Интеллектуальный анализ текстов

Курс «Интеллектуальные информационные системы»

Кафедра управления и интеллектуальных технологий

НИУ «МЭИ»

Осень 2021 г.

Text Mining – интеллектуальный анализ текстов

Интеллектуальный анализ текстов (*ИАТ*, *text mining*) — направление в искусственном интеллекте, целью которого является получение информации из коллекций текстовых документов, основываясь на применении эффективных в практическом плане методов машинного обучения и обработки естественного языка. (Wikipedia)

- **Категоризация текстов** (*classification*) –
  - отнесении документов из коллекции к одной или нескольким группам (классам, кластерам) схожих между собой текстов
- **Извлечение информации** (*information extraction*) –
  - это задача автоматического извлечения (построения) структурированных данных из неструктурированных или слабоструктурированных машиночитаемых документов (распознавание имен людей, названий организаций, поиск ключевых слов для текста, автореферирование)
- **Информационный поиск** (*information retrieval*) –
  - процесс поиска *неструктурированной* документальной информации, удовлетворяющей информационные потребности (процесс выявления в некотором множестве документов всех тех, которые посвящены указанной теме)

Проблемы, возникающие при работе с документами, написанными на ЕЯ

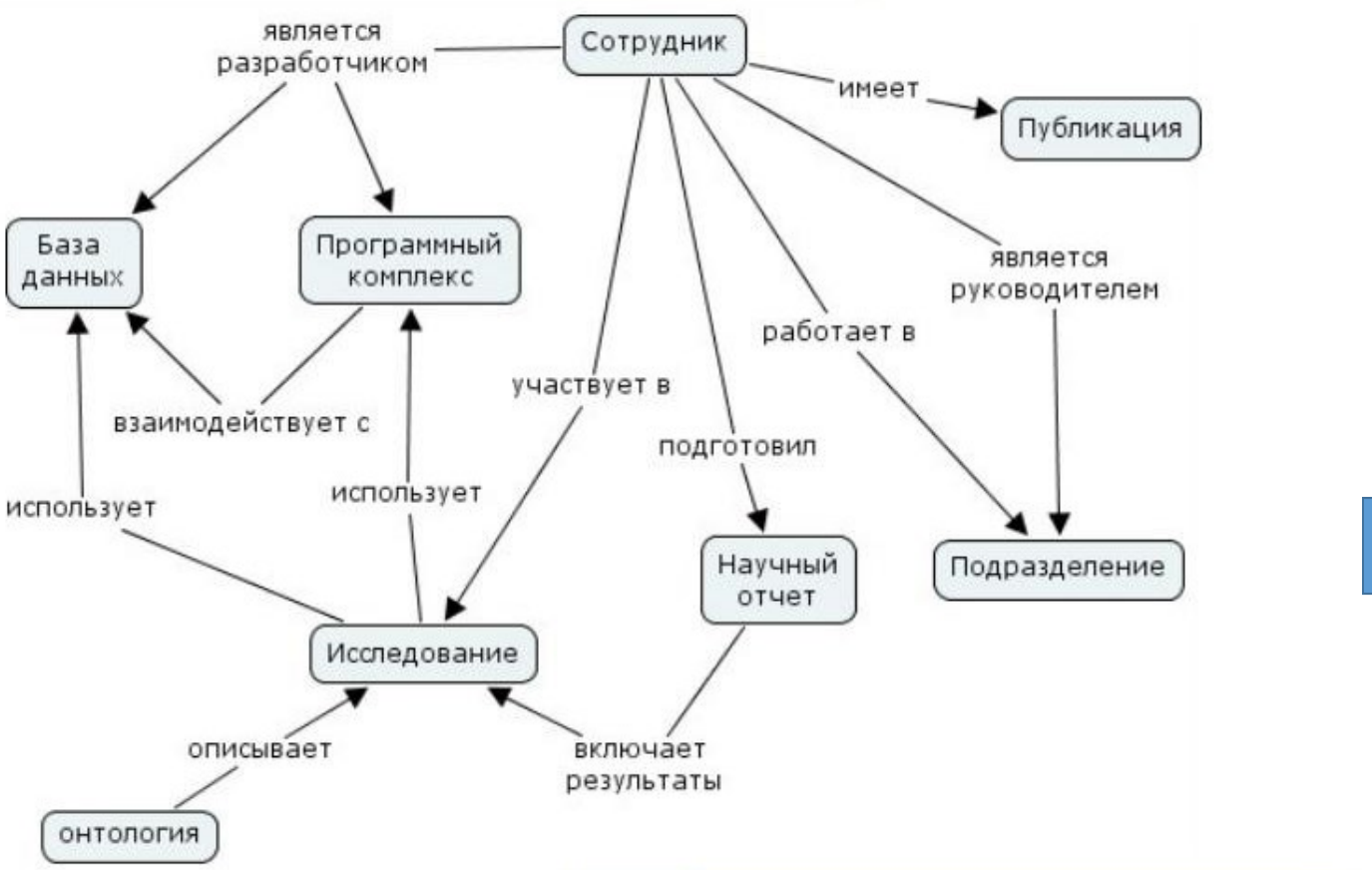
- ***Семантическая неоднозначность:***
  - *Синонимия: экран-дисплей*
  - *Полисемия: команда (судна; футбольная)*
  - *Омонимия: Ключ (родник) – Ключ (от замка)*
  - *Эллипсность: пропуски слов или слова-заменители*
- ***Многообразие средств передачи смысла:***
  - *Лексика ЕЯ*
  - *Контекст*
  - *Ссылки на слова*
- ***Высокая размерность задачи***
- ***Субъективность оценки качества классификации***
- ***Различная длина документов***

