Обработка на естествен език (ЕЕ) и Information Retrieval

Терминът *Information Retrieval* (IR) се използва днес като традиционно събирателно название на онези задачи, които се занимават най-общо с намиране, извличане на информация или класифициране на текстови документи в големи електронни архиви[1] (наред с текста, извличането на информация от реч, изображения, аудио и видео става все по-актуална област, и при фокусиране върху мултимедийно съдържание се говори за *Content Retrieval*).

Компютърната лингвистика по принцип се занимава с моделиране и автоматична обработка на лингвистични единици; предполага се, че се построява модел на текстовите елементи поне с разглеждане на езикови характеристики, които позволяват изработка на адекватни stemmers (програми за премахване на окончанията и свеждане на думите към „основи"). Класически учебник за компютърни лингвисти е [1]; за да се подчертае разликата с подходите, които не се занимават на лингвистични модели, в компютърната лингвистика се говори за „статистически методи за обработка на естествен език (ЕЕ)". А класически учебник по IR е [2]; в него анализът на текста започва с преброяване на честотите на думите.

Очевидно най-богатият, истински, неподреден, суров и най-предизвикателен масив са текстовете в Интернет вкл. блогове, социални межи и т.н. Появяват се и много големи многоезични ресурси, структурирани в някаква степен: например Wikipedia. Известно е обаче, че сегашните търсещи машини работят само по ключови думи. Не можем да задаваме към Интернет въпроси от рода на *Кои са документите, "най-близки" до текста, отворен на екрана в момента?* (т.е. задача за търсене на подобни по смисъл документи) или заявки като например *Класифицирай ми в 10 групи наличните html-страници на български фирми* (задача за клъстеризация) и т.н. Известно е също така, че търсенето по ключови думи дава много шум. В тази лекция ще се опитаме да разясним коя е основната причина за този негативен ефект, какъв е пътят за достигане на по-прецизни резултати при търсене и защо те са трудно постижими при индустриални системи, които работят над големи архиви в реално време. /По прицнип курсът е фокусиран върху автоматичната обработка на естествен език и затова извличането на информация в тесен смисъл не е обект на нашите разглеждания./

Знаем, че компютърната лингвистика през последните 50 години е произвела само отделни (лабораторни) прототипи за разбиране на естествен език. Знаем и причината за тези незадоволителни практически резултати - при класическата постановка за "разбиране" се изискват много сложни процедури за морфологичен и синтактичен анализ, а семантичният анализ на текста се сблъсква с непреодолими засега трудности при интерпретацията в контекста на вложеното в системата знание за света, поради липсата на алгоритмичен модел

1 Тези и други подобни задачи (напр. търсене на близост между документи) често се решават чрез т. нар. *text mining*. При *mining* -"копаене", за разлика от "разбиране" - целта е системата да намира различни единици в текста без детайлен анализ на естественя език. Обикновено не се строи и поддържа вътрешно представяне на текста, в което думите са представени чрез семантични единици; практиката е да се работи направо над буквените низове в текста и документът да се разглежда като „торба от думи" (bag of words).

на комуникативните цели на човешкия диалог и т.н. Всички тези алгоритми работят със съмнителен успех над реален текст, който - както знаем от демонстрациите - съдържа какви ли не низове, референции към таблици, картинки и т.н. На този фон през 80-те години се появиха извънредно големи текстови ресурси, включително тези в Интернет, и това изведе на преден план прагматичния подход към обработката на езика предимно със статистически методи. Идеята е: след като компютърът не може да анализира естествения език и да го "разбере напълно" в реално време, то нека се ограничим с"приблизително разбиране" - например да сведем значението на текста до набор от значещи единици - и да обработваме ефективно само тях. Да видим какво означава това на практика.

Нека разгледаме изречението *Мария отива на кино*. След премахване на незначещите (свързващи) думи то се свежда до значещите единици {*мария*, *отива*, *кино*}. Към тях се свеждат и изреченията *Мария не отива на кино*. *Мария отива към киното*. *Мария отива до киното*. *На кино ли ще отива Мария?* Съдържанието на документа обикновено се третира като съвкупност от неподредени значещи думи, като се пренебрегват връзките между тях и се предполага, че подобните документи са изградени от едни и същи думи. Това предположение изглежда особено правдоподобно при специализирани текстове, тъй като сходните документи би трябвало да съдържат едни и същи термини. Горните изречения наистина са подобни - в крайна сметка защо не? - понеже в тях става дума за *Мария*, *отиване* и *кино*. Ако не се идентифицират синоними и парафрази, няма начин да се разпознае подобието между документите, но за професионални текстове приближението е задоволително. За жалост се оказва, че дори тази естествена и проста мярка за близост между документите трудно работи на практика, тъй като разпознаването на едините в текста не е елементарна задача. В тази лекция ще се спрем на предварителната обработка на текста, преди той да бъде подаден към компонента за статистическа обработка, и на основния метод за пресмятане на близост между документи - т.нар. Vector Space Model.

Обща постановка на IR-задачата

Интересният модел днес е ad-hoc retrieval, тоест отговор на въпроси въвеждани непосредствено, които не са известни предварително на търсещата система (забележете, че в някои системи заявките може да са изброени в предварително зададено меню и потребителят да избира една или друга). Има два главни модела за търсене в Интернет - (1) *exact match*, където системата дава като отговор документи съдържащи точно думите от въпроса, от този вид системи познаваме булевото търсене по ключови думи, и (2) извличане на отговора с *подреждане на документите по релевантност спрямо въпроса*. Ще се спрем на втория вид системи, при които е актуален въпросът за обработка на лингвистичните единици в текста. Очевидно основните проблеми са поне два: (1) как да сведем думите/документа до значещи единици (тоест, какво броим) и (2) как да изберем мярка за релевантността. За това има най-различни подходи. Експериментите потвърждават, че *само експериментално може да се установи кой подход при каква предварителна обработка дава най-добри резултати*,

поради което прототипите се оценяват от няколко различни тестващи потребители и се прави статистика над техните преценки дали системата е добра. Така системите могат да се настройват към различни предметни области с лексикони от важни значещи единици. Тук ще разгледаме три отделни, почти независими компоненти на IR системите: I, II и III.

I. Дизайн на модулите за обработка на естествен език

Повечето IR системи пазят своите наблюдения над документите във вид на инвертиран индекс (*inverted index*), матрица която показва колко пъти наблюдаваните до момента думи са се срещали в съдържащите ги документи (срещанията се наричат *postings*, а за броя срещания използваме честота - *frequency*). Инвертираният индекс е полезен при търсене, понеже позволява бързо да се види коя дума от въпроса в кои документи се съдържа. Той може да се комбинира с информация за позицията на думата в документа (цяло число като относително разположение спрямо началото). Така може лесно да се търси за *фрази*: например, във въпроса на потребителя идва фразата "застраховка на кола". След намиране на информацията за "застраховка" веднага се гледа къде и колко пъти се среща "кола/коли" и може бързо да се провери в кои документи двете думи се срещат заедно една след друга, почти чрез пресичане на съответните редове (или стълбове, в зависимост от организацията) на инвертирания индекс. По принцип IR-системите работят с примитивна дефиниция на понятието фраза и не разпознават напр. *car insurance rates*, ако се зададе напр. като *rates for car insurance* (компютърната лингвистика може да помогне за правилното разпознаване на фразите чрез частичен синтактичен анализ на главни групи в изречението, наричаме ги *chunks* /пънове/ - нещо стабилно, но изрязано – но това не се прави, защото след десетилетия натрупан опит се вижда, че ползата от лингвистичния анализ е минимална). Така че, по отношение на обработката на естествения език, първият основен въпрос във всяка IR-система е: *Кой буквен низ е дума и как да се разпознае тя в произволен текст, за да се построи най-правилният инвертиран индекс като модел на пространството от релевантни документи?* Отговорът на въпрос 1 съвсем не е тривиален. Вторият основен въпрос би трябвало да бъде: *Какво ще правим с колокациите (напр. сложни термини), с фразите и парафразите, с имената съставени от няколко низа - ще ги разпознаваме ли като една единица или не?* Отговорът на въпрос 2 се дава от разработчиците на системата, като частичен компромис между ефективност и възможост, и обикновено е НЕ. Нека се ограничим с въпрос 1, той е достатъчно сложен. Ще използваме за илюстрация примерни документи.

Таблица 1 с двата текста 1 и 2 ни припомня детайлно и многократно обсъжданите проблеми за разпознаване на думите в текста, като показва как дадени низове функционират с многозначна роля в парадигмите на няколко основни форми. Две думи са семантично многозначни - съществителното КОСА и глаголът БЕ'ЛЯ - и както знаем могат евентуално да бъдат различени чрез word-sense disambiguation, но това е невъзможно в практически задачи за значителен брой думи /поне засега/. Някои низове могат да бъдат различени след морфологичен анализ и POS-tagging при наличие на надеждни процедури за дадения език.

Текстове, постъпващи в системата	Изход след морфологичен анализ (налагане върху речник), с разпознаване на многозначност на словоформи	Изход след успешен POS-tagging (разпознаване на части на речта)	Изход след успешна процедура за Word sense disambiguation (както я прави човек)
Текст 1: Коси слънчеви лъчи огряха русите коси на Мария. Тя сплете разпиляната си коса. Няколко коса кацнаха наблизо, един запя. Песента на коса разсъни Мария и тя се замисли колко бели ще направи днес.	Коси слънчеви лъчи огряха русите коси на Мария. Тя сплете разпиляната си коса. Няколко коса и бели гълъби кацнаха навън, един запя. Песента на коса разсъни Мария и тя се замисли колко бели ще направи днес.	Коси (*прил. кос*) слънчеви лъчи огряха русите коси (*същ.*) на Мария. Тя сплете разпиляната си коса (*същ.*). Няколко коса (*същ. кос*) и бели (*прил. бял*) гълъби кацнаха навън, един запя. Песента на коса (*същ.*) разсъни Мария и тя се замисли колко бели (*същ. беля*) ще направи днес.	Коси (*прил. кос*) слънчеви лъчи огряха русите коси (*същ. коса1*) на Мария. Тя сплете разпиляната си коса (*същ. коса1*). Няколко коса (*същ. кос*) и бели (*прил. бял*) гълъби кацнаха навън, един запя. Песента на коса (*същ. кос*) разсъни Мария и тя се замисли колко бели (*същ. беля*) ще направи днес.
Текст 2: Бащата на Мария взе една коса и тръгна да коси. Брат й цял ден ще бели царевица, а майка й ще бели платно на реката.	Бащата на Мария взе една коса и тръгна да коси. Брат й днес ще бели царевица, а майка й ще бели платно на реката.	Бащата на Мария взе една коса (*същ.*) и тръгна да коси (*глагол кося*). Брат й днес ще бели (*глагол беля*) царевица, а майка й ще бели (*глагол беля*) платно на реката.	Бащата на Мария взе една коса (*същ. коса2*) и тръгна да коси (*глагол кося*). Брат й днес ще бели (*глагол беля1*) царевица, а майка й ще бели (*глагол беля2*) платно на реката.

Таблица 1. Примерни текстове (стълб 1), подложени на три последователни процедури за все по-презицен анализ на словоформите на думите КОСА и БЕЛЯ: в стълб 2 намираме словоформите в речника, но не знаем как да разрешим многозначността им – видяхме това в домашно 1; в стълб 3 разпознаваме частите на речта; в стълб 4 разпознаваме смисъла на думите.

Ако обаче в нашата система няма голям морфологичен речник (не е задължително да имаме, може да сме още в началото на нашите дейности по обработка на текст и да не сме съставили компютърен речник), то тогава първо ни хрумва да дефинираме думите като низове-"гнезда" по следния елементарен начин, който се нарича "свеждане към основа" или *stemming* [2]:

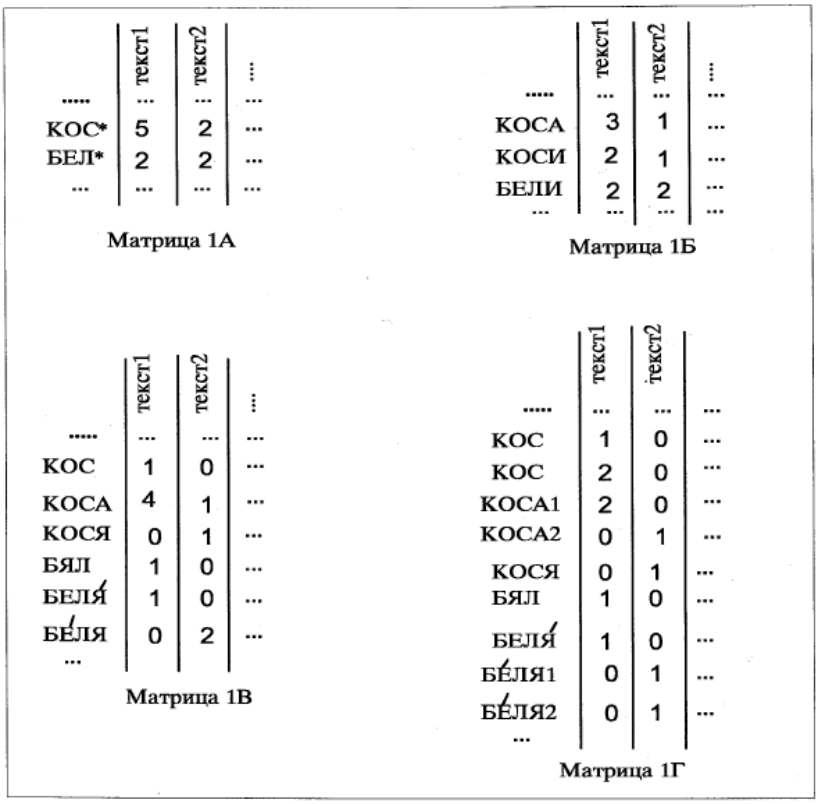
КОС* за *коса, косата, коси, косите*, и

БЕЛ* за *беля, белята, бели, белите*.

Така съставяме нашата първа версия на елементарен речник на значещите думи в текста. Алгоритъмът за разпознаване на единиците от този речник в произволен текст е прост: започваме разпознаването от по-дългите основи, например при появата на ... *белина* ... в текста ще разпознаем БЕЛИН*, а не БЕЛ*. Търсенето по ключови думи често се основава върху подобни принципи, така че простотата не е повод да се откажем от ефективен алгоритъм (ако той работи приемливо и върши работа).

2 Има свободни stemmer-и в Интернет, особено за английския (намират се с google).

Да разгледаме инвертирания индекс (Фиг. 1), който би се получил при различните процедури за предварителна обработка на текстове 1 и 2 от Таблица 1. Матрица 1А е построена по прекалено примитивен модел за преброяване на честотата, при който всички срещания на КОС* и БЕЛ* в текстове 1 и 2 са преброени като срещания на една единица - обикновено така действа stemming-ът. Матрица 1Г е непостижимият засега идеал, според който интуитивно задаваме въпросите си в Интернет (тази матрица съществува в главите ни, когато питаме, но не на практика). Читателят може да оцени сам каква е разликата между двете матрици, ако ги използваме като отправна точка за мерене на подобие между документи. От гледна точка на компютърната лингвистика, желателно е IR-системата да разпознае 100% от важните единици в текста като части на речта и след това като значения. Предполага се, че така ще се постигне максимално добрият резултат за търсене на подобие на документи (но никой не е проверявал тази хипотеза на практика). Реалните системи са далече от този идеал.



Фигура 1. Различни варианти на инвертиран текст в зависимост от разпознаване на единиците в текстове 1 и 2 на Таблица 1. Оформят се различни по размер пространства за търсене на близост. Различните размерности на индекса са в известен смисъл пречка за сравнителния анализ на IR-системите, понеже те работят над различен вход.

Матрица 1Б показва как се преброяват честотите на низовете, ако ги разгледаме като отделни последователности от букви. Повечето системи работят с алгоритми между 1А и 1Б (но те са и многоезични; разпознават се отделните низове и най-много по някакъв алгоритъм се свеждат грубо до основни форми, което за английския не е особено трудно при сравнително простата морфология). Редица системи извършват stemming за английския език с разпознаване на деривации като една дума, например invest - investor - investment - investing -

investition. Матрица 1B се получава след разпознаване на частите на речта. Ясно е, че при текстове на силно флективни езици - каквито са славянските - търсенето на документи би следвало да се основава върху индекси от типа 1B. Причината е, че във флективните езици всяка дума има много словоформи и те съвпадат с форми на други думи - както е показано в таблица 1. За сравнение, английското съществително има 2 форми, глагола - 3, прилагателното - 1 (т.е. всяка дума се проявява в текста като малък брой различни низове); докато на български съответните бройки са 4 за съществителните женски и среден род и 7 за съществителни мъжки род, до 52 за глагола и 8 за прилагателното. Изобилието на съвпадащи форми затруднява разграничаването на единиците в текста и води до получаване на неадекватни индекси, поради което можем да очакваме по-лош резултат при търсене.

Друга важна предварителна обработка е разпознаване на имената (named entities recognition), които носят важна информация за съдържанието на документа. На имената се присвоява по-голяма тежест при категоризиране и автоматично резюмиране на документите. По-тежки са и думите от заглавието, от първото изречение на началния и заключителен параграфи и т.н. Обикновено не се разпознават отделни основи в сложни думи (compounds), съставени от различни компоненти на значещи думи, които се изписват слято.

Премахването на стоп-думите от текста е почти задължителен момент[3]. Обикновено като стоп-думи се изброяват всички служебни, свързащи единици: предлози, съюзи, местоимения, членове, числителни, междуметия. Такива думи се срещат твърде често във всеки текст и биха уголемили ненужно индекса, без да носят съществена информация. Няма особено значение в кой момент ще изтрием стоп-думите, стига да не попречим на евентуалния опит за правилното разпознаване на фрази от рода на *цена на тока* (която съдържа предлога *на*) и на имената. Очевидно извличането на документи се извършва чрез доста грубо моделиране на смисъла и тази техника не подхожда за задачи, предполагащи по-фино разбиране. Предлага се усъвършенстване чрез въвеждането на т.нар. "полу-стоп думи" (които са стоп-думи, третирани като значещи единици при специфични условия) и т.н.

В заключение ще кажем, че винаги можем *някак* да пресметнем инвертирани индекси за набор от документи, но трябва да си даваме сметка честотите на какви единици се отразени в тях. Очевидно NLP-проблемите нямат универсално добро решение. Често учебниците по компютърна лингвистика не включват детайлни описания на подходите за Information Retrieval, понеже те не се базират върху обработка на лингвистичните единици в текста. Трябва също така да различаваме термините Information Retrieval и Information Extraction, понеже вторият се използва при автоматична обработка за частичен анализ и частично разбиране на текста.

---

3 Стоп-думи (stop words) са тези думи, които биха довели до спиране (задръстване) на търсенето в Интернет, ако ги зададем като ключови. Терминът възникна в зората на Интернет. Такава дума е например the, тя се среща във всеки документ на английски език и затова търсене с ключова дума "the" води до извличане на всички интернет страници на английски език. Произволна търсачка ще ви даде списък на различни www-адреси със стоп-думи, ако попитате за "stop words".

II. Оценяване работата на IR-системата: точност на извличане

Качеството на търсачките зависи и от това как мерим релевантността на отговора. Повечето на наличните метрики комбинират понятията за **precision** и **recall** (*точност* и *възвръщаемост*, познаваме ги от Information Extraction). Precision е процентът на релевантните документи в множеството на отговора от изведени документи - напр. ако от 100 изведени документа 60 са релевантни, точността е 60%. Recall е процентът на всички документи от релевантната колекция, които са извлечени в отговора. Напр., в цялата колекция има 360 релевантни документа изобщо, и ако само 60 от тях са включени в отговора, системата има recall 13.3% (1/6).

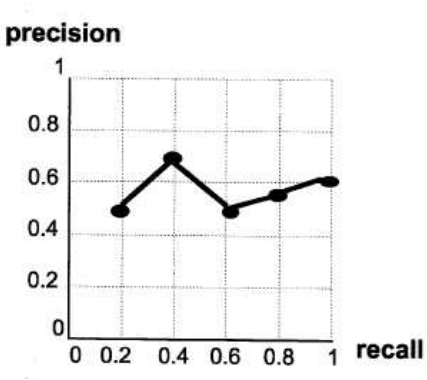
Оценки	Наредба 1	Наредба 2	Наредба 3
	d1: рел.	d10: нерел.	d6: нерел.
	d2: рел.	d9: нерел.	d1: рел.
	d3: рел.	d8: нерел.	d2: рел.
	d4: рел.	d7: нерел.	d10: нерел.
	d5: рел.	d6: нерел.	d9: нерел.
	d6: нерел.	d5: рел.	d3: рел.
	d7: нерел.	d4: рел.	d5: рел.
	d8: нерел.	d3: рел.	d4: рел.
	d9: нерел.	d2: рел.	d7: нерел.
	d10: нерел.	d1: рел.	d8: нерел.
Точност за 5 документа, т.е. колко от първите 5 документа са релевантни	1.0	0.0	0.4
Точност за 10 документа	0.5	0.5	0.5
Неинтерполирана средна точност	1.0	0.3544	0.5726
Интерполирана средна точност (в 11 точки)	1.0	0.5	0.6440

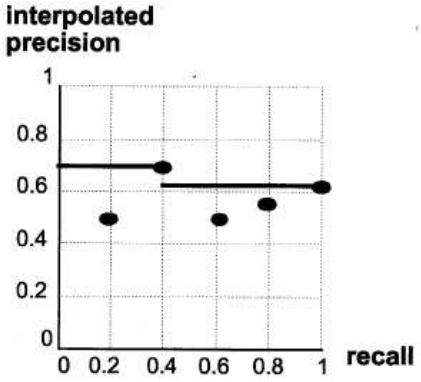
Таблица 2. Примери за наредба (ranking) и мерки за точност

Да разгледаме Таблица 2. Стълбове 2-4 показват 3 наредби на документите d1-d10 (d1-d5 са релевантни, а d6-d10 - не), които могат да се изведат като отговор на въпрос в Интернет (забележете, че именно търсачките решават как да разположат извлечените документи на екрана в линейна последователност; ако нямаме мярка за релевантност възниква проблем как да се построи наредбата). Наредбите са оценени съгласно четири мерки за точност: точност в първите 5 показани документа, в първите 10 показани документа, неинтерполирана средна точност и интерполирана в 11 точки (вж. по-долу). Разпространена мярка е точността в определена извадка (*precision at a particluar cutoff*), например точност в 5/10/20 или 100 документа. Това е добра идея, понеже измерването на точността в няколко начални сегмента на извлечената колекция документи дава добро чувство за метода и неговата способност да извежда релевантни и нерелевантни документи. Тази таблица показва защо подредбата е важна - и трите отговора съдържат едно и също количество релевантни документи и са еднакво добри, ако е зададен **праг за точност** например 50% - но очевидно наребда 1 е по-добра от наредба 2, понеже четем отгоре надолу и не искаме да виждаме първо нерелевантните документи. Последните два реда показват "по-контекстни" мерки за точност.

Неинтерполираната средна точност агрегира много точности в една мярка. Точността се изчислява във всяка точка на списъка, в която е намерен релевантен документ, и след това тези точности се осредняват. Например за наредба 1 релевантни документи има в позиции 1, 2, 3, 4 и 5. Точността е 100% за всичките 5 позиции: 1 от 1 за d1, 2 от 2 за d2, 3 от 3 за d3, 4 от 4 за d4 и 5 от 5 за d5. Средната стойност е 1 (в нерелевантните точки, точността е 0 и те не се сумират). За наредба 3 имаме следните точности: 1/2 за d1, 2/3 за d2, 3/6 за d3, 4/7 за d5 и 5/8 за d4, средно 0.5726. Наредба 3 е по-добра от наредба 2 по тази мярка, *понеже релевантните документи се появяват относително по-напред*. Неинтерполираната средна точност показва колко 'напред' се появяват релевантните документи в отговора.

Интерполираната средна точност е свързана с мярката recall. Точността се изчислява за различни нива на recall, например на нива 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% и 100% в случай на тези "средни" 11 точки (това е най-често използваната оценка, чертаят се схеми в координатна система като показаната на Фиг. 2.1 и Фиг. 2.2). За recall на ниво A, точността B се изчислява в точката на наредения списък, където пропорцията на изведените документи достига A. Да разгледаме пак наредба 3 на Таблица 2 и Фигура 2.1 по-долу. Предполагаме, че релевантните 5 документа са единствените релевантни в колекцията. За recall 20% - *изваждане на 1 от 5-те релевантни документа* - според наредба 3 имаме, че това се случва в документ d1, втори в наредбата, и там точността е 0.5. Така е начертана точката (20%, 0.5) на Фиг. 2.1. След това виждаме, че recall 40% - *изваждане на 2 от 5-те релевантни документа* - се достига при извеждането на документ d2, трети в наредбата, и там точността е 2/3 или 0.66. Така е начертана точката (40%, 0.66) на Фиг. 2.1. И така нататък, за всички 5 точки на Фигура 2.1., които по абцисата се "случват" съответно за документите d1, d2, d3, d5 и d4 на наредба 3 и са съединени с отсечки. Така получаваме една крива - какви са тенденциите да се мени точността според количеството извадени релевантни документи. Фигура 2.2 е получена от Фиг. 2.1. както следва: ако точността нарасне докато се движим надолу по списъка с изведените документи - както става от 20% до 40% recall и после от 0,6% до 100% recall, тогава интерполираме и вземаме най-високата точност извън мястото, където за първи път сме достигнали recall A. Така интерполираната точност за recall 0-40% е 0.66, а от 40 до 100% е 5/8 (0.625), стойността за точка d4, с която сме интерполирали точността между recall 0.4-1. Тоест, заместваме с локални максимуми "от ляво на дясно". Така получаваме картинката на Фигура 2.2. После намираме средното с отчитане на тежестта, тоест 0.66 е "валидно" за 40% от поведението на системата и 5/8 за останалите 60%. Получаваме с точност до закръгляване 0.64, което е стойността на интерполираната средна точност за наредба 3 на таблица 2. Графики като получените на Фиг. 2.1 и 2.2 се наричат "криви на връзката между точност и възвръщаемост" и са по-глобални характеристики на системата. Забележете, че точност и възвръщаемост са две различни мерки: може да се извлече цялата колекция от документи с възвръщаемост 100%, но с много ниска точност; както и да се извлекат само високо-релевантни документи с голяма точност, но с ниско ниво на възвръщаемост.





Извеждат се:
0, 20, 40, 60, 80, 100%
от наличните релевантни документи
с неинтерполираната средна
точност в съответните точки

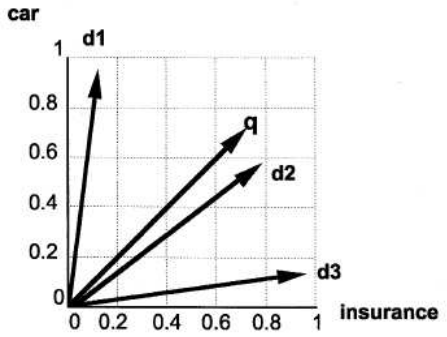
Фигура 2.1. **Фигура 2.2.**

В Интернет приемаме за *оптимална* подредбата в намаляващ ред на релевантност на документите. Предполага се (което не е вярно за всички колекции от документи изобщо), че *(i)* документите са независими, *(ii)* поредица от въпроси се третира като редица от изолирани заявки и *(iii)* присвоената оценка за релевантност на документа е само приближение, тъй като има дисперсия и други фактори на "разсейване" и шум.

III. Оценка за релевантност на документите: модел на векторните пространства (Vector Space Model)

Това е един от най-разпространените модели за динамично извличане на документи (по неизвестни отпреди въпроси), главно поради концептуалната му простота и привлекателността на метафората да се използва пространствена близост като апроксимация на семантичната близост. Документите и заявките се представят като вектори в пространство с много голям размер, като всяко измерение съответства на една дума от колекцията документи (стоп-думите се трият). Казано по-точно, инвертираният индекст задава единиците, съответстващи на оси в пространството – ние ги разглеждаме като модел на смисъла на текста. Най-релевантните документи за даден въпрос са тези вектори, които са най-близо до вектора на въпроса, тоест документите съдържащи същите думи. Близостта се изчислява като се гледат ъглите и се избират документите, които сключват най-малък ъгъл с вектора на въпроса (виж формулата за косинус-близост по-долу).

Пример е даден на Фиг. 3. Векторното пространство има два размера, съответстващи на думите (термините) car и insurance. Вече не питаме как са извлечени тези думи - направено е *както можем*, това се обсъжда в раздел I на този текст. Въпросът е векторът q с координати



Фигура 3. Колекция от три документа и въпрос във векторно пространство с размер 2, съответстващо на двата термина **car** и **insurance**

(0.71,0.71) и той съответства на текста '***car insurance***', а трите документа измежду които търсим са d_1 с координати (0.13,0.99), d_2 с координати (0.8,0.6) и d_3 с координати (0.99,0.13). Координатите съответстват на **тежестта** на термините car и insurance в трите документа и въпроса, по-долу ще обясним как строим такива вектори по термините в текста (в IR под **term** се разбират и фрази и имена, изобщи всичко което системата разглежда и смогва да отдели като отделна и значима текстова единица). По принцип всички документи са релевантни понеже изобщо съдържат car и insurance - очевидно и в трите документа се използват тези думи, тоест решаваме, че в тях се говори за car и insurance каквото и да означават тези низове. На фиг. 3, документът d_2 сключва най-малък ъгъл с q, значи той е най-релевантен и като най-близък ще бъде изведен като първи (или единствен). Това е правилно, защото в него думите car и insturance са еднакво *salient* (забележителни, изпъкващи) - което е вярно и за въпроса, тези думи там са еднакво важни независимо от подредбата - но очевидно не е вярно за d_1 и d_3, в които поне един от двата термина не е от голямо значение. По-долу първо ще се спрем на формулата за пресмятане на близост между векторите и после ще минем към построяването им.

Подобие /близост/ на вектори (vector similarity) в *n*-пространство

По формулата за косинус между два вектора, разстоянието (наричано още нормализиран коефициент на корелация) между въпрос и документ е:

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

където q и d са векторите на въпроса и документа. Смята се доколко срещането на термина ***i*** (изразено чрез q_i и d_i) корелира във въпроса и в документа и след това сумата се дели на

Евклидовата дължина на двата вектора, за да се "оразмери" според дължината на векторите и "тежестите" на другите думи, които се срещат и които са изразени чрез отделните стойности на q_i и d_i. Забележете, че векторите на документите на Фиг. 3 са нормализирани, т.е.

$$\sqrt{\sum_{i=1}^{n} d_i^{\,2}} = 1$$

Нормализацията е обичайна трансформация и нещо добро. Например, без нея дълги вектори (за дълги документи с много повтарящи се думи) ще имат непропорционално "тежки" стойности спрямо векторите на късите документи, така че в IR има нужда да се разглеждат и такива "по-глобални" параметри на отделните документите и колекциите документи.

Присвояване на тежест на термините (term weighting)

Споменаваме (начални) идеи как да се мери тежестта на думите в модела на векторното пространство. Може просто да преброите думите в документа, но това е много неточно определяне на "срещането" спрямо поведението на думата в колекцията документи. Затова в IR се използват понятия като term frequency, document frequency и collection frequency [2]:

- **term frequency** - $tf_{i,j}$ - брой срещания на дума w_i в документа q_j
- **document frequency** - df_i - брой документи от колекцията, в които се среща думата w_i
- **collection frequency** - cf_i - общ брой срещания на w_i в колекцията

Последните две характеристики могат да се ползват само в случаи на налична колекция, което не винаги е вярно (при търсене в Интернет, ако on-line архиви се разглеждат като временни колекции).

Чрез **честотата на термина** изразяваме колко значим е даден термин за документа. Ако една дума се повтаря много често в документа, тя е добър кандидат да опише съдържанието му (от друга страна, забележете, че ако една дума се повтаря равномерно често в дадена колекция, тя е лош кандидат да служи за характеризиращ разграничител на документите един спрямо друг). Честотата на термина често се скалира чрез функции като

$f(\text{tf}) = SQRT(\text{tf})$ или $f(\text{tf}) = 1 + log(\text{tf}),\ \text{tf} > 0$

понеже повече срещания на една дума предполагат нейната по-голяма важност, но не чак толкова колкото самият брой срещания. Например, счита се, че ако една дума се среща 3 пъти в документ, стойностите $sqrt(3)$ или $1+log(3)$ характеризират по-добре нейната важност, отколкото числото 3. Експериментално е прието да не се използва самата бройка на срещанията, а нейно изображение към по-"сгъстена" скала. Така че документ с 3 срещания на една дума е по-важен от този с едно срещане, но не чак 3 пъти по-важен.

Честотата в документите е индикатор за информативността на думата в колекцията. Дума със семантично ударение ще се случва често в документи от дадена тема, ако се случва изобщо. Дума без семантично ударение ще бъде със случайно или равномерно споменаване

във всички документи. Погледнете долната таблица за думите insurance и try в корпус от статии на New York Times. Двете думи имат един и същи брой срещания в цялата колекция, но insurance се среща в два пъти по-малко документи:

дума	честота в колекцията	честота в документите
insurance	10440	3997
try	10433	8760

Очевидно, почти във всяка област - всеки документ - може да се говори за: *try* (с около едно споменаване в документ), докато *insurance* се споменава повече от веднъж (средно по 3 пъти) в документите, които дискутират такива теми.

Честотата в документите също се скалира логаритмично. Формулата

$$log(N/df_i) = log\ N - log\ df_i$$

където *N* е броят на всички документи в колекцията, дава най-голямо тегло на думите, които се срещат в един документ (*log* N - *log* 1 = *log* N) и тегло 0 на думите, присъстващи във всички документи (*log* N - *log* N = 0).

Честотата на термина в документа $tf_{i,j}$ и честотата в документите df_i могат да се комбинират в "тегло" на *i*-тата дума в *j*-тия документ както следва:

$$\textbf{тегло}\boldsymbol{(i,j)} = (1+log(tf_{i,j})) * log\ (N/df_i), \quad \text{ако } tf_{i,j} >= 1$$
$$= 0, \quad \text{ако } tf_{i,j} = 0,$$

където *N* е броят на всички документи в колекцията.

Формулата *log* (N/df_i) често се нарича "инвертирана честота в документите" (inverse document frequency) или ***idf*-тегло**. Теглото, зададено в горната дефиниция, е пример за голямата фамилия на т.нар. *tf.idf*-схеми. Те задават тегло на думата според нейното срещане, като по някакъв начин включват скалираната честота в документите и използват нормализация. Понякога ги критикуват като ad-hoc формули, защото не използват математически модел на дистрибуцията на термините в документите; на практика обаче тези схеми са ефективни и често използвани в ситуации, където са достатъчни по-груби мерки на близост и подобие.

IV. Заключение

Видяхме на идейно ниво класическия метод за мерене на близост, който продължава да се използва много активно и до днес. Изглежда прецизната предварителна лингвистична обработка на морфо-синтактично ниво не води до значително подобрение, освен ако не се направи Word Sense Disambiguation, което засега не се случва в реално време и за „големи" данни. Забележете, че описаните в III подходи са чист data mining след свеждане на текста до единици за броене, понеже след получаване на основната матрица (инвертиран индекс) *думи* x *документи* е все едно едно в какви данни се търси — текстови или други. Изискват се големи вложения на труд и време за радикални подобрения на установените в момента техники.

Литература:

1. Manning, C, and H. Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.
2. C. J. van Rijsbergen. *Information Retrieval*. Класически учебник, публично достъпен на http://www.dcs.gla.ac.uk/Keith/Preface.html