Векторное представление слов. Профильные методы классификации

Курс «Интеллектуальные информационные системы»

Кафедра управления и информатики НИУ «МЭИ»

Осень 2018 г.

Word2Vec

Word2vec — программный инструмент анализа семантики естественных языков, представляющий собой технологию, которая основана на дистрибутивной семантике и векторном представлении слов. Этот инструмент был разработан группой исследователей Google в 2013 году. Работу над проектом возглавил Томаш Миколов

- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space // In Proceedings of Workshop at ICLR, 2013

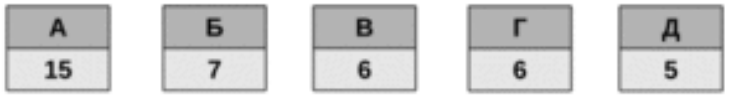
Работа этой технологии осуществляется следующим образом: word2vec принимает большой текстовый корпус в качестве входных данных и сопоставляет каждому слову вектор, выдавая координаты слов на выходе. Сначала он создает словарь, «обучаясь» на входных текстовых данных, а затем вычисляет векторное представление слов. Векторное представление основывается на контекстной близости: слова, встречающиеся в тексте рядом с другими словами (а следовательно, имеющие схожий смысл), в векторном представлении будут иметь близкие координаты векторов-слов.
Более формально задача стоит так: максимизация косинусной близости между векторами слов (скалярное произведение векторов), которые появляются рядом друг с другом, и минимизация косинусной близости между векторами слов, которые не появляются друг рядом с другом. Рядом друг с другом в данном случае значит в близких контекстах.

- Контекст - в широком смысле – среда, в которой существует объект.

Word2Vec – как работает?

- Читается корпус, и рассчитывается встречаемость каждого слова в корпусе (т.е. количество раз, когда слово встретилось в корпусе — и так для каждого слова)
- Массив слов сортируется по частоте (слова сохраняются в хэш-таблице), и удаляются редкие слова
- Строится дерево Хаффмана. Дерево Хаффмана (Huffman Binary Tree) часто применяется для кодирования словаря — это значительно снижает вычислительную и временную сложность алгоритма.
- Из корпуса читается т.н. субпредложение (sub-sentence) и проводится субсэмплирование наиболее частотных слов (sub-sampling). Субпредложение — это некий базовый элемент корпуса, обычно — просто предложение, но это может быть и абзац, например, или даже целая статья. Субсэмплирование — это процесс изъятия наиболее частотных слов из анализа, что ускоряет процесс обучения алгоритма и способствует значительному увеличению качества получающейся модели.

Дерево Хаффмана



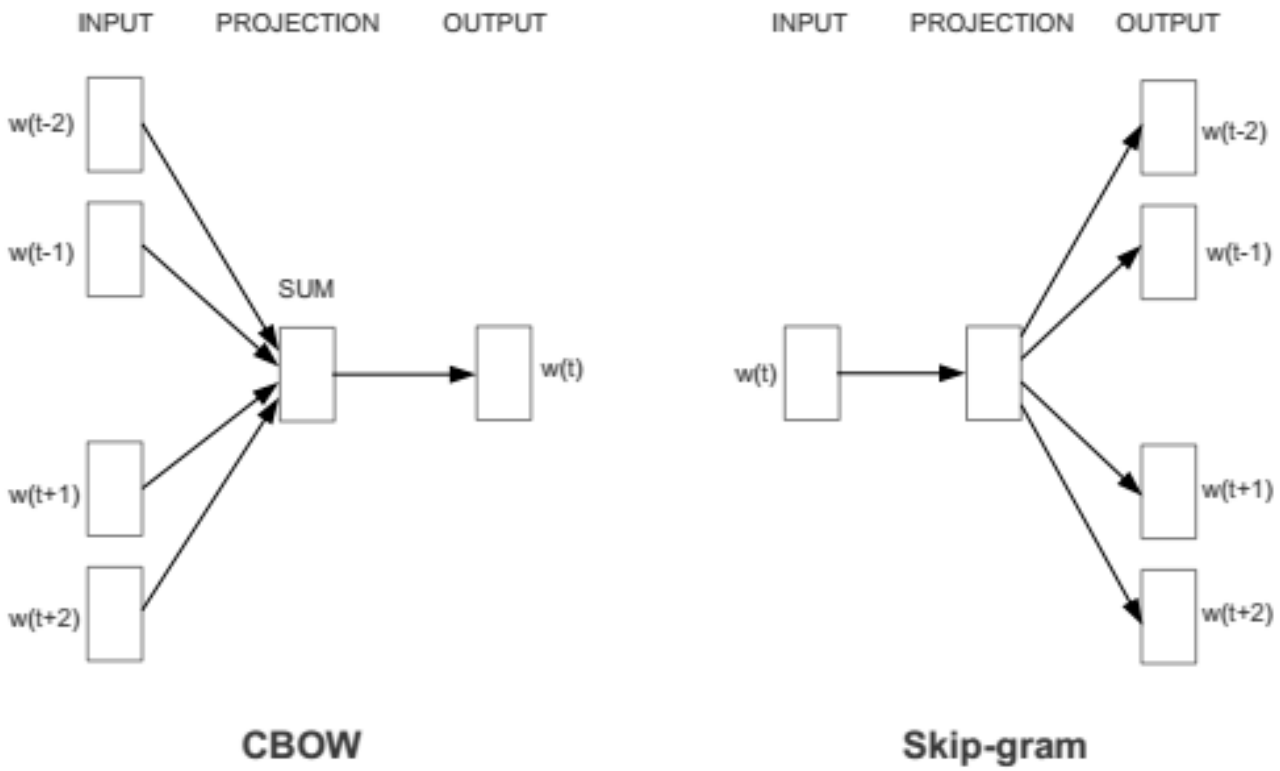
Word2Vec – как работает?