Подходы Text Mining

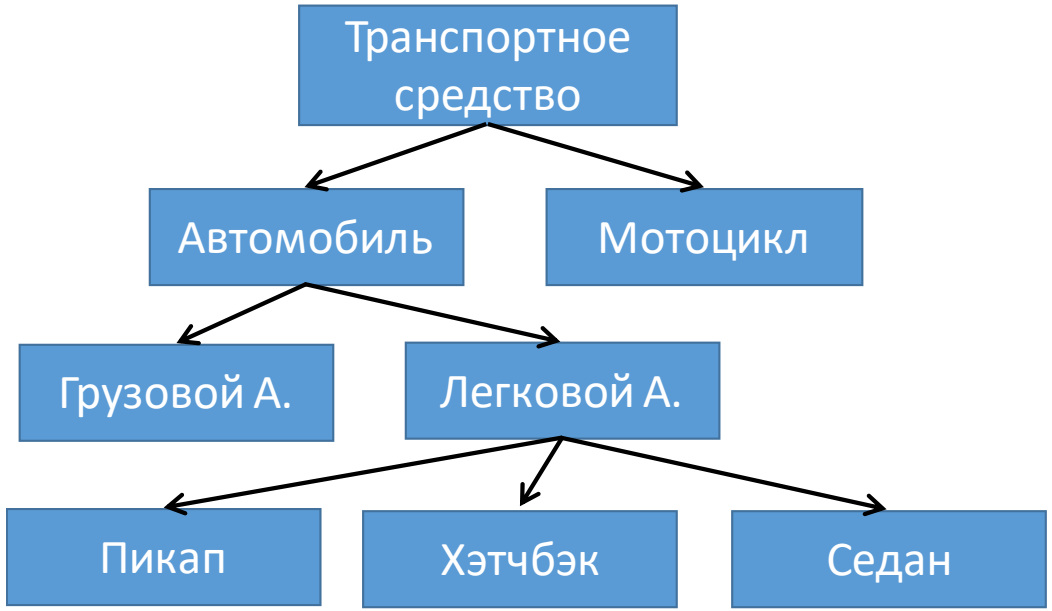
Лингвистический анализ	Статистический анализ
Системы ЛА обычно состоят из модели предметной области, содержащей основные тематические термины и их взаимосвязи, а также специализированной базы данных (БД) грамматических конструкций и семантических правил, свойственных конкретному языку – онтологий и тезаурусов. При этом модель предметной области обычно используется для проведения морфологического анализа, а специализированная БД – для синтаксического и семантического анализа	**Текст – набор ключевых слов. Вес слов зависит от различных факторов, в частности – от частоты встречаемости термина в документе. Предполагается, что появление одних и тех же терминов в различных документах говорит об их подобии**

