$, е в сила $\sum_{x\in V} \Pr[x \,|\, x_1x_2\ldots x_{n-1}] = 1$ и $\Pr[\$ \,|\, x_1x_2\ldots x_{n-1}] = \alpha$ за някое фиксирано $\alpha \in (0,1)$, то е изпълнено $\sum_{\mathbf{x}\in\Omega} \Pr[\mathbf{x}] = 1$.
  б) *** Може ли да се докаже равенството ако α не е фиксирано, т.е. ако α зависи от $x_1x_2\ldots x_{n-1}$?

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. **Ентропия, перплексия и оценяване на езиков модел (20 мин)**
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. Ранкиране чрез документни езикови модели (10 мин)

Пример: задача за компресиране

- Нека имаме 8 състезателни коня — A, B, C, D, E, F, G, H. Вероятността за печалба на даден кон е:

A	B	C	D	E	F	G	H
1/2	1/4	1/8	1/16	1/64	1/64	1/64	1/64

- Нека са проведени n състезания между конете и сме записали резултатите от надбягванията. Колко най-малко памет ни е нужна?

- Наивен подход— за обозначаването на даден кон от 8 възможни са ни нужни 3 бита. Следователно ще ни трябват $3n$ бита.

- Можем ли да подобрим представянето на резултатите, като се възползваме, че кон A ще се среща много по-често от кон F. Можем ли да представим A с по-малък брой битове за сметка на другите коне?

Решение: Код на Хъфман

A	B	C	D	E	F	G	H
1/2	1/4	1/8	1/16	1/64	1/64	1/64	1/64
0	10	110	1110	111100	111101	111110	111111

- Код на Хъфман $h : \Sigma \to \{0,1\}^*$

    1. Префиксен код — никой код не е префикс на друг и следователно всяка последоветелност от кодове позволява еднозначно декодиране.

    2. За всеки символ σ е изпълнено $|h(\sigma)| = \lceil -\log_2 \Pr[\sigma] \rceil$

- Горните свойства могат да бъдат удовлетворени за всяко крайно разпределение

Очакване за размера на представянето с кодиране на Хъфман

- Очакваме, че при всеки n надбягвания, конят σ ще спечели средно $n \Pr[\sigma]$ надбягвания.

- Тогава очакването за размера на представянето е:

$$\sum_{\sigma \in \Sigma} n \Pr[\sigma] \, | \, h(\sigma) \, | = - n \sum_{\sigma \in \Sigma} \Pr[\sigma] \log_2 \Pr[\sigma] =$$

$$= - n \left(\frac{1}{2} \log_2 \frac{1}{2} + \frac{1}{4} \log_2 \frac{1}{4} + \frac{1}{8} \log_2 \frac{1}{8} + \frac{1}{16} \log_2 \frac{1}{16} + \frac{4}{64} \log_2 \frac{1}{64} \right) =$$

$$= 2n$$

- Доказва се, че при условие че набюденията са независими и еднакво разпределени, кодирането на Хъфман е оптимално (показва се в курса ОСОЕ).

- Ако разпределението беше равномерно, то получаваме представяне с $3n$ бита, което изисква най-много памет.

Ентропия

- **Ентропията** на дадена случайна величина X наричаме:

$$H_X = -\sum_{x \in X(\Omega)} \Pr[x]\ \log_2 \Pr[x]$$

- Ентропията е мярка за очакваното (средното) количество информация (брой битове) за представяне на резултат на случаен опит. Тя е мярка за неопределеността, хаоса или изненадата на дадена случайна величина.

- Очакваната памет (в брой битове) необходима за предаването на n резултата от случаен опит на случайна величина X е nH_X.

- Ентропията възниква естествено в различни области като теория на вероятностите, теория на информацията, термодинамиката, и други.

