Труды Карельского научного центра РАН
№0. 2015. С. 1–24

УДК 81.32

ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РАЗРЕШЕНИЯ ЛЕКСИЧЕСКОЙ МНОГОЗНАЧНОСТИ: ВВЕДЕНИЕ

Т. В. Каушинис[2], А. Н. Кириллов[1], Н. И. Коржицкий[2], А. А. Крижановский[1], А. В. Пилинович[2], И. А. Сихонина[2], А. М. Спиркова[2], В. Г. Старкова[1], Т. В. Степкина[2], С. С. Ткач[2], Ю. В. Чиркова[1], А. Л. Чухарев[1], Д. С. Шорец[2], Д. Ю. Янкевич[2], Е. А. Ярышкина[2]

[1] *Институт прикладных математических исследований Карельского научного центра РАН*
[2] *Петрозаводский Государственный Университет*

Разрешение лексической многозначности — это задача выбора между разными значениями слов и словосочетаний в словаре в зависимости от контекста. В статье представлен краткий обзор методов и алгоритмов разрешения лексической многозначности. Представлены (1) методы, основанные на машинном обучении, (2) методы, не использующие никаких размеченных корпусов для различения значений слов, и (3) методы, использующие внешние словарные источники информации (машиночитаемые словари, тезаурусы, онтологии). Статья распространяется на правах свободной лицензии “CC Attribution”.

Ключевые слова: алгоритм, метод, разрешение лексической многозначности.

T. V. Kaushinis[2], A. N. Kirillov[1], N. I. Korzhitsky[2], A. A. Krizhanovsky[1], A. V. Pilinovich[2], I. A. Sikhonina[2], A. M. Spirkova[2], V. G. Starkova[1], T. V. Stepkina[2], S. S. Tkach[2], J. V. Chirkova[1], A. L. Chuharev[1], D. S. Shorets[2], D. Y. Yankevich[2], E. A. Yaryshkina[2]. **WORD-SENSE DISAMBIGUATION METHODS AND ALGORITHMS REVIEW: INTRODUCTION**

The word sense disambiguation (WSD) task is the classification task, where the goal is to predict the meaning of words and phrases with the help of surrounding text. The purpose of this short review is to acquaint the reader with the general directions of word sense disambiguation methods and algorithms. The paper consists of three main parts: (1) supervised (machine learning) approaches to WSD, (2) unsupervised approaches based on unlabeled corpora and (3) knowledge-based approaches to WSD, where machine-readable dictionaries, thesauri, ontologies are used. This work is licensed under the CC Attribution license.

Key words: algorithm, method, word-sense disambiguation.

Введение

В статье представлен обзор методов и алгоритмов разрешения лексической многозначности (word-sense disambiguation или WSD). Верный выбор в словаре одного из значений многозначного слова или фразы в зависимости от контекста и является успешным результатом решения WSD-задачи.

Приведем несколько примеров употребления слова «коса» и «косой», найденных с помощью Национального корпуса русского языка (http://ruscorpora.ru) по запросу «коса»:

1. Поп сам в первой *косе* идет, но прихожане не торопятся, смотрят на солнышко и часа через полтора уже намекают, что обедать пора. [*М. Е. Салтыков-Щедрин. Мелочи жизни (1886-1887)*]
2. Но работа даже и после этого идет все вялее и вялее; некоторые и *косы* побросали. [*М. Е. Салтыков-Щедрин. Мелочи жизни (1886-1887)*]
3. В особенности жестоко было крепостное право относительно дворовых людей: даже волосы крепостных девок эксплуатировали, продавая их *косы* парикмахерам. [*М. Е. Салтыков-Щедрин. Мелочи жизни (1886-1887)*]
4. Это одинокая скала, соединяющаяся с материком намывной *косой* из песка и гальки. [*В. К. Арсеньев, «По Уссурийскому краю», 1917 г.*]
5. Первая черепашка подскочила к гвардейцу и воткнула ему в спину сверкающий *косой* меч. [*Виктор Пелевин. S.N.U.F.F, 2011*]

Первые четыре примера дают три разных значения существительного «коса»: ряд косарей, сельскохозяйственное орудие, заплетенные волосы, протяженная речная отмель. Последний пример содержит прилагательное «косой», совпадающее с одной из форм существительного «коса». Все эти значения и часть речи читатель легко определяет по контексту.

Именно многозначность слов, их неоднозначность и зависимость значений слов от контекста являются причиной возникновения такой задачи и одновременно обуславливают сложность ее решения. Уверенное решение WSD-задачи необходимо во многих приложениях, связанных с автоматической обработкой текста (информационный поиск, машинный перевод и т.п.) и, на наш взгляд, является предтечей искусственного интеллекта.

Для этой задачи известно большое количество алгоритмов и методов решения, которые можно разделить на [4], [41]:

- WSD-методы с учителем (*supervised*) — методы, базирующиеся на машинном обучении и работающие на размеченных корпусах текстов;
- WSD-методы без учителя (*unsupervised*), не использующие никаких размеченных корпусов для различения значений слов.

Другая классификация методов строится на противопоставлении используемых ресурсов [41]:

- WSD-методы, основанные на знаниях (*knowledge-based*); в этих методах используются внешние словарные источники информации (машиночитаемые словари, тезаурусы, онтологии);
- WSD-методы, основанные на корпусах текстов (*corpus-based*).

Также применяются комбинации этих методов.

На сегодняшний день на русском языке нет, по-видимому, достаточно объёмных и серьёзных обзоров по разрешению многозначности. Наиболее полное описание истории развития методов (20 страниц) есть в диссертации Д. Ю. Турдакова [5]. Такое положение дел послужило одной из причин написания этой статьи, которая послужит заделом для полновесного обзора по данной теме.

Далее будут представлены примеры методов и алгоритмов разрешения лексической многозначности (1) с использованием машинного обучения, (2) без машинного обучения и (3) методы, основанные на знаниях. Данная статья является «введением» в проблематику WSD, поскольку эта тема является чрезвычайно обширной и существуют сотни интересных работ по каждому из указанных направлений.

WSD-методы с учителем

Разрешение лексической многозначности методом ансамбля байесовских классификаторов

А. Л. Чухарев, Т. В. Каушинис

В работе Педерсена [43] рассматривается подход к разрешению лексической многозначности слов (WSD), подразумевающий создание ансамбля наивных байесовских классификаторов, каждый из которых основан на оценке вероятности вхождения определенных слов

в контекст целевого слова, значение которого определяется.

При разрешении лексической многозначности, представленной как задача обучения с учителем, применяют статистические методы и методы машинного обучения к размеченному корпусу. В таких методах словам корпуса, для которых указано значение, соответствует набор языковых свойств. Педерсен [43] относит к языковым свойствам два вида особенностей: так называемые простые лексические особенности (shallow lexical features) и более сложные лингвистически обусловленные особенности (lingvistically motivated features). К первым относятся совместная встречаемость слов (co-occurence) и словосочетания (collocations), в то время как вторые включают в себя такие свойства как часть речи и отношение действие-объект. Обычно алгоритмы обучения строят модели классификаторов значений по этим языковым свойствам.