Онтологии и тезаурусы

Пример онтологии



Пример тезауруса



Институт систем энергетики им Л.А.Мелентьева СО РАН

Что такое текст?

- Текст – конечное множество слов (терминов), объединенных лексическими, грамматическими, смысловыми, частотными отношениями и образующих информативное сообщение.
- Главное в тексте – информация, новая для читателя, которая заключена в авторском изложении, и которую мы хотим извлечь.
- Чем больше информации извлечем – тем лучше.
- Не всегда большой текст = большому количеству информации

Модели представления текстовых документов

- *Неструктурированная модель* – «мешок слов» (“bag of words”) – каждый термин рассматривается в качестве независимой случайной величины. Не учитываются возможные связи с другими словами в тексте.
- *Частично структурированная модель*
  - учет дополнительной информации о положении слова в тексте (заголовок, ключевые слова, первый абзац,…),
  - учет оформления слова (*курсив*, **полужирный**, подчеркивание,…),
  - выделение словосочетаний: $w = \frac{w_{kj}}{w_k w_j}$
- *Полностью структурированная модель*
  - *Использование информации из тезаурусов, онтологий, специальных словарей (WordNet)*

Как документ представляется в математическом виде?

Векторная модель:

Документ:

$$\vec{X}_j = \begin{bmatrix} x_j^{(1)} \\ \vdots \\ x_j^{(i)} \\ \vdots \\ x_j^{(M)} \end{bmatrix}$$

Матрица «Документ-термин»:

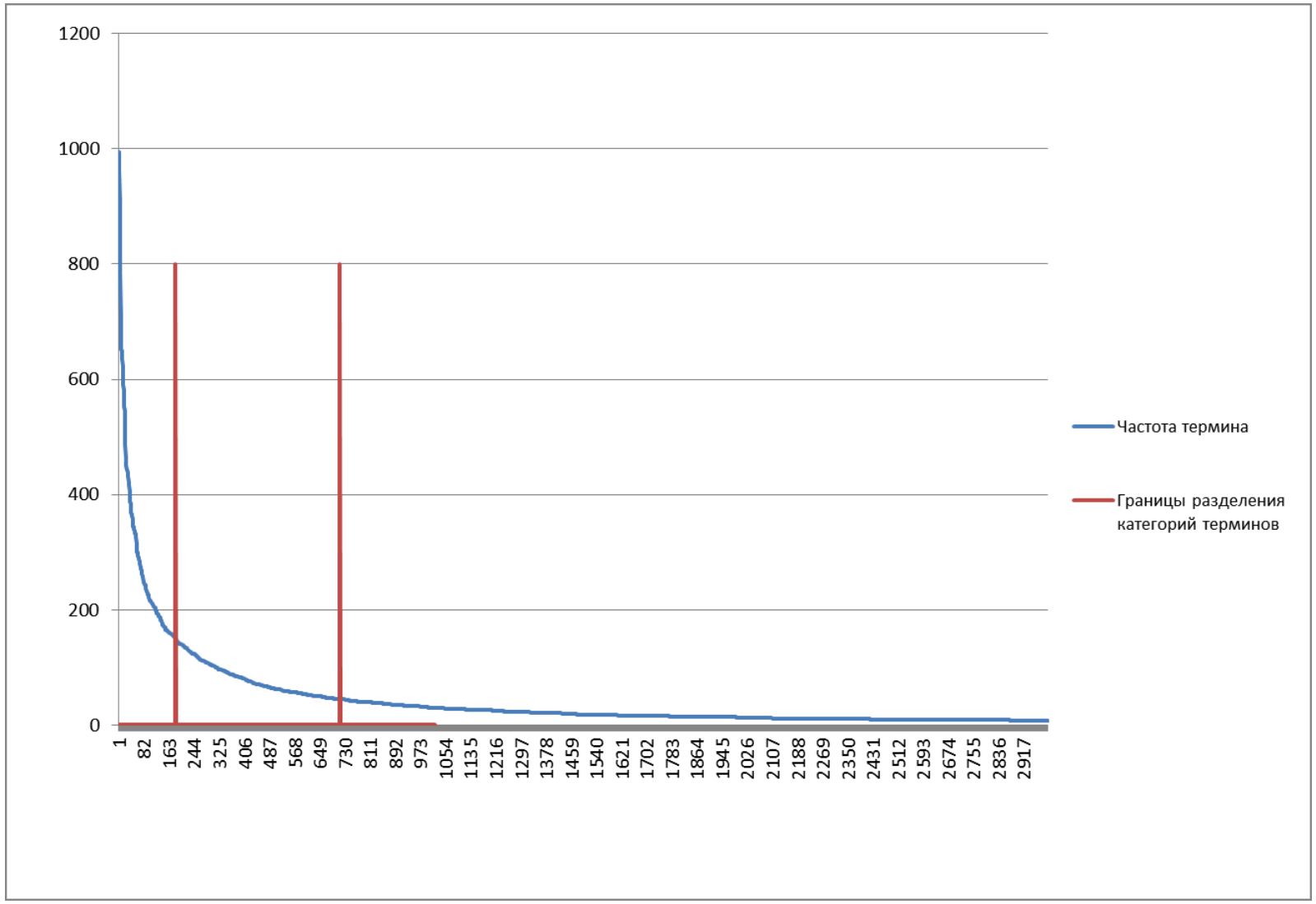
$$X = \begin{pmatrix} x_1^{(1)} & \dots & x_1^{(M)} \\ \vdots & \ddots & \vdots \\ x_N^{(1)} & \dots & x_N^{(M)} \end{pmatrix}$$

$x_j^{(i)}$ - Вес *термина* i в документе j
i = 1..M, j = 1..N

Размерность матрицы крайне высокая, $M \to 10^4 - 10^5$

* Вместо термина (или слова) могут использоваться n-граммы - последовательность из n элементов:
Триграммы Hello world: Hel, ell, llo, lo ,o w, wo, orl, rld.

Выявление информативных признаков



Закон Ципфа:
$w_n = w_1/n$

Предварительная обработка документов

Стемминг и отсечение стоп-слов

Стемминг — это процесс нахождения основы слова для заданного исходного слова. Основа слова не обязательно совпадает с морфологическим корнем слова. Наиболее известный алгоритм – Алгоритм Портера

Стоп-слова (шумовые слова) – слова, не несущие смысловой нагрузки – частицы, предлоги, местоимения,..

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.

Semantic Interpretation in Computer Text Analysis Systems

The article describes an approach to semantic component building in computer text analysis systems for a natural language text. The approach is based on applying special patterns to a net of syntactic and semantic relations between words in a text, which is formed by a syntactic parser. The patterns define the way to interpret parts of the net according to given frames with identification of participants of the situation and their roles.
Keywords: text mining, semantic interpretation, semantic network, syntactic parser, frames.

Определение весов терминов

Название	Формула
Логическое взвешивание	$x_j^{(i)} = \begin{cases} 1, & f_{ij} > 0 \\ 0, & f_{ij} = 0 \end{cases}$
Взвешивание частотой слова (term frequencies, tf)	$x_j^{(i)} = f_{ij}$
tf–idf - взвешивание (term frequencies – inverse document frequencies)	$x_j^{(i)} = f_{ij} \log\left(\frac{N}{N_i}\right)$
tfc - взвешивание	$x_j^{(i)} = \frac{f_{ij} \log\left(\frac{N}{N_i}\right)}{\sqrt{\sum_{i=1}^{M}\left[f_{ij} \log\left(\frac{N}{N_i}\right)\right]^2}}$

Определение весов терминов (2)

Название	Формула
ltc – взвешивание. Данный подход заключается в использовании логарифма частоты слова вместо f_{ij}. Это позволяет сократить характерный для большинства текстовых документов существенный разброс в частотах различных терминов	$x_j^{(i)} = \frac{\log(f_{ij}+1)\log\left(\frac{N}{N_i}\right)}{\sqrt{\sum_{i=1}^{M}\left[\log(f_{ij}+1)\log\left(\frac{N}{N_i}\right)\right]^2}}$
atc – взвешивание. При таком взвешивании веса будут изменяться от 0,5 до 1, что в ряде случаев приводит к улучшению качества классификации, позволяя учесть значимые термины, имеющие редкую встречаемость в конкретной выборке	$x_j^{(i)} = \frac{(0{,}5+0{,}5\frac{f_{ij}}{\max f})\log\left(\frac{N}{N_i}\right)}{\sqrt{\sum_{k=1}^{Mk}\left[(0{,}5+0{,}5\frac{f_{ij}}{\max f})\log\left(\frac{N}{N_k}\right)\right]^2}}$

Кроме взвешивания применяются и другие методы выявления информативных терминов:

- Факторный и компонентный анализ (переход к новой системе признаков)
- Статистический подход (Хи-квадрат критерий)
- Теоретико-информационный подход

Факторный анализ (ФА) и Метод Главных Компонент (МГК)

- ФА: различные признаки являются одним и тем же явлением, следовательно можно создать новые переменные – «факторы», позволяющие «вскрыть» логическую структуру выборки.
- МГК: переход к новым переменным, которые являются линейной комбинацией исходных.

Проведение снижения размерности с помощью ФА и МГК особенно эффективно для отображения объектов в трехмерное пространство и на плоскость.

Статистический подход выявления информативных признаков

Q / X	Q_1	Q_2	...	Q_K	$\sum_{k=1}^{K} n_{ik} = n_{i*}$
$x^{(1)}$	n_{11}	n_{12}	...	n_{1K}	n_{1*}
$x^{(2)}$	n_{21}	n_{22}	...	n_{2K}	n_{2*}
...	...	...	...	...	...
$x^{(M)}$	n_{M1}	n_{M2}	...	n_{MK}	n_{M*}
$\sum_{i=1}^{M} n_{ik} = n_{*k}$	n_{*1}	n_{*2}	...	n_{*K}	$n_{**} = N$

n_{ik} – клеточная частота – число объектов в выборке, обладающих данным сочетанием переменных $\{x^{(i)}, Q_k\}$

Проверяется гипотеза H_0: $n_{ik} - \hat{n}_{ik} = 0$

$$\hat{n}_{ik} = P(x^{(i)}, Q_k) = P(x^{(i)})P(Q_k) = N\frac{n_{i*}}{N}\cdot\frac{n_{*k}}{N} = \frac{n_{i*}n_{*k}}{N}$$

$$\chi^2 = \sum_{i=1}^{M}\sum_{k=1}^{K}\frac{(n_{ik} - \hat{n}_{ik})^2}{\hat{n}_{ik}}$$

Гипотеза о независимости отвергается с уровнем значимости α, если рассчитанная величина χ^2 превышает критическое значение $\chi^2_{\alpha,S}$

где S=(M-1)(K-1)

Частный случай Хи-квадрат критерия

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

$$\chi^2(x^{(i)},Q_k) = N\frac{(n_{11}n_{22}-n_{12}n_{21})^2}{n_{1*}n_{2*}n_{*1}n_{2*}} = N\frac{(AD-CB)^2}{(A+B)(C+D)(A+C)(B+D)}$$

$$\chi^2_{средний}(x^{(i)}) = \sum_{k=1}^{K} P(Q_k)\chi^2(x^{(i)},Q_k)$$

$$\chi^2_{\max 1}(x^{(i)}) = \max_{k=1}^{K} P(Q_k)\chi^2(x^{(i)},Q_k)$$

$$\chi^2_{\max 2}(x^{(i)}) = \max_{k=1}^{K} \chi^2(x^{(i)},Q_k)$$

$$P(Q_k) = \frac{N_k}{N}$$

Недостатки χ^2 - критерия:

- Вычислительная сложность
- Невысокая точность для редких терминов

Критерий взаимной информации (Mutual information)

Взаимная информация, как среднее количество информации, содержащееся в X относительно Q:

$I(X,Q) = H(X) - H(X \mid Q)$, где $H(X), H(X \mid Q) -$ соответственно энтропия и условная энтропия.

$$I(X,Q) = -\sum_{i=1}^{M} P(x^{(i)}) \log P(x^{(i)}) + \sum_{i=1}^{M} \sum_{k=1}^{K} P(x^{(i)}, Q_k) \log \frac{P(x^{(i)}, Q_k)}{P(Q_k)}$$

$$P(x_i) = P(Q_1)P(x^{(i)} \mid Q_1) + P(Q_2)P(x^{(i)} \mid Q_2) + \ldots + P(Q_K)P(x^{(i)} \mid Q_K) \quad = \quad P(x^{(i)}) = \sum_{k=1}^{K} P(x^{(i)}, Q_k)$$

Критерий взаимной информации (Mutual information) (2)

$$I(X,Q) = -\sum_{i=1}^{M} P(x^{(i)}) \log P(x^{(i)}) + \sum_{i=1}^{M}\sum_{k=1}^{K} P(x^{(i)},Q_k) \log \frac{P(x^{(i)},Q_k)}{P(Q_k)} =$$

$$= \sum_{i=1}^{M}\sum_{k=1}^{K} P(x^{(i)},Q_k) \log \frac{P(x^{(i)},Q_k)}{P(Q_k)P(x^{(i)})}.$$

$$P(x^{(i)}) = \frac{A+B}{N}$$

$$P(Q_k) = \frac{A+C}{N}$$

$$P(x^{(i)},Q_k) = \frac{A}{N}$$

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

$$I_{сред}(X,Q) = \sum_{k=1}^{K} P(Q_k) MI(x^{(i)},Q_k)$$

$$I_{\max}(X,Q) = \max_{k=1}^{K} \{MI(x^{(i)},Q_k)\}$$

Данный критерий, в отличие от Хи-квадрат, большие веса дает редким признакам

Векторное представление слова

или word embedding – как из слова получить вектор?

- Взять вектор размерностью = длине словаря, каждое слово в словаре – вектор, состоящий из 0 и одной 1 на соответствующей позиции (one-hot encoding, OHE)

‘стол’ = [0,0,0,0,0,1,0,0,0,0,0,0,0,0,0]

‘стул’ = [0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0] - не обладает свойством семантической близости

По сути – получили обычный «мешок слов»

- Хорошо бы, чтобы близкие друг к другу с семантической точки зрения слова имели бы близкие вектора.

Нужен вектор в «пространстве смыслов».

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

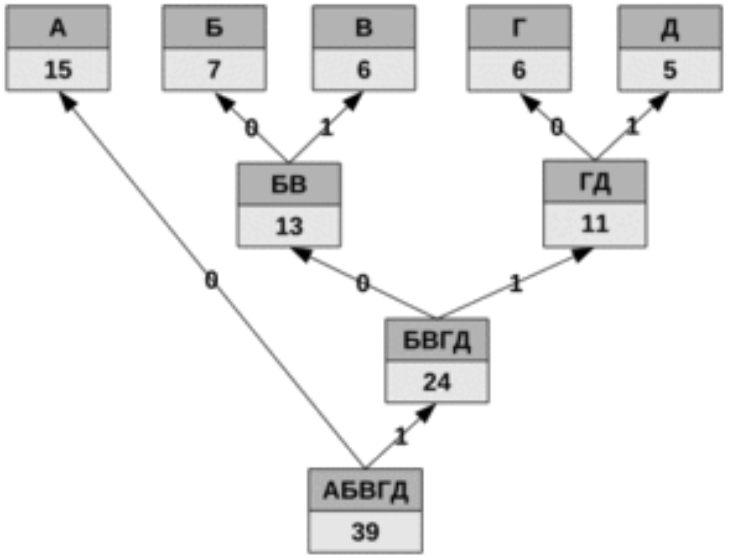
Работа этой технологии осуществляется следующим образом: word2vec принимает большой текстовый корпус в качестве входных данных и сопоставляет каждому слову вектор, выдавая координаты слов на выходе. Сначала он создает словарь, «обучаясь» на входных текстовых данных, а затем вычисляет векторное представление 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



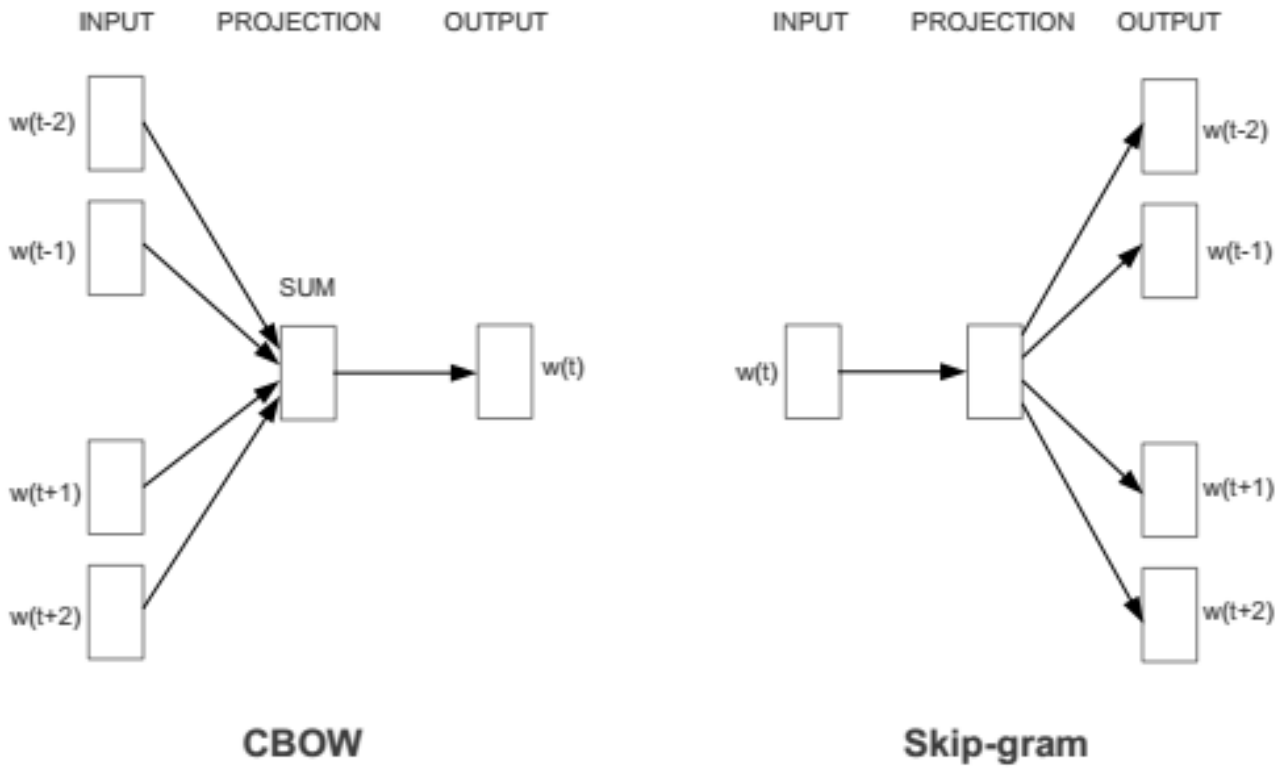
Word2Vec – как работает?

- 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.



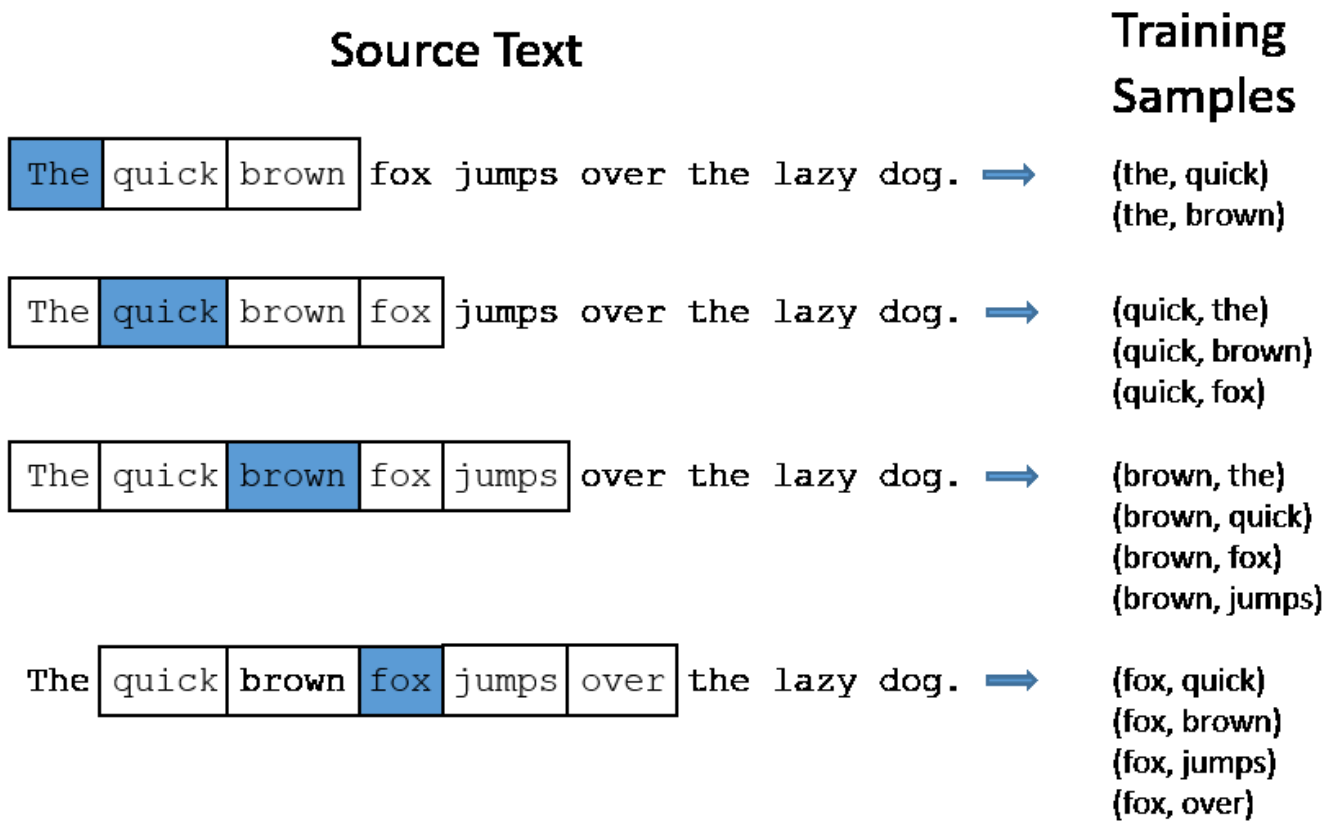
Skip-gram

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 “моторное", то для таких слов, как “масло” и “двигатель”, полученная вероятность будет намного выше, чем для несвязанных слов, например, “яблоко” и “стул”.

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 «двигатель» появляется в контексте автомобилей, то вектор слова «двигатель» будет ближе к вектору слова «автомобиль», чем вектора некоторых иных случайно выбранных слов (например стол, 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)]
```