- По субпредложению проходим окном (размер окна задается алгоритму в качестве параметра). В данном случае под окном подразумевается максимальная дистанция между текущим и предсказываемым словом в предложении. То есть, если окно равно трем, то для предложения «The quick brown fox jumps over the lazy dog» анализ будет проходит внутри блока в три слова — для «The quick brown», «quick brown fox», «brown fox jumps» и т.д. Окно по умолчанию равно пяти, рекомендуемым значением является десять.

- Применяется нейросеть прямого распространения (Feedforward Neural Network) для получения векторных представлений слов.

Word2Vec (2)

Существует две архитектуры нейросети - CBOW (Continuous Bag Of Words) и Skip-gram, которые описывают, как именно нейросеть «учится» на данных и «запоминает» представления слов. Принципы у обоих архитектур разные. Принцип работы CBOW — предсказывание слова при данном контексте, а Skip-gram наоборот — предсказывается контекст при данном слове.



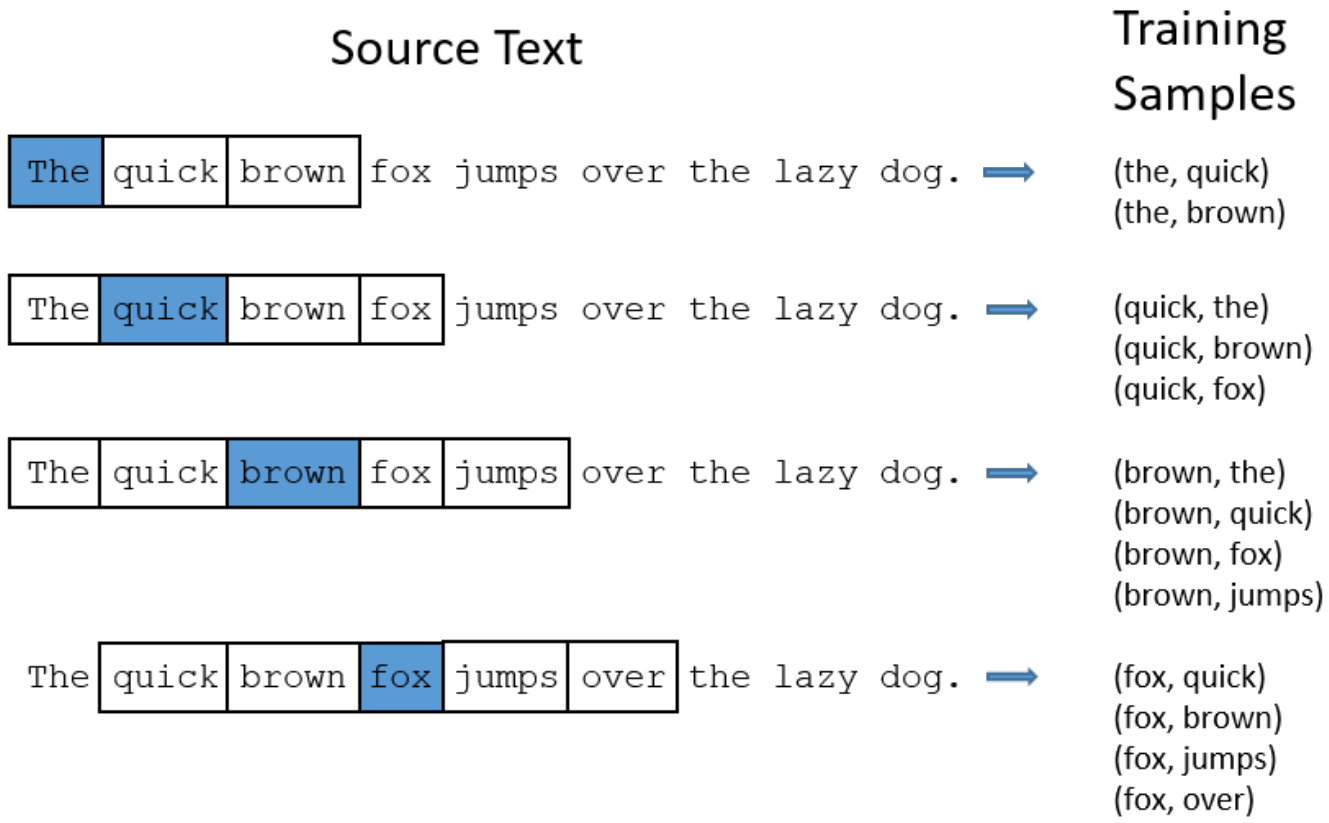
Skip-gram

Мы обучим нейронную сеть следующим действиям. Возьмем определенное слово в середине предложения (вводное слово, input word). Настроенная сеть должна выдать для каждого слова в нашем словаре вероятности того, что оно будет находиться «рядом» с вводным.

Полученная вероятность показывает то, насколько вероятно каждое слово из словаря будет находится в пределах размера окна с вводным словом.

Например, если вы задали обученной сети входное слово “Советский", то для таких слов, как “Союз” и “Россия”, полученная вероятность будет намного выше, чем для несвязанных слов, например, “арбуз” и “кенгуру”.