Автор статьи [43] предлагает подход, основанный на объединении ряда простых классификаторов в ансамбль, который разрешает многозначность с помощью голосования простым большинством голосов. Педерсен утверждает [43], что, во-первых, более сложные алгоритмы обычно не улучшают точность разрешения. Во-вторых, совместная встречаемость слов и словосочетаний имеют большее влияние на точность разрешения, чем оперирование более сложной лингвистической информацией.

В рассматриваемой статье [43] в ансамбль объединяются наивные байесовские классификаторы. При таком подходе предполагается, что все переменные, участвующие в представлении проблемы, — условно независимы при фиксированном значении переменной классификации. В проблеме разрешения лексической многозначности существует понятие контекста, в котором встречается многозначное слово. Этот контекст представляется в виде функции переменных $(F_1, F_2, \ldots, F_n)$, а значение многозначного слова представлено в виде классификационной переменной (S). Все переменные бинарные. Переменная, соответствующая слову из контекста, принимает значение ИСТИНА, если это слово находится на расстоянии определенного количества слов слева или справа от целевого слова. Совместная вероятность наблюдения определенной комбинации переменных контекста с конкретным значением слова выражается следующим образом:

$$p(F_1, F_2, \ldots, F_n S) = p(S)\Pi_{i=1}^{n} p(F_i \vee S)$$

где $p(S)$ и $p(F_i|S)$ — параметры данной модели. Для оценки параметров достаточно знать частоты событий, описываемых взаимозависимыми переменными (F_i, S). Эти значения соответствуют числу предложений, где слово, представляемое F_i, встречается в некотором контексте многозначного слова, упомянутого в значении S. Если возникают нулевые значения параметров, то они сглаживаются путем присвоения им по умолчанию очень маленького значения. После оценки всех параметров модель считается обученной и может быть использована в качестве классификатора.

Контекст в [43] представлен в виде bag-of-words (модель «мешка слов»). В этой модели выполняется следующая предобработка текста: удаляются знаки препинания, все слова переводятся в нижний регистр, все слова приводятся к их начальной форме (лемматизация). В [43] контексты делятся на два окна: левое и правое. В первое попадают слова, встречающиеся слева от неоднозначного слова, и, соответственно, во второе — встречающиеся справа.

Окна контекстов могут принимать 9 различных размеров: 0, 1, 2, 3, 4, 5, 10, 25 и 50 слов. Первым шагом в ансамблевом подходе является обучение отдельных наивных байесовских классификаторов для каждого из 81 возможных сочетаний левого и правого размеров окон. В статье [43] наивный байесовский классификатор (l, r) включает в себя l слов слева от неоднозначного слова и r слов справа. Исключением является классификатор (0,0), который не включает в себя слов ни слева, ни справа. В случае нулевого контекста классификатору присваивается **априорная вероятность** многозначного слова (равная вероятности встретить наиболее употребимое значение).

Следующий шаг в [43] при построении ансамбля — это выбор классификаторов, которые станут членами ансамбля. 81 классификатор группируется в три общие категории, по размеру окна контекста. Используются три таких диапазона: узкий (окна шириной в 0, 1 и 2 слова), средний (3, 4, 5 слов), широкий (10, 25, 50 слов). Всего есть 9 возможных комбинаций, поскольку левое и правое окна отделены друг от друга. Например, наивный байес (1,3) относится к диапазону категории (узкий, средний) поскольку он основан на окне из одного слова слева и окне из трех слов справа. Наиболее точный классификатор в каждой из 9 категорий диапазонов выбирается для включения в ансамбль. Затем каждый из 9 членов классификаторов голосует за наиболее вероятное значение слова с учетом контекста. После

этого ансамбль разрешает многозначность путем присвоения целевому слову значения, получившего наибольшее число голосов.

Экспериментальные данные. Для экспериментов были выбраны английские слова *line* и *interest*. Источником статистических данных по этим словам послужили работы [32], [10]. В статье приводится информация о частоте использования шести значений для каждого из этих слов (Табл. 1, Табл. 2).

Таблица 1: Число употреблений слова *line* (столбец *count*) для шести наиболее часто встречаемых значений (значения из тезауруса WordNet, столбец *sense*) по данным корпусов *ACL/DCI Wall Street Journal и American Printing House for the Blind*

sense	count
product	2218
written or spoken text	405
telephone connection	429
formation of people or things; queue	349
an artificial division; boundary	376
a thin, flexible object; cord	371
total	4148

Таблица 2: Число употреблений слова *interest* (столбец *count*) для шести наиболее часто встречаемых значений (значения из словаря Longman Dictionary of Contemporary English, столбец *sense*). Этот набор данных был получен в 1994 году Брюсом и Виебе [10] путем указания значений для всех вхождений слова *interest* в корпус ACL/DCI Wall Street Journal

sense	count
money paid for the use of money	1252
a share in a company or business	500
readiness to give attention	361
advantage, advancement or favor	178
activity that one gives attention to	66
causing attention to be given to	11
total	2368

Результаты экспериментов. Итогом проделанной работы стали обучение и проверка 81 наивного байесовского классификатора на многозначных словах *line* и *interest*. Точность разрешения лексической многозначности составила 89% для слова *interest* и 88% для слова *line*. В [43] было получено, что ансамбль классификаторов с голосованием простым большинством дает более высокую точность, чем взвешенное голосование. Например, для слова *interest* при голосовании простым большинством точность составила 89%, а взвешенное голосование дало только 83%.

WSD на основе нейронных сетей, построенных по данным машиночитаемых словарей

А. Н. Кириллов

Использование нейронных сетей (NN) для WSD было предложено в 80-е гг. в работах [12, 55]. В типичной NN на вход подается слово, значение которого требуется установить, т.е. целевое (*target*) слово, а также — контекст (фраза) его содержащий. Узлы выхода соответствуют различным значениям слова. В процессе обучения, когда значение тренировочного целевого слова известно, веса связующих узлы соединений (связей), настраиваются таким образом, чтобы по окончании обучения выходной узел, соответствующий истинному значению целевого слова, имел наибольшую активность. Веса соединений могут быть положительными или отрицательными, и настраиваются посредством рекуррентных алгоритмов (алгоритм обратного распространения ошибки, рекуррентный метод наименьших квадратов и т.д.). Сеть может содержать скрытые (hidden) слои, состоящие из узлов, соединенных как прямыми, так и обратными связями. Для представления входной информации обычно используется одна из двух схем: распределенная (distributed) или локалистская (localist) ([27], [13], [7]).

В работе [51] описан метод автоматического построения ***очень больших нейронных сетей*** (VLNN) с помощью текстов, извлекаемых из машиночитаемых словарей (MRD), и рассмотрено использование этих сетей в задачах разрешения лексической неоднозначности. Поясним основную идею VLNN. Широко известен метод Леска [33] использования информации из MRD для задачи WSD. Суть этого метода состоит в вычислении так называемой *степени пересечения*, т.е. количества общих слов в словарных определениях слов из контекста («окна») условного размера, содержащего целевое слово. Основной недостаток метода Леска — зависимость от словарной статьи, то есть от слов, входящих в нее. Стратегия преодоления этого недостатка — использование словарных статей, определяющих слова, входящие в другие словарные статьи, начиная со словарных статей, соответствующих словам из контекста. Таким образом, образуются достаточно длинные пути из слов, входящих в словарные статьи. Эта идея лежит в

основе топологии VgN. В работе [51] для построения VLNN использован словарь Collins English Dictionary.