Релативна ентропия и крос-ентропия

- **Крос-ентропия** на двете функции на разпределение $\Pr$ и $\hat{\Pr}$ на случайната величина X дефинираме като:
- $$H_X(\Pr \| \hat{\Pr}) = -\sum_x \Pr[x] \, \log_2 \hat{\Pr}[x]$$
- Крос-ентропията измерва очаквания брой битове, необходими за предаването на резултат от случаен опит, ако вместо действителната функция на разпределение на случайната величина $\Pr$, за кодиране се използва функцията на разпределение $\hat{\Pr}$.
- **Релативната ентропия** (разстояние на Кулбек-Лайблер) на двете функции на разпределение $\Pr$ и $\hat{\Pr}$ на случайната величина X дефинираме като:
- $$D(\Pr \| \hat{\Pr}) = H_X(\Pr \| \hat{\Pr}) - H_X = \sum_x \Pr[x] \, \log_2 \frac{\Pr[x]}{\hat{\Pr}[x]}$$
- Релативната ентропия измерва доколко функцията на разпределени $\Pr$ се различава от $\hat{\Pr}$.
- **Теорема**: $D(\Pr \,|\,|\hat{\Pr}) \geq 0$, като равенство се достига т.с.т.к. $\Pr[x] = \hat{\Pr}[x]$ за всяко $x \in X(\Omega)$. (Доказателство в курса ОСОЕ)

Оценка на езиков модел

- Нека е даден езиков модел M с разпределение $\hat{\Pr}[x_n \mid x_1 x_2 \ldots x_{n-1}]$

- **<u>Задача</u>**: Как да оценим езиковия модел? **<u>Идея</u>**: Да измерим крос-ентропията спрямо истинското разпределение.

- За да оценим действителното разпределение на езика, ще използваме достатъчно голям корпус от текстове $x_1 x_2 \ldots x_m$ (често броят на думите в корпуса m е в порядък от милиони думи). Корпусът за оценяване не трябва да е използван за обучението на модела.

Перплексия

- **Перплексията** на езиковия модел M дефинираме като: $2^{-\frac{1}{m}\sum_{n=1}^{m}\log_2 \hat{\Pr}[x_n|x_1x_2\ldots x_{n-1}]}$.
- $-\dfrac{1}{m}\displaystyle\sum_{n=1}^{m}\log_2 \hat{\Pr}[x_n\,|\,x_1x_2\ldots x_{n-1}]$ оценява емпирично ***скоростта на крос-ентропията*** (Cross Entropy Rate) между действителното разпределение на езика $\Pr$ и разпределението дадено от езиковия модел $\hat{\Pr}$.
- Добрият езиков модел следва да даде високи вероятности на наблюденията в корпуса, което води до по-ниска крос-ентропия и съответно перплексия.
- Емпиричната скорост на крос-ентропията е най-често използваната целева функция в дълбокото машинно обучение.

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. **n-грамни езикови модели и изглаждане на езиков модел (15 мин)**
8. Ранкиране чрез документни езикови модели (10 мин)

Марковско предположение

- Марковско предположение от ред k:
  Вероятността за следващата дума зависи само от предишните $k-1$ думи. Т.е.

$$\Pr[x_n \mid x_1 x_2 \ldots x_{n-1}] = \Pr[x_n \mid x_{n-k+1} x_{n-k+2} \ldots x_{n-1}]$$

- За улеснение добавяме пред последователността $k-1$ символа "^".

- Например, ако премем Марковско предположение от ред 2, получаваме:

$$\Pr[x_1 x_2 \ldots x_n] = \Pr[x_1] \Pr[x_2 \mid x_1] \Pr[x_3 \mid x_1 x_2] \Pr[x_4 \mid x_1 x_2 x_3] \ldots \Pr[x_n \mid x_1 x_2 \ldots x_{n-1}] =$$
$$= \Pr[x_1 \mid \hat{}\,] \Pr[x_2 \mid x_1] \Pr[x_3 \mid x_2] \Pr[x_4 \mid x_3] \ldots \Pr[x_n \mid x_{n-1}]$$

Марковски езиков модел

- Марковски езиков модел от ред k наричаме езиков модел, в който е сила Марковското предположение от ред k.

- Марковският езиков модел от ред k се определя от условните вероятности $\Pr[x_k \mid x_1 x_2 \ldots x_{k-1}]$ за всички последователности $x_1 x_2 \ldots x_k$ на думи от V.

- Марковските езикови модели са широко използвани поради простотата и изчислителната ефективност.

- По-модерните методи базирани на дълбоки невронни мрежи представят не-Марковски езикови модели — ще разгледаме по-нататък в курса.

- Ефективни изчислителни методи за представяне на Марковски езиков модел се базират на претеглени крайни преобразуватели (WFST — Weighted Finite-State Transducers), които се разглеждат в курса по ПКА.

Обучение на Марковски езиков модел

- Принцип за максимизиране на правдоподобието:
- $$\hat{\Pr}_{MLE}[x_k \,|\, x_1 x_2 \ldots x_{k-1}] = \frac{\#(x_1 x_2 \ldots x_k)}{\#(x_1 x_2 \ldots x_{k-1} \bullet)}$$
- Свеждаме обучението до броене на срещания на k-орки в корпус от текстове.
- **Недостатък**: Ако дадена k-орка не се е срещнала в корпуса, то съответната вероятност $= 0$.

Изглаждане на Марковски езиков модел

- Броят на различните k-орки от думи расте експоненциално с k. При речник от 30000 думи има 900 000 000 биграми и 27 000 000 000 000 триграми.

- Неправилно е да предполагаме, че всички k-орки от думи от езика са се срещали.

- Най-просто изглаждане — add α изглаждане:
$$\hat{\Pr}_{\mathrm{add}\alpha}[x_k \,|\, x_1 x_2 \ldots x_{k-1}] = \frac{\#(x_1 x_2 \ldots x_k) + \alpha}{\#(x_1 x_2 \ldots x_{k-1} \bullet) + \alpha \,|\, V \,|}$$

- При $\alpha = 1$ получаваме изглаждане на Лаплас

- **<u>Проблем</u>**: изглаждането add α дава една и съща вероятност на всички k-орки, които не са се срещнали в корпуса.

Изглаждане **back-off**

- **Идея**: Вероятностите на k-орките, които не са се срещнали, е пропорционална на вероятностите на съответните $k-1$-орки.

- Изглаждане на Катц:

$$\hat{\Pr}_{\mathrm{bo}}[x_k \mid x_1 x_2 \ldots x_{k-1}] = \begin{cases} d\,\dfrac{\#(x_1 x_2 \ldots x_k)}{\#(x_1 x_2 \ldots x_{k-1} \bullet)} & \text{if } \#(x_1 x_2 \ldots x_k) > 0 \\ \alpha \hat{\Pr}_{\mathrm{bo}}[x_k \mid x_2 x_3 \ldots x_{k-1}] & \text{otherwise} \end{cases}$$

- Изглаждане с интерполация на Йелинек-Мерсер (Jelinek-Mercer interpolated smoothing):

$$\hat{\Pr}_{\mathrm{int}}[x_k \mid x_1 x_2 \ldots x_{k-1}] = \lambda \frac{\#(x_1 x_2 \ldots x_k)}{\#(x_1 x_2 \ldots x_{k-1} \bullet)} + (1-\lambda)\hat{\Pr}_{\mathrm{int}}[x_k \mid x_2 x_3 \ldots x_{k-1}]$$

- Параметрите при изглаждането d, α, λ се настройват, така че да се получи вероятностно разпределение и да минимизира перплексията.

- Съвременната (най-успешна) техника за изглаждане на k-грамен езиков модел е модифицираното изглаждане на Кнесер-Ней (modified Knesser-Ney smoothing).

Пример за двуграмен Марковски езиков модел и изглаждане с интерполация

Корпус

^ Иван кара кола $

^ Мария кара $

^ Иван гони Мария $

^ Мария купи кола $

^ Мария кара колело $

Монограми	Брой	Биграми	Брой
Иван	2	Иван кара	1
Мария	4	Мария кара	2
кара	3	Иван гони	1
купи	1	Мария купи	1
гони	1	кара кола	1
кола	2	кара колело	1
колело	1	гони Мария	1
^	5	купи кола	1
$	5	^ Мария	3
Общо	**24**	^ Иван	2
		кара $	1
		кола $	2
		Мария $	1
		колело $	1
		Общо	**19**

При $\lambda = 0.75$:

$$\Pr[\text{^ Мария кара кола \$}] = \Pr[\text{Мария} \,|\, \text{^}]\,\Pr[\text{кара} \,|\, \text{Мария}]\,\Pr[\text{кола} \,|\, \text{кара}]\,\Pr[\text{\$} \,|\, \text{кола}] =$$

$$= \left(\lambda\frac{3}{19} + (1-\lambda)\frac{4}{24}\right)\left(\lambda\frac{2}{19} + (1-\lambda)\frac{3}{24}\right)\left(\lambda\frac{1}{19} + (1-\lambda)\frac{2}{24}\right)\left(\lambda\frac{2}{19} + (1-\lambda)\frac{5}{24}\right) =$$

$$= 0.0001484730727$$

План на лекцията

1. Формалности за курса (5 мин)
2. Ранкирано търсене на информация (10 мин)
3. Документно представяне чрез вектори от тегла $\mathrm{tf{\cdot}idf}$ (10 мин)
4. Ранкиране при документно представяне чрез вектори от тегла (10 мин)
5. Езикови модели (10 мин)
6. Ентропия, перплексия и оценяване на езиков модел (20 мин)
7. n-грамни езикови модели и изглаждане на езиков модел (15 мин)
8. **Ранкиране чрез документни езикови модели (10 мин)**

Ранкиране с езиков модел

- **ИДЕЯ**:
  - Всеки документ дефинира езиков модел.
  - Използвайки езиковия модел на даден документ намираме вероятността заявката да бъде генерирана от този езиков модел.
  - Ранкираме документите по вероятността да генерират дадената заявка.

Формализация на ранкирането с езиков модел

- Търсим $\hat{d} = \arg\max_d \Pr[d\,|\,q]$
- $$\Pr[d\,|\,q] = \frac{\Pr[q\,|\,d]\Pr[d]}{\Pr[q]}$$
- Вероятността $\Pr[q]$ е фиксирана и не зависи от d, затова я игнорираме.
- Вероятността $\Pr[d]$ или приемаме за константа — т.е. всички документи са равновероятни и съответно я игнорираме, или използваме априорни вероятности, базирани на критерии като авторитетност на документа, дължина, жанр, кога е създаден, и брой ползватели, които са го достъпвали.

- За да намерим $\Pr[q \mid d]$, ще използваме езиков модел M_d извлечен от документа d.

- Индивидуалните документи в дадена колекция обикновено са сравнително къси (например около 1000 думи). Поради това обикновено се използва Марковски модел от ред 1 (монограмен модел).

- Монограмният езиков модел е еквивалентен на мултиномния наивен Бейсов модел (от предишната лекция), като всеки документ се третира като отделен клас. При този модел имаме:

$$\Pr[q \mid M_d] = \prod_{t \in V} \Pr[t \mid M_d]^{\mathrm{tf}_{t,q}},$$

Оценяване на документен монограмен езиков модел

- За да оценим параметрите на модела, използваме принципа за максимизиране на правдоподобието: $\hat{\Pr}_{MLE}[t \mid M_d] = \frac{\mathrm{tf}_{t,d}}{L_d}$
- За да изгладим разпределението, ще използваме линейна интерполация между два езикови модела: $\hat{\Pr}[t \mid M_d] = \lambda \hat{\Pr}_{MLE}[t \mid M_d] + (1 - \lambda)\hat{\Pr}_{MLE}[t \mid M_C]$, където M_C е монограмният езиков модел извлечен от цялата колекция.
- Параметърът $\lambda \in (0,1)$ следва да бъде внимателно настроен. Често λ се настройва да зависи от дължината на q. При по-къси заявки се избира по-висока стойност, за да се засили значението всички термове от заявката да се срещат в документа.
- Ролята на изглаждането не е само за избягване на нулеви вероятности, но води и до подобряване качеството на модела.

Обобщение

1. Извличаме всички документи от колекцията, в които се среща някой от термовете на заявката

2. Изчисляваме за всеки документ d:
   $$\Pr[d\,|\,q] \propto \Pr[d]\,\Pr[q\,|\,d] = \Pr[d]\prod_{t\in q}(\lambda\,\Pr[t\,|\,M_d] + (1-\lambda)\,\Pr[t\,|\,M_C])$$

3. Извеждаме първите k на брой документа подредени по вероятността да генерират заявката.

Забележка: Вместо да се ранкират документите по $\Pr[q\,|\,d]$, по-добри резултати се получават като се ранкират обратно пропорционално на релативната ентропия:

$$D(M_q\,||\,M_d) = \sum_{t\in V}\Pr[t\,|\,M_q]\,\log_2\frac{\Pr[t\,|\,M_q]}{\Pr[t\,|\,M_d]}$$