Обучать сеть мы будем, подавая пары слов из обучающей выборки.
В приведенном примере показаны некоторые примеры таких обучающих пар. Слово, выделенной синим является вводных словом, размер окна = 2



Negative sampling

Задача построения модели word2vec выглядит так: максимизация близости векторов слов (скалярное произведение векторов), которые появляются рядом друг с другом, и минимизация близости векторов слов, которые не появляются друг рядом с другом.

Упрощенное уравнение этой идеи:

$$\frac{(w_v \times w_c)}{\sum(w_v \times w_{c1})}$$

В числителе мы имеем близость слов контекста (w_c) и целевого слова (w_v), в знаменателе — близость всех других контекстов (w_{c1}) и целевого слова (w_v). Проблема в том, что считать все это долго и сложно — контекстов может быть огромное множество для каждого слова, а кроме того, многие слова могут вообще не встречаться вместе, поэтому большая часть вычислений является избыточной. Негативное сэмплирование — один из способов справиться с этой проблемой. Принцип — мы не считаем ВСЕ возможные контексты, а выбираем случайным образом несколько негативных контекстов (w_{c1}). Под «негативным» имеется ввиду контекст, вероятность появления которого рядом с вводным словом близка к 0.

Если слово кошка появляется в контексте еды, то вектор слова еда будет ближе к вектору слова кошка, чем вектора некоторых иных случайно выбранных слов (например автомобиль, Windows, кондиционер и т.д.), таким образом, необязательно привлекать вообще все слова из обучающего корпуса.

Примеры использования библиотеки gensim

```
# build vocabulary and train model
    model = gensim.models.Word2Vec(
        documents,
        size=150,
        window=10,
        min_count=2,
        workers=10)
    model.train(documents, total_examples=len(documents), epochs=10)
```

```
# similarity between two different words
model.wv.similarity(w1="dirty",w2="smelly")
```

```
0.76181122646029453
```

```
# similarity between two identical words
model.wv.similarity(w1="dirty",w2="dirty")
```

```
1.0000000000000002
```

```
# similarity between two unrelated words
model.wv.similarity(w1="dirty",w2="clean")
```

```
0.25355593501920781
```

```
# look up top 6 words similar to 'france'
w1 = ["france"]
model.wv.most_similar (positive=w1,topn=6)
```

```
[('canada', 0.6603403091430664),
 ('germany', 0.6510637998580933),
 ('spain', 0.6431018114089966),
 ('barcelona', 0.61174076795578),
 ('mexico', 0.6070996522903442),
 ('rome', 0.6065913438796997)]
```

```
w1 = "dirty"
model.wv.most_similar (positive=w1)
```

```
[('filthy', 0.871721625328064),
 ('stained', 0.7922376990318298),
 ('unclean', 0.7915753126144409),
 ('dusty', 0.7772612571716309),
 ('smelly', 0.7618112564086914),
 ('grubby', 0.7483716011047363),
 ('dingy', 0.7330487966537476),
 ('gross', 0.7239381074905396),
 ('grimy', 0.7228356599807739),
 ('disgusting', 0.7213647365570068)]
```

Профильные методы классификации

Профиль класса – это формальный объект, способный охарактеризовать все остальные элементы класса.

Например – центроид

Профили классов могут быть разделены на следующие категории:

1) Логический профиль, который состоит из признаков, представленных в классе: (вес признака в таком профиле равен *“0”* или *“1”*).

2) *Рассчитываемый профиль*, в этом случае вес признаков рассчитывается на основе какого-либо правила. Примером может служить центроид класса

3) *Экспертный профиль*, задаваемый пользователем на основе собственных знаний и опыта.

Что такое профиль класса?

Взвешивание и отбор признаков может проводиться на основе известных процедур выявления информативных терминов:

- Частотный,
- Статистический,
- Теоретико-информационный,
- Эвристический,
- …

Q_k / X	Принадлежность классу Q_k	Непринадлежность классу Q_k	$\sum$
Наличие признака $x^{(i)}$	A	B	$A+B$
Отсутствие признака $x^{(i)}$	C	D	$C+D$
$\sum$	$A+C$	$B+D$	N

Статистический подход выявления информативных терминов

Хи-квадрат - профиль:

$$\chi^2(x^{(i)}, Q_g) = N \cdot \frac{(AD - BC)^2}{(A+B)(C+D)(A+C)(B+D)}$$

РО-профиль:

$$\rho(x^{(i)}, Q_k) = \frac{(AD - CB)}{\sqrt{(A+B)(C+D)(A+C)(B+D)}}$$

Профиль Юла:

$$Q(x^{(i)}, Q_g) = \frac{AD - BC}{AD + BC}$$

Теоретико-информационный подход выявления информативных терминов

МИ-профиль (Mutual information):

$$MI(x^{(i)}, Q_g) = \log_2 \frac{A \cdot N}{(A+B) \cdot (A+C)}$$

Нормированный МИ-профиль:

$$NMI(x^{(i)}, Q_k) = \frac{A \log_2 \frac{AN}{(A+B)(A+C)}}{(A+B) \log_2 \frac{N}{A+B}}$$

Эвристический подход выявления информативных терминов

Простой коэффициент совстречаемости

$$S = \frac{A + D}{A + B + C + D}$$

Первый коэффициент несогласия

$$SN1 = \frac{C + B}{A + B + C + D}$$

Коэффициент Рассела-Рао

$$RR = \frac{A}{A + B + C + D}$$

Коэффициент Роджерса-Танимото

$$RT = \frac{A + D}{A + D + 2(B + C)}$$

Первый коэффициент Сокала-Сниса

$$SS2 = \frac{2(A + D)}{2(A + D) + B + C}$$

Коэффициент Хаммана

$$H = \frac{(A + D) - (B + C)}{A + B + C + D}$$

Эвристический подход выявления информативных терминов (2)

Коэффициент Джаккарда (Жаккара)

$$J = \frac{A}{A + B + C}$$

Второй коэффициент несогласия

$$SN2 = \frac{C + B}{A + B + C}$$

Коэффициент Dice

$$Dice = \frac{2A}{2A + B + C}$$

Второй коэффициент Сокала-Сниса

$$SS2 = \frac{A}{A + 2(B + C)}$$

Первый коэффициент Кульчинского

$$K1 = \frac{A}{B + C}$$

Профильные методы

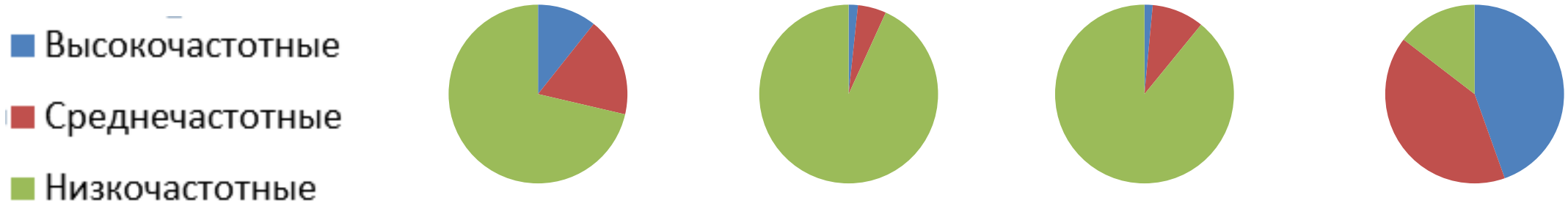
$$W_k = \sum_{i=1}^{Mk} tf_i \cdot \Pr of(x^{(i)}, Q_k) \qquad W_k = \max \text{ (для } \forall k, k = 1,\ldots, K)$$

Профили класса «Базы данных»

	РО-профиль		НМИ-профиль		J-профиль		С-С-профиль	
	databas	0,750	субд	0,689	databas	0,644	запрос	0,952
	субд	0,615	databas	0,652	баз	0,459	databas	0,949
	запрос	0,588	sql	0,623	запрос	0,420	субд	0,949
	баз	0,579	dbm	0,613	субд	0,415	реляцион	0,944
	sql	0,525	запрос	0,608	queri	0,352	queri	0,942
	реляцион	0,522	реляцион	0,599	реляцион	0,318	sql	0,939
	queri	0,512	sql-запрос	0,523	sql	0,308	dbm	0,937
	dbm	0,511	queri	0,520	модел	0,295	sql-запрос	0,933
	structur	0,409	ms	0,510	dbm	0,292	ms	0,932
	relat	0,385	mysql	0,466	relat	0,269	mysql	0,929
	sql-запрос	0,385	таблиц	0,466	структур	0,255	oracl	0,929
	ms	0,367	oracl	0,450	structur	0,234	postgresql	0,928
	структур	0,314	postgresql	0,450	data	0,184	relat	0,928
	mysql	0,306	баз	0,443	sql-запрос	0,169	sql-queri	0,928
	таблиц	0,306	sql-queri	0,431	model	0,161	таблиц	0,928
	oracl	0,283	inject	0,411	manag	0,160	end-us	0,927
	postgresql	0,283	pargresql	0,411	ms	0,154	laplas-stilt	0,927
	tabl	0,279	клиентск	0,411	выполнен	0,153	lst	0,927
	выполнен	0,271	tabl	0,392	access	0,153	secur	0,927
	access	0,271	db	0,388	base	0,153	бд	0,927

Сравнение профильных методов классификации

	PO	НМИ	C-C	J
Какие термины выделяет	Высокочастотные, среднечастотные	Среднечастотные, низкочастотные	Среднечастотные	Высокочастотные
Интервал значений весов	[-1;1]	[0;1]	[0;1]	[0;1]
Коэффициент убывания весов терминов λ	Средний 4.8	Низкий 2.2	Очень низкий 1.05	Высокий 7.4
Используемые значения из таблицы сопряженности	A, B, C, D	A, B, C	A, B, C, D	A, B, C



UNI-профили

Разнообразие профильных методов позволяет использовать их для увеличения точности классификации:

- Комбинирование профилей для создания более сильных классификаторов
- Использование знаний о структуре документа для увеличения точность классификации

«Union» - объединение разных подходов к выявлению информативных терминов – статистического, теоретико-информационного, эвристического.

Для их построения используются различные комбинации PO-, НМИ-, J- и C-C-профилей, которые, за счет различных принципов определения наиболее информативных понятий, позволят скомпенсировать слабые стороны каждого из подходов.

Алгоритм построения профиля UNI6

- *Входными данными алгоритма* являются: обучающая выборка документов, моноязычные РО- , НМИ- и J-профили
- *Выходные данные:* профили классов, представленные в виде вектора терминов и упорядоченные по убыванию веса.
- *Шаг 1*. Задаются параметры метода h и t. L=h+t.
- *Шаг 2*. Суммируем веса профилей для каждого общего термина из русскоязычных и англоязычных исходных профилей:

$w_{uni6} = (w_{PO} + w_{НМИ} + w_J)$, , здесь w_{uni6} – вес термина в UNI6-профиле, w_{PO}, $w_{НМИ,}$ w_J– вес термина в РО-, НМИ-, J- профиле соответственно.

- *Шаг 3*. Среди русских терминов выбираем h терминов с наибольшими весами, среди английских - t терминов с наибольшими весами.
- *Шаг 4*. Полученные термины упорядочиваются по убыванию веса.

Как использовать структуру документа?

- Слова в разных частях документа (названия, аннотации, ключевые слова) неравнозначны
- Как учесть эту неравнозначность при классификации?

Семантическая интерпретация в системах компьютерного анализа текста

Описывается подход к построению семантического компонента в системах компьютерного анализа текста на естественном языке. Подход основан на применении специальных шаблонов к сети синтактико-семантических отношений между словами текста, которая строится синтаксическим анализатором. Шаблоны определяют способ интерпретации фрагментов сети в заданные фреймы с идентификацией участников ситуаций и их ролей.
Ключевые слова: компьютерный анализ текста, семантическая интерпретация, семантическая сеть, синтаксический анализ, фреймы.

Как использовать структуру документа? (2)

Подход №1. Использовать разные способы выявления информативных терминов для разных частей документа

$$W_k = \sum_{i=1}^{Mk} tf_T^{(i)} \Pr of_T(x^{(i)}, Q_k) + tf_A^{(i)} \Pr of_A(x^{(i)}, Q_k) + tf_K^{(i)} \Pr of_K(x^{(i)}, Q_k)$$

Подход №2. Использовать специальные веса для терминов из разных частей документа

$$W_k = \sum_{i=1}^{Mk} \alpha * tf_T^{(i)} \Pr of\ (x^{(i)}, Q_k) + \beta * tf_A^{(i)} \Pr of\ (x^{(i)}, Q_k) + \gamma * tf_K^{(i)} \Pr of\ (x^{(i)}, Q_k)$$

Подход №1

Использовать разные способы выявления информативных терминов для разных частей документа

$$W_k = \sum_{i=1}^{Mk} tf_T^{(i)} \Pr of_T(x^{(i)}, Q_k) + tf_A^{(i)} \Pr of_A(x^{(i)}, Q_k) + tf_K^{(i)} \Pr of_K(x^{(i)}, Q_k)$$

Основная проблема – выбор методов, которыми будут оцениваться термины из разных частей документов.

Решение:

1. Анализ встречаемости низко-, средне- и высокочастотных терминов в различных разделах БО с целью выбора таких методов, которые предпочитают отбирать в профиль и присваивать более высокие веса той категории терминов, которая наиболее часто появляется в разделе.
2. Полный перебор всех возможных вариантов путем комбинирования разных профилей.

Подход №2

Использовать специальные веса для терминов из разных частей документа

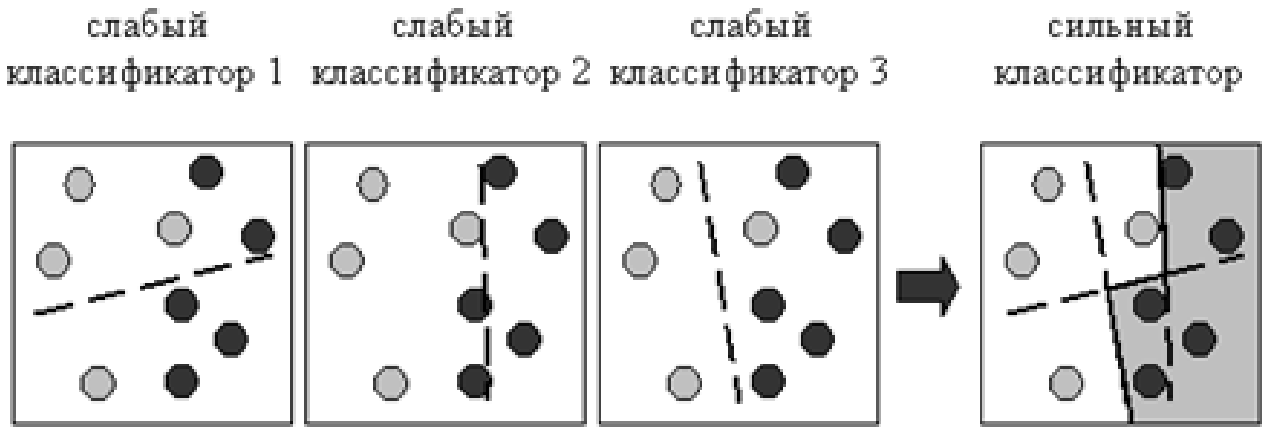
$$W_k = \sum_{i=1}^{Mk} \alpha * tf_T^{(i)} \Pr of\ (x^{(i)}, Q_k) + \beta * tf_A^{(i)} \Pr of\ (x^{(i)}, Q_k) + \gamma * tf_K^{(i)} \Pr of\ (x^{(i)}, Q_k)$$

Веса α, β, γ будем настраивать по методу Фишберна*.
В качестве $\Pr of\ (x^{(i)}, Q_k)$ будем использовать наиболее точный профиль.

*веса Фишберна - это рациональные дроби, в знаменателе которых стоит сумма арифметической прогрессии N первых членов натурального ряда с шагом 1, а в числителе - убывающие на 1 элементы натурального ряда от N до 1 (например, 3/6, 2/6, 1/6 в сумме дают единицу)

Коллективная классификация

Коллективом решающих правил (КРП) называется совокупность методов классификации, объединенных для выработки общего решения.



Коллективная классификация

Какие методы включать в коллектив?

- Наиболее точные
- Наиболее разнородные – ошибающиеся на разных объектах

Как померить разнородность?

– Использовать меры сходства (см. методы выявления информативных терминов)

Способы обеспечения дополнительной разнородности:
- обучение *КРП* с помощью методов *bagging* и *boosting*;
- обучение комитета классификаторов на различных независимых обучающих выборках;

Коллективная классификация (2)

Сколько методов включать в коллектив?

Вероятность правильной классификации в зависимости от количества и точности методов:

	m = 3	m = 5	m = 7	m = 9
p = 0,6	0,648	0,682	0,710	0,733
p = 0,7	0,784	0,837	0,874	0,901
p = 0,8	0,896	0,942	0,966	0,980
p = 0,9	0,972	0,991	0,997	0,999

Стратегии принятия решений

- *Простое голосование* – каждый классификатор имеет равный вес при принятии решения. Новое наблюдение относится к тому классу, за который проголосовало большинство членов *КРП*
- *Взвешенное голосование* – каждому классификатору присваивается вес в зависимости от количества допускаемых ошибок Δ_p (Δ_p –ошибка *p*–го классификатора,). Решение об отнесении нового наблюдения к какому-либо из классов принимается по формуле:

$$C(\vec{X}_{N+1}) = \sum_{p=1}^{m} \left(\frac{\Delta_p}{\sum_{s=1}^{m} \Delta_s} J_p \right).$$

- *Определение областей компетенции* для классификаторов, включенных в комитет (например, в случае неоднородных *КРП* можно выявить для каждого *p*–го решающего правила «зону ответственности», в которой классификатор ошибается меньше других

Что делать, если классификаторы не пришли к решению?

Вводят понятие «Отказ от классификации» (метка «Джокер»)

Если все члены комитета присваивают полностью не совпадающие метки одному и тому же объекту, то это означает, что, скорее всего, данный объект является нехарактерным шумовым элементом и к нему целесообразно применить операцию “*Отказ от классификации*”.

При этом наблюдения, получившие метку “*Джокер*” не включаются в расчет общей ошибки, т.е. в этом случае общая ошибка вычисляется по формуле:

$$\Delta = \frac{(N^{-})^{*}}{N^{*}}$$

$(N^{-})^{*}$ - число документов, отнесенных к неправильным классам, *$(N^{-})^{**}$* - число документов, получивших метку «Джокер», *N* – размер экзаменационной выборки, *$N^{*} = N - (N^{-})^{**}$*