Топология сети. Целевое слово представлено узлом, соединенным активирующими связями со смысловыми узлами, представляющими все возможные значения слова, имеющиеся в словарных статьях. Каждый смысловой узел, в свою очередь, соединен активирующими связями с узлами, представляющими слова в словарной статье, соответствующей толкованию данного значения. Процесс соединения повторяется многократно, создавая сверхбольшую сеть взаимосвязанных узлов. В идеале сеть может содержать весь словарь. Авторы, по практическим соображениям, ограничиваются несколькими тысячами узлов и 10–20 тысячами соединений. Слова представлены своими леммами (каноническими формами). Узлы, представляющие различные значения данного слова, соединены запрещающими (inhibitory) связями.

Алгоритм. При запуске сети первыми активируются узлы входного слова (согласно принятой кодировке). Затем каждый входной узел посылает активирующий сигнал своим смысловым узлам, с которыми он соединен. В результате сигналы распространяются по всей сети в течение определенного числа циклов. В каждом цикле узлы слова и его значений получают обратные сигналы от узлов, соединенных с ними. Узлы конкурирующих значений посылают взаимно подавляющие сигналы. Взаимодействие сигналов обратной связи и подавления, в соответствии со стратегией «победитель получает все», позволяет увеличить активацию узлов-слов и соответствующих им правильных узлов-значений, одновременно уменьшая активацию узлов, соответствующих неправильным значениям. После нескольких десятков циклов сеть стабилизируется в состоянии, в котором активированы только узлы-значения с наиболее активированными связями с узлами-словами. При обучении сети используется метод обратного распространения (*back propagation*).

Краткое введение в бустинг

Т. Степкина, Ю. В. Чиркова

В статье [21] говорится о бустинге — общем методе улучшения точности алгоритма обучения. В работе приводится алгоритм AdaBoost с описанием необходимой теории бустинга, а также показывается связь с методом опорных векторов и объясняются причины, по которым переобучение не влияет на бустинг.

Бустинг — это общий и доказуемо-эффективный метод получения очень точного правила предсказания путем комбинирования грубых и умеренно неточных эмпирических правил [21]. Метод бустинга разработан на основе модели обучения "PAC"(probably approximately correct learning).

Алгоритм AdaBoost был предложен в 1995 году Фройндом и Шапиро [22]. В нём исправлены многие недостатки предыдущих алгоритмов бустинга.

AdaBoost является адаптивным алгоритмом [21], поскольку он может адаптироваться к уровням ошибок отдельных слабых гипотез. В названии "Ada"является сокращением от «adaptive» (адаптивный).

На вход алгоритма поступает обучающая выборка $(x_i; y_i); ..; (x_m; y_m)$, где каждый элемент x_i принадлежит некоторому домену или признаковому пространству X, и каждая метка y_i принадлежит некоторому набору меток Y. Для каждого обучающего примера i вес распределения для целых t обозначается D_t (i), где t — это шаг алгоритма. За начальное распределение весов принимается D_1 $(i) = 1/m$. Пусть метки принимают значения из множества $Y = \{-1, 1\}$.

Далее на каждом шаге t, где $t = 1 \ldots T$, выполняется обучение слабого обучаемого с использованием текущего распределения D_t, после чего строится слабая гипотеза $h_t : X \to \{-1; 1\}$ с ошибкой первого рода $\varepsilon_t = \Sigma_{(i\,:\,h_t(x_i) \neq y_i)} D_t(i)$, по которой выбирается уровень значимости $\alpha_t = \frac{1}{2} ln(\frac{1-\varepsilon_t}{\varepsilon_t})$ и строится новое распределение для следующего шага

$$D_{t+1}(i) = \frac{D_{t^i}}{Z_t} \times \begin{cases} e^{-\alpha t}, \text{ если } h_t(x_i) = y_i \\ e^{\alpha t}, \text{ если } h_t(x_i) \neq y_i \end{cases} =$$

$$= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}.$$

Конечная гипотеза $H(x)$ — это среднее из большинства решений T слабых гипотез, где α_t — вес, присвоенный гипотезе h_t.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Идея алгоритма заключается в выборе набора весов для обучающей выборки. Первоначально все веса примеров устанавливаются одинаково, но на каждом круге цикла веса неправильно классифицированных по гипотезе h_t примеров увеличиваются, таким образом, получают веса, которые относятся к сложным примерам.

Основное теоретическое свойство AdaBoost — это способность алгоритма уменьшать ошибку обучения [21]. Фройнд и Шапиро показали, что, так как каждая слабая гипотеза немного лучше случайного выбора, ошибка обучения уменьшается с экспоненциальной скоростью.

В статье [21] показано как ограничена ошибка обобщения конечной гипотезы в терминах ошибки обучения, размера выборки m, VC размерности (размерности Вапника — Червоненкиса [50]) пространства слабых гипотез и количества циклов T. Также получена граница, не зависящая от T. Это показывает, что бустинг AdaBoost не подвержен эффекту переобучения.

Так как ошибка обучения и ошибка обобщения ограничены, как показано в статье [21], этот алгоритм действительно является бустинговым алгоритмом в том смысле, что он может эффективно преобразовать слабый алгоритм обучения в сильный, который может породить гипотезу со сколь угодно малой частотой ошибок, имея достаточное количество данных.

После того, как авторы рассмотрели бинарный случай, где целью является различие лишь между двумя возможными классами, они переходят к рассмотрению мультиклассного, более приближенного к реальности. Есть несколько способов приведения AdaBoost к мультиклассному случаю. Самое простое обобщение называется AdaBoost.M1 [23], которое является приемлемым, если слабый обучаемый может достичь достаточно высокой точности на распределениях, созданных AdaBoost. Тем не менее, этот метод завершается неудачно, если слабый ученик не может достичь хотя бы 50% точности при работе на этих распределениях. Для такого случая было разработано несколько методов:

1. Методы, которые работают за счет преобразования мультиклассной задачи в большую бинарную задачу или в набор бинарных задач. Однако, эти методы требуют дополнительных усилий в разработке слабого алгоритма обучения.

2. Технология, которая включает в себя метод Диттерича и Бакири — метод выходных кодов, исправляющих ошибки [50].

AdaBoost обладает определенными преимуществами. Его быстро и просто запрограммировать. Он не имеет никаких параметров для настройки, за исключением количества циклов. Он не требует никаких предварительных знаний о слабом обучаемом и поэтому может быть скомбинирован с любым методом для нахождения слабых гипотез.

Недостатки метода заключаются в следующем. Фактическая производительность бустинга на конкретной задаче явно зависит от данных и слабого обучаемого. Теоретически, бустинг может выполниться плохо, если данных недостаточно, слабые гипотезы слишком сложные, или наоборот слишком слабые. Также бустинг особенно восприимчив к шуму.

AdaBoost был протестирован эмпирическим путем многими исследователями. Например, Фройнд и Шапиро проверили AdaBoost на множестве эталонных наборов данных UCI [36] с использованием C4.5 [47] как слабого алгоритма обучения, а также алгоритм, который находит самое лучшее дерево решений с одним тестом. После проведения эксперимента был сделан вывод, что бустинг даже слабых деревьев решений с одним тестом, как правило, дает хорошие результаты, в то время как бустинг C4.5, как правило, дает алгоритм дерева принятия решений значительно улучшенной производительности.

Почти во всех этих экспериментах и для всех показателей эффективности бустинг работает так же хорошо или значительно лучше, чем в других методах испытаний. Бустинг также применяется к фильтрации текстов, проблемам ранжирования и проблемам классификации, возникающих при обработке естественного языка.

Сравнительные эксперименты в WSD: роль предпочтений в машинном обучении

Н. И. Коржицкий

В заключение главы, посвященной разрешению многозначности, приведем эксперементальное сравнение методов. В работе Рэймонда Муни [38] представлено одно из первых сравнений разных по природе методов WSD на одних и тех же данных. В статье [38] проведена серия экспериментов, в которых сравнивалась способность различных обучающихся алгоритмов определять значение слова в зависимости от контекста.

В машинном обучении под термином *bias* (пристрастие, тенденция, предпочтение) понимается любое основание для выбора одного обобщения другому, вместо строгого соответствия примерам [38]. В деревьях принятия решений предпочтение (*bias*) отдается простым деревьям решений, в нейронных сетях — линейным пороговым функциям, а в байесовском классификаторе — функциям, учитывающим

условную независимость свойств. Чем лучше «предпочтение» обучающегося алгоритма соответствует характеристикам конкретной задачи, тем лучше будет результат. Большинство обучающихся алгоритмов обладают «предпочтением» наподобие Бритвы Оккама, в таких алгоритмах выбираются гипотезы, которые могут быть представлены меньшим количеством информации на каком-нибудь языке представлений. Однако компактность, с которой (деревья решений, дизъюнктивная нормальная форма, сети с линейным пороговым значением) представляют конкретные функции — может существенно различаться. Поэтому различные «предпочтительные» оценки могут работать лучше или хуже в конкретных задачах. Одной из основных целей в машинном обучении является поиск «предпочтений» с целью решения прикладных практических задач.

Выбор правильного «предпочтения» и обучающегося алгоритма является сложной задачей. Простым подходом является автоматизация выбора метода при помощи внутренней перекрестной валидации. Другой подход metalearning заключается в том, чтобы обучиться набору правил (или другому классификатору), предсказывающему, когда обучающийся алгоритм будет срабатывать наилучшим образом на примере с набором свойств присущих проблеме.

Описанный в [38] эксперимент заключается в определении значения слова *line* (англ. *линия*) среди 6 возможных вариантов (*строка, ряд, дивизия, телефон, веревка, продуктовая линия*). Данные для проведения экспериментов взяты из работы [32].

Для получения обучающей выборки брались предложения со словом *line*, и им в соответствие ставилось одно из 6 значений. Распределение значений неравномерно: включение в список иточников журнала The Wall Street Journal привело к тому, что одно из значений встречалось в 5 раз чаще всех остальных [32].

Таблица 3: Шесть значений слова *line* из Английского Викисловаря и Русского Викисловаря

ключевое слово	**перевод**	**толкование на английском (Английский Викисловарь)**	**толкование на русском (Русский Викисловарь)**
text	строка	A small amount of text	ряд слов, букв или иных знаков, написанных или напечатанных в одну линию
formation	ряд	A more-or-less straight sequence of people, objects, etc., either arranged as a queue or column and often waiting to be processed or dealt with, or arranged abreast of one another in a row (and contrasted with a column), as in a military formation	несколько объектов, расположенных в линию или следующих один за другим
division	дивизия	A formation, usually made up of two or three brigades	тактическое воинское соединение
phone	телефон	The wire connecting one telegraphic station with another, a telephone or internet cable between two points: a telephone or network connection	то же, что телефонный номер
cord	веревка	A rope, cord, string, or thread, of any thickness	гибкое и длинное изделие, — чаще всего сплетенное или свитое из льняных (или пеньковых, полимерных и т. п.) волокон или прядей
product	продуктовая линия	The products or services sold by a business, or by extension, the business itself	совокупность однородной продукции единого назначения

В работе [14] было установлено, что наиболее эффективными при решении задачи WSD являются алгоритмы на основе дерева решений (decision tree). Данный класс методов обходил по точности и скорости работы класс нейронных сетей. Другие исследования [39] показали, что класс методов индуктивного логического программирования (inductive logic programming) справляется с задачей разрешения лексической многозначности слова лучше алгоритмов на основе дерева решений.

В серии экспериментов в [38] сравнивались следующие методы: байесовский классификатор, перцептрон, C4.5, метод k-ближайших соседей и модификации алгоритма FOIL: PFOIL-DLIST, PROIL-DNF, PFOIL-CNF. Все алгоритмы были реализованы на языке Common Lisp, за исключением C4.5, который был написан на языке C.

После проведения сравнительных экспериментов, заключавшихся в обучении и определении значения слова *line,* было выяснено, что байесовский классификатор и перцептрон работают точнее других рассмотренных методов.

Эксперименты проводились с разными размерами обучающей выборки для того, чтобы выяснить, какого рода зависимость имеет место между точностью определения значения и размером выборки. На Рис. 1 отображена зависимость точности работы алгоритмов от размера выборки. При увеличении размера обучающей выборки сначала происходит резкий рост точности, последующий прирост точности становится незначительным.

Эксперименты учитывали не только точность определения значения, но и требовательность алгоритма к ресурсам в процессе обучения и работы. На Рис. 2 можно увидеть зависимость времени обучения от размера выборки. Самыми быстрообучаемыми оказались байесовский классификатор и перцептрон, а самыми медленными — нормальные формы (Рис. 2).

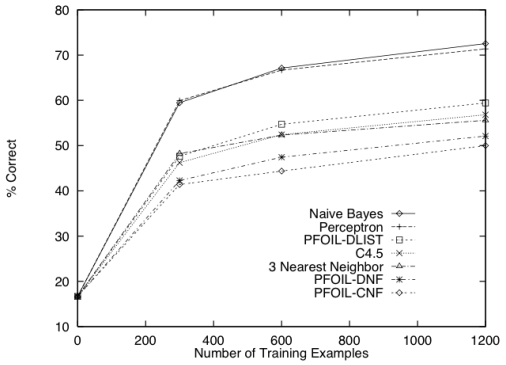


Рис. 1: Рост точности решения WSD задачи для разных алгоритмов (определение значения слова *line*) при увеличении размера обучающей выборки [38]. Алгоритмы: PFOIL-DLIST, PROIL-DNF, PFOIL-CNF, C4.5, Naive Bayes — наивный баейсовский классификатор; Perceptron — персептрон; 3 Nearest Neighbor — метод 3-х ближайших соседей

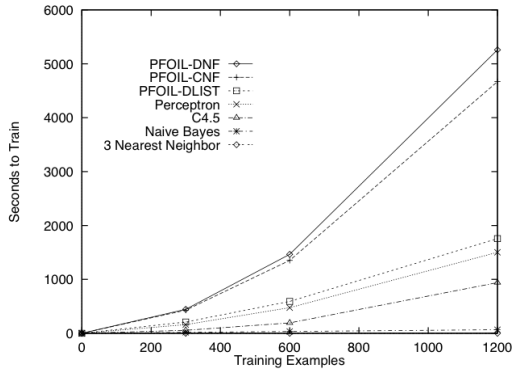


Рис. 2: Зависимость времени затраченного на обучение алгоритмов от размера обучающей выборки [38]. Алгоритмы: PFOIL-DLIST, PROIL-DNF, PFOIL-CNF, C4.5, Naive Bayes — наивный баейсовский классификатор; Perceptron — персептрон; 3 Nearest Neighbor — метод 3-х ближайших соседей

На Рис. 3 представлена зависимость времени работы алгоритмов от размера обучающей выборки. Время работы алгоритмов дает другую картину: байесовский классификатор и перцептрон работают долго при максимальном размере обучающей выборки, в то время как остальные методы решают WSD задачу за постоянное время (Рис. 3).

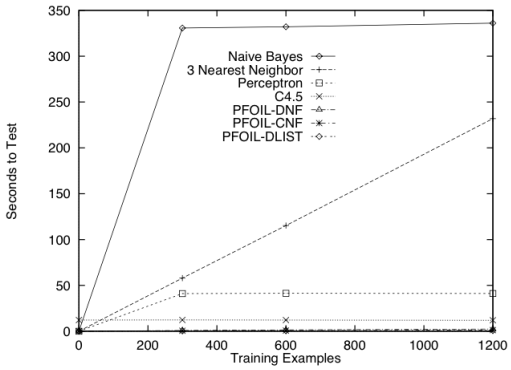


Рис. 3: Зависимость времени работы алгоритмов от размера обучающей выборки при определении значения слова *line* [38]. Алгоритмы: PFOIL-DLIST, PROIL-DNF, PFOIL-CNF, C4.5, Naive Bayes — наивный баейсовский классификатор; Perceptron — персептрон; 3 Nearest Neighbor — метод 3-х ближайших соседей

WSD-методы без учителя

Различение значений слов на основе векторов свойств, расширенных словарными толкованиями

А. М. Спиркова

Амрута Пурандаре и Тед Педерсен в 2004 году разработали «Алгоритм различения значений на основе контекстных векторов» (*Context vector sense discrimination*) [45]. В этом алгоритме (1) берется набор примеров употреблений исследуемого слова, (2) выполняется кластеризация этих примеров так, чтобы близкие по значению или связанные каким-либо образом слова объединились в одну группу [45].

Word sense discrimination — это задача группировки нескольких употреблений данного слова в кластеры, где каждому класте-

ру соответствует определенное значение целевого слова. Подходы к решению этой проблемы основываются на дистрибутивной гипотезе, которая говорит о том, что: лингвистические единицы, встречающиеся в схожих контекстах, имеют близкие значения. Следует различать понятия *различение значений слов* и *разрешение лексической многозначности*. При *различении значений слов* нет никаких предопределенных значений слова, присоединенных к кластерам; здесь, скорее, слова, употребляющиеся в схожих контекстах, группируются в кластеры (значения).

При решении задачи *различения значений* используются контекстные вектора: если целевое слово встречается в тестовых данных, то контекст этого слова представляется в виде вектора контекста. *Вектор контекста* — это средний вектор по векторам свойств каждого из слов контекста. *Вектор свойств* содержит информацию о совместной встречаемости данного слова с другими словами, этот вектор строится по данным корпуса текстов на этапе обучения.

Метод различения значений Пурандаре и Педерсена [45] предназначен для работы при недостаточном объеме текстовых данных, при этом вектор свойств расширяется данными, извлеченными из толкований словарей. Этот метод группирует в кластеры близкие по значению употребления целевого слова.

Построение матрицы встречаемости слов. Первоначально строится матрица совместной встречаемости слов по данным обучающего корпуса (были использованы тексты Wall Street Journal и Британского национального корпуса).

Вектор свойств (строка матрицы) содержит информацию о совместной встречаемости данного слова с другими. Было решено в [45], что слова «встречаются», если они находятся в тексте на расстоянии не более пяти словопозиций (то есть между ними находится не более трех слов).

Обработка матрицы. После создания матрицы выполняется разделение тестовых данных, то есть группировка примеров употреблений (фраз) с целевым словом. Каждому слову в примере употребления в тестовых данных соответствует вектор свойств из матрицы встречаемости. Средний вектор свойств по всем словам соответствует вектору контекста. Таким образом, набор тестовых данных, включающих употребление исследуемого слова, преобразуется в набор контекстных векторов, каждый из которых соответствует одному из употреблений целевого слова.

Различение значений происходит путем кластеризации контекстных векторов с помощью разделяющего (partitional) или иерархического «сверху вниз» (agglomerative) алгоритма кластеризации [31], [30], [57]. Получающиеся кластеры составлены из употреблений близких по значению фраз, и каждый кластер соответствует отдельному значению целевого слова.

Векторы свойств, расширенные текстами толкований из словаря. Векторы свойств, полученные по небольшому корпусу текстов, имеют очень малую размерность (несколько сотен), что не позволяет полностью описать закономерности совместной встречаемости слов. Для решения этой проблемы векторы свойств слов расширяются содержательными словами (content words), извлеченными из словарных толкований разных значений данного слова. В Табл. 4 представлены примеры толкований и содержательные слова для восьми значений слова «история» из Русского Викисловаря.

Таблица 4: Словарные толкования (и содержательные слова) по данным статьи «история» из Русского Викисловаря. Серым цветом и курсивом выделены те слова, которые уже были в векторе слов, черным – новые слова из толкований, которыми будет расширен вектор свойств

№	Текст значения	Содержательные слова
1	закономерное, последовательное развитие, изменение действительности	*развитие,* изменение
2	наука, изучающая факты, тенденции и закономерности развития человеческого общества	*наука,* факт, тенденция, закономерность
3	наука, изучающая ход развития, последовательные изменения какой-либо области природы или культуры	*наука, развитие,* изменение
4	последовательный ход развития, изменения чего-либо, совокупность фактов о развитии какого-либо явления	*развитие,* изменение, факт
5	отдаленное время с его событиями, происшествиями; прошлое	время, событие, происшествие
6	эпическое повествование, рассказ	повествование, *рассказ*
7	смешная или неожиданная ситуация, происшествие, случай	ситуация, случай, происшествие
8	скандал, неприятность	скандал, неприятность

Предположим, например, что вектор свойств (столбец в матрице встречаемости) для слова *история* имеет непустые значения в строках, соответствующих словам: *книга, мир, наука, образование, развитие, рассказ.*

В Русском Викисловаре различные значения слова *история* (Табл. 4) включают содержательные слова: *время, закономерность, изменение, наука, неприятность, повествование, происшествие,* развитие, рассказ, *ситуация, скандал, случай, событие, тенденция, факт.* Таким образом, вектор свойств, соответствующий слову «история», будет расширен новыми (отсутствующими ранее) словами из словаря: *время, закономерность, изменение, неприятность, повествование, происшествие, ситуация, скандал, случай, событие, тенденция, факт.*

В итоге, вектор свойств будет включать слова: *время, закономерность, изменение,* книга, мир, наука, *неприятность,* образование, *повествование, происшествие,* развитие, рассказ, *ситуация, скандал, случай, тенденция, факт.*

Для оценки результатов тестовым примерам употребления присваивали вручную теги значений. Кластеру присваивалось то значение, примеров употребления которого в нем было больше всего.

Авторами было проведено 75 экспериментов с использованием 72 слов из корпуса SENSEVAL-2 и со словами *line, hard* и *serve.*

В тестовых данных SENSEVAL-2 примеры употреблений включали 2-3 предложения. Для каждого слова было дано от 50 до 200 примеров употреблений в тестовых и тренировочных данных. Для этих слов известно много (порядка 8-12) значений. Малое число примеров при большем числе значений привело к тому, что для некоторых значений оказалось мало примеров употреблений. 43 из 72 слов SENSEVAL-2 показали улучшение F-меры и полноты (recall) при расширении вектора свойств текстами толкований словаря. Однако для 29 слов F-мера стала хуже, что, возможно, говорит о трудностях и несовершенстве метода. Для окончательной оценки необходима большая экспериментальная база: не десятки слов, а десятки и сотни тысяч.

Данный метод может быть полезен при различении значений слов без учителя при небольшом количестве обучающих данных.

Автоматический поиск и кластеризация похожих слов

Д. С. Шорец

В работе [15] представлена методология автоматического создания тезауруса, основанная на анализе корпуса текста и вычислении сходства слов, близости их значений. Значение незнакомого слова часто можно определить по контексту [20]. Рассмотрим, например, следующий текст:

(1) *Бутылка Tezgüino стоит на столе. Всем нравится Tezgüino. Tezgüino может привести к опьянению. Мы делаем Tezgüino из зерна.*

Из этого контекста можно предположить, что *Tezgüino* — это алкогольный напиток, приготовленный из зерна.

Задача поиска похожих слов *(similar words)* является первым шагом в определении значения слова. Тогда при обработке корпуса, включающего предложение (1), результатом должно быть определение близости значения слова *Tezgüino* к словам *пиво, вино, водка.*

Методология автоматического создания тезауруса. Для вычисления сходства между словами в работе [15] использован парсер [16], извлекающий тройки из текста. Тройки зависимостей (от англ. *dependency triple*, далее просто *тройки*) состоят из двух слов и грамматического отношения между ними. Символ $||w, r, w'||$ означает частоту в корпусе тройки (w, r, w'), где w, w' — это слова в нормальной форме, r — синтаксическое отношение. Произвольное слово или отношение обозначается символом-джокером «*». Например, *||cook, obj, *||* означает число троек со словом *cook* и отношением *obj*.

Например из предложения *«У меня есть коричневая собака»* будут извлечены следующие тройки:

$$||\textit{коричневый, прил\_сущ, собака}||$$
$$||\textit{есть, гл\_сущ, собака}||$$

Определим следующие моменты:

1. *Описание слова* w — это частоты всех троек (w, *, *) в корпусе, то есть всех троек, включающих w. Описание слова w является вектором.

2. *«Пересечение» двух слов* — это тройки, представленные в описании обоих слов; это пересечение векторов.

Сходство между двумя объектами вычисляется как количество информации в «пересечении» двух объектов (2), деленное на количество информации в описании двух объектов (1), далее обозначено как функция $sim(w_1, w_2)$ [17].

Предположив, что частоты троек не зависят друг от друга, получаем, что информация, представленная в описание слова w, равна сумме информации по каждой из уникальных троек в описании слова w.

Для измерения информации в утверждении $||w, r, w'||=c$ выполним следующее:

1. измерим количество информации в утверждении, что произвольная тройка, извлеченная из текста, будет наша тройка *(w, r, w')* при условии, что значение $||w, r, w'||$ — не известно;

2. измерим то же при условии, что значение $||w, r, w'||$ — известно;

3. разница этих двух количеств является ответом.

Вероятность встретить в тексте тройку *(w, r, w')* можно рассматривать как одновременное возникновение трех событий:

A: случайно выбранное слово - это w;
B: случайно выбранное отношение- это r;
C: случайно выбранное слово - это w';

1. Когда значение $||w, r, w'||$ неизвестно, то предполагаем, что **A** и **C** являются условно независимыми при наличии события **B**. Вероятность наступления сразу трех этих событий составляет $P_{MLE}(B)P_{MLE}(A|B)P_{MLE}(C|B)$, где P_{MLE} — это оценка максимального правдоподобия распределения вероятностей (*maximum likelihood estimation*)
$$P_{MLE}(B) = \frac{||*, r, *||}{||*, *, *||}$$
$$P_{MLE}(A|B) = \frac{||w, r, *||}{||*, r, *||}$$
$$P_{MLE}(C|B) = \frac{||*, r, w'||}{||*, r, *||}$$

2. Когда значение $||w, r, w'||$ известно, можно сразу получить $P_{MLE}(A, B, C)$:
$$P_{MLE}(A, B, C) = \frac{||w, r, w'||}{||*, *, *||}$$

3. Пусть ***I(w,r, w')*** обозначает количество информации, содержащейся в утверждении $||w, r, w'||=c$. Можно вычислить это значение так:

$$I(w, r, w') =$$
$$= -log(P_{MLE}(B)P_{MLE}(A|B)P_{MLE}(C|B)) -$$
$$-(-log(P_{MLE}(A, B, C)) =$$
$$= log\frac{||w, r, w'|| \times ||*, r, *||}{||w, r, *|| \times ||*, r, w'||}.$$

Отметим, что значение *I(w,r,w')* равно количеству взаимной информации (*mutual information*) между w и w' [18].

Пусть $T(w)$ — это множество пар *(r, w')*, при которых $log\frac{||w, r, w'|| \times ||*, r, *||}{||w, r, *|| \times ||*, r, w'||}$ имеет положительное значение. Определим значение сходства (похожести) двух слов *w1* и *w2* с помощью формулы:

$$sim(w_1, w_2) =$$
$$\frac{\sum_{(r,w) \in T(w_1) \cap T(w_2)} (I(w_1, r, w) + I(w_2, r, w))}{\sum_{(r,w) \in T(w_1)} I(w_1, r, w) + \sum_{(r,w) \in T(w_2)} I(w_2, r, w)}.$$

Практическая реализация метода. Был обработан корпус, включающий 64 млн. слов. Из него было извлечено 56,6 миллионов троек, включающих 8,7 миллиона уникальных троек.

Сам корпус был разбит на классы по частям речи. Исследовалось попарно сходство между всеми глаголами, всеми существительными, всеми прилагательными/наречиями по формуле $sim(w_1, w_2)$. Для каждого слова был построен аналог словарной статьи в тезаурусе, включающий упорядоченный набор 200 наиболее похожих слов. Статья в тезаурусе для слова w имела следующий формат:

$$w(pos) : w_1, s_1, w_2, s_2, \ldots, w_N, s_N$$

где pos — это часть речи, w_i — это похожее слово, s_i — это значение сходства между w и w_i, слова упорядочены по убыванию значения сходства.

Два слова являются *парой взаимных ближайших соседей (RNN от respective nearest neighbors)*, если они являются наиболее похожими словами друг для друга (первыми в списке из двухсот слов). С помощью программы удалось получить 543 пары RNN существительных, 212 пар RNN глаголов, 382 пары RNN прилагательных/наречий в созданном автоматически тезаурусе. В Табл. 5 представлен список каждого 10-го RNN для глаголов.

Таблица 5: Список пар взаимных ближайших соседей (RNN) глаголов

Ранг	**RNN**	**Значение сходства**
1	*fall rise*	0,67
11	*injure kill*	0,38
21	*concern worry*	0,34
31	*convict sentence*	0,29
41	*limit restrict*	0,27
51	*narrow widen*	0,26
61	*attract draw*	0,24
71	*discourage encourage*	0,23
81	*hit strike*	0,22
91	*distregard ignore*	0,21
101	*overstate understate*	0,20
111	*affirm reaffirm*	0,18
121	*inform notify*	0,17
131	*differ vary*	0,16
141	*scream yell*	0,15
151	*laugh smile*	0,143
161	*compete cope*	0,136
171	*add whisk*	0,130
181	*blossom mature*	0,12
191	*smell taste*	0,11
201	*bark howl*	0,10
211	*black white*	0,07

Разрешение многозначности в биомедицинских текстах с помощью методов кластеризации без учителя

Е. А. Ярышкина

В статье [48] изучаются уже существующие методы кластеризации без учителя и их эффективность для решения лексической многозначности при обработке текстов по биомедицине. Решение проблем лексической многозначности в данной области включает в себя не только традиционные задачи присвоения ранее определенных смысловых значений для терминов, но так же и обнаружения новых значений для них, еще не включенных в данную онтологию.

Авторы описали методологию способа разрешения лексической многозначности без учителя, учитываемые лексические признаки и наборы экспериментальных данных. В каче-

стве оценки эффективности алгоритмов кластеризации текста была предложена F-мера.

Подход для решения поставленной задачи — это разделение контекстов (фрагментов текста), содержащих определенное целевое слово, на кластеры, где каждый кластер представляет собой различные значения целевого слова. Каждый кластер состоит из близких по значению контекстов. Задача решается в предположении, что используемое целевое слово в аналогичном контексте будет иметь один и тот же или очень похожий смысл.

Процесс кластеризации продолжается до тех пор, пока не будет найдено предварительно заданное число кластеров. В данной статье выбор шести кластеров основан на том факте, что это больше, чем максимальное число возможных значений любого английского слова, наблюдаемое среди данных (большинство слов имеют два-три значения). Нормализация текста не выполняется.

Данные в этом исследовании состоят из ряда контекстов, которые включают данное целевое слово, где у каждого целевого слова вручную отмечено — какое значение из словаря было использовано в этом контексте. Контекст — это единственный источник информации о целевом слове. Цель исследования — преобразовать контекст в контекстные вектора первого и второго порядка [3]. Контекстные вектора содержат следующие «лексические свойства»: биграммы, совместную встречаемость и совместную встречаемость целевого слова. Биграммами являются как двухсловные словосочетания, так и любые два слова, расположенные рядом в некотором тексте. Для лингвистических исследований могут быть полезны только упорядоченные наборы биграмм [1].

Экспериментальные данные — это набор NLM WSD [54] (NLM — национальная библиотека медицины США), в котором значения слов взяты из UMLS (единая система медицинской терминологии). UMLS имеет три базы знаний:

- Метатезаурус включает все термины из контролируемых словарей (SNOMED-CT, ICD и другие) и понятия, которые представляют собой кластера из терминов, описывающих один и тот же смысл.

- Семантическая сеть распределяет понятия на 134 категории и показывает отношения между ними. SPECIALIST-лексикон содержит семантическую информацию для терминов Метатезауруса.

- Medline — главная библиографическая база данных NLM, которая включает приблизительно 13 миллионов ссылок на журнальные статьи в области науки о жизни с уклоном в биомедицинскую область.

Авторы успешно проверили по три конфигурации существующих методов (PB — Pedersen and Bruce [44], SC — Schütze [53]) и оценили эффективность использования SVD (сингулярное разложение матриц). Методы PB основаны на контекстных векторах первого порядка — признаки одновременного присутствия целевого слова или биграммы. Рассчитывается среднее расстояние между кластерами или применяется метод бисекций. PB методы подходят для работы с довольно большими наборами данных. Методы SC основаны на представлениях второго порядка — матрицы признаков одновременного присутствия или биграммы, где каждая строка и столбец — вектор признаков первого порядка данного слова. Так же рассчитывается среднее расстояние между кластерами или применяется метод бисекций. SC методы подходят для обработки небольших наборов данных.

Метод SC2 (признаки одновременного присутствия второго порядка, среднее расстояние между элементами кластера в пространстве подобия) с применением и без SVD показал лучшие результаты: всего 56 сравниваемых экземпляров, в 47 случаях метод SC2 показал наилучшие результаты, в 7 случаях результаты незначительно отличаются от других проверяемых методов.

Все эксперименты, указанные в исследовании, выполнялись с помощью пакета SenseClusters [52]. В ходе исследования было проведено два эксперимента для разных наборов данных. Маленький тренировочный набор — это набор NLM WSD, который включает 5000 экземпляров для 50 часто встречаемых неоднозначных терминов из Метатезауруса UMLS. Каждый неоднозначный термин имеет по 100 экземпляров с указанным вручную значением. У 21 термина максимальное число экземпляров находится в пределах от 45 до 79 экземпляров. У 29 терминов число экземпляров от 80 до 100 для конкретного значения. Стоит отметить, что каждый термин имеет категорию «ни одно из вышеупомянутых», которая охватывает все оставшиеся значения, не соответствующие доступным в UMLS. Большой тренировочный набор является реконструкцией «1999 Medline», который был разработан Weeber [56]. Были определены все формы из набора NLM WSD и сопоставле-

ны с тезисами «1999 Medline». Для создания тренировочного набора экземпляров использовались только те тезисы из «1999 Medline», которым было найдено соответствие в наборе NLM WSD.

Использование целиком текста аннотации статьи в качестве контекста приводит к лучшим результатам, чем использование отдельных предложений. С одной стороны, большой объем контекста, представленный аннотацией, дает богатую коллекцию признаков, с другой стороны, в коллекции WSD представлено небольшое число контекстов.

Выявление значений слов из текста

Д. Ю. Янкевич

В статье [42] представлен алгоритм автоматического обнаружения значений слов в тексте, названный *кластеризация посредством комитетов* (Clustering By Committee (CBC)). Также авторы предлагают методологию оценки для автоматического измерения точности и полноты найденных значений.

Алгоритм первоначально находит множество небольших кластеров, называемых комитетами, каждый из которых представляет собой одно из значений определяемого слова. Центр тяжести членов комитета (мера связности с определяемым словом) используется в качестве вектора признаков кластера.

CBC состоит из трех этапов.

На этапе I для каждого элемента (слова) вычисляются k наиболее похожих слов. Сначала весь список относящихся к слову значений сортируется по убыванию значений связи согласно формуле точечной взаимной информации (pointwise mutual information (PMI) [35]), а затем, с помощью иерархического кластерного анализа по *методу средней связи* [2], вычисляется сходство между всеми элементами кластера попарно. Значение функции PMI [35] между предполагаемым значением слова (контекстом) и элементом (словом) вычисляется следующим образом: пусть x — это рассматриваемый элемент, а y — контекст. *Точечная взаимная информация* между x и y определена как:

$$pmi(x;y) \equiv log\frac{p(x,y)}{p(x)p(y)} =$$

$$= log\frac{p(x|y)}{p(x)} = log\frac{p(y|x)}{p(y)}.$$

При кластеризации посредством метода средних связей *(average-link clustering)* вычисляется среднее сходство между данным объектом и всеми объектами в кластере, а затем, если найденное *среднее значение сходства* достигает или превосходит некоторый заданный пороговый уровень сходства, объект присоединяется к этому кластеру [2]. Сложность этого алгоритма $O(n^2 * log(n))$, где n — число кластеризуемых элементов [31].

В этапе I оценка (PMI) означает предпочтение большим и тесно связанным кластерам.

На II этапе строится набор небольших кластеров, где элементы каждого кластера образуют комитет. В ходе работы «Алгоритма 1» формируется как можно больше комитетов при условии, что каждый вновь созданный комитет не слишком похож на любой из уже существующих комитетов. Если условие нарушается, комитет просто отбрасывается. Алгоритм описан ниже подробно.

Data: Список элементов E, которые будут сгруппированы, база данных сходства S, из фазы I, пороги $\theta 1$ и $\theta 2$ (с помощью порога $\theta 1$ сохраняются только те кластеры, которые имеют значения, отличные от ранее обнаруженных, порог $\theta 2$ позволяет обнаружить элементы, не принадлежащие ни одному из кластеров)
Result: C — список комитетов
Step 1:
foreach $\in E$ **do**
1. Кластер k наиболее «близких» (похожих) элементов e из S с помощью метода средней связи
2. Для каждого обнаруженного кластера c вычислить следующую оценку: $val = |C| * avgsim(c)$, где $|C|$ — количество элементов c и $avgsim(c)$ — усредненное сходство между всеми парами элементов кластера c.
3. Записать кластер с наивысшей оценкой в список L

end
Step 2:
Сортировка кластеров в списке L в порядке убывания их оценок val
Step 3:
$C = \varnothing$ // Пусть перечень комитетов, изначально пустой
foreach $c \in L$ **do**
// в отсортированном по убыванию порядке:
1. Вычислить центр тяжести, усредняя поэлементно значение векторов, и вычислить вектор PMI центроида (точно так же, как мы делали для отдельных элементов)
2. Если схожесть c и центроида каждого комитета, ранее добавленного к C, ниже порогового $\theta 1$, то следует добавить c в C

end
Step 4:
if $C = \varnothing$ **then**
  return C
end
$R = \varnothing$ // R — это множество остатков, то есть элементов, не охваченных ни одним из кластеров
foreach $e \in E$ **do**
  foreach $c \in C$ **do**
    if $sim(e, c) < \theta 2$ // $sim(e, c)$ — *схожесть e каждому комитету из* C **then**
      $R+ = e$ // то следует добавить e в список остатков R
    end
  end
end
if $R = \varnothing$ **then**
  return C **else**
    return $C \cup Algorithm1(R, S, \theta 1, \theta 2)$
  end
end

Algorithm 1: II фаза кластеризации посредством комитетов (CBC)

В результате второго этапа построения CBC остаются кластеры, связанные более тесно (имеющие большее значение val, см. step 1 и 3 алгоритма 1).

На заключительном III этапе работы алгоритма каждый элемент e присваивается наиболее подходящему кластеру следующим образом (при этом центроид членов комитета используется в качестве вектора характеристик кластера):

Result: Итоговые кластеры с максимальным значением связи (val) между словами
Пусть список кластеров (изначально пустых)
Пусть S это топ-200 кластеров, схожих с e
while S *не пуст* **do**
  пусть $c \in S$ наиболее близкий кластер к e
  if *сходство* $(e, c) < \sigma$ **then**
    конец цикла
  end
  if c *не схож ни с одним кластером в* C **then**
    присвоить e к c
    удалить из e его характеристики, которые перекрываются с характеристиками c
  end
  удалить c из S
end

Algorithm 2: Присвоение элементов кластерам

На III этапе кластер сохраняется только в случае, если его сходство со всеми ранее полученными кластерами ниже установленного порогового значения.

Согласно дистрибутивной гипотезе (Distributional Hypothesis) [25] слова, употребляемые в сходных контекстах, близки по смыслу. Алгоритм *Кластеризации посредством комитетов* [42] разрешает лексическую многозначность, группируя слова согласно сходству их контекстов. Каждому полученному кластеру соответствует одно из значений слова.

Таблица 6: Для построения кластеров использовались данные словарных статей Викисловаря: «одежда», «рукав» и «сердце»

Рукав		
Nq34	0.39	деталь, манжета, полотно
Nq137	0.20	протока, русло, отмель, поток
Nq217	0.18	шланг, труба, огнетушитель
Сердце		
Nq72	0.27	орган, костный мозг, почка
Nq866	0.17	душа, рассудок, сознание
Одежда		
Nq215	0.41	мануфактура, юбка, брюки
Nq235	0.20	покрытие, оболочка, дорога

В Табл. 6 каждая запись показывает кластеры, которым принадлежит заглавное слово. Имена для кластеров Nq34, Nq137, ... генерируются автоматически. После каждого имени кластера находится число, обозначающее сходство между кластером и заглавным словом (т.е. рукав, сердце и одежда). Далее перечисляются четыре слова, наиболее близкие центроиду кластера. Каждый кластер соответствует одному значению заглавного слова. Например, Nq34 соответствует значению «деталь одежды», а Nq137 соответствует значению «ответвление русла реки».

Сравнение с алгоритмом UNICON. CBC является разновидностью алгоритма UNICON [34], который также строит центроид кластера, используя небольшой набор похожих элементов.

Одним из основных различий между UNICON и CBC является то, что UNICON гарантирует, что различные комитеты не имеют одинаковых элементов, тем не менее, центры тяжести двух комитетов по-прежнему могут быть очень близкими (похожими). В UNICON'е эта проблема решается объединением таких кластеров. В отличие от этого, на II этапе CBC создаются только те комитеты, центры тяжести которых отличны от всех ранее созданных комитетов.

Есть разница и на III этапе CBC. Алгоритм UNICON плохо работает со словами, которые имеют несколько широко используемых (доминирующих) значений. Например, пусть значение «отмычка» является более употребимым для слова «ключ», чем значение «водный источник». Приведем смесь слов-синонимов к разным значениям слова «ключ»: пневмоключ, электроключ, родник, родничок, источник, криница, гидроключ, ключик, тангента, треншальтер, тумблер, знак, контролька, отпирка, виброплекс, шифр. В этом списке 10 значений относятся к значению «отмычка()», 4 к «водный источник()» и 2 к значению «знак()». По этому списку алгоритмом UNICON будут сгенерированы кластеры «отмычка», «шифрование», «происхождение»,«криптография», «кнопка», «водный источник»,«переключатель», «намек». Сходство между словом и полученными кластерами является очень низким, к тому же есть кластеры, содержащие одинаковые слова. С другой стороны, CBC удаляет «пересекающиеся» (общие для двух кластеров) характери-

стики после того, как присвоит значение кластеру (допустим характеристики, относящиеся к значению «отмычка» слова «ключ» из вектора характеристик «отмычка»). В результате, сходство между кластером «водный источник» {родник, родничок, источник, криница} и пересмотренным вектором характеристик кластера «водный источник» становится намного выше. Что в свою очередь приводит к тому, что кластеры становятся гораздо точнее.

WSD-методы, основанные на знаниях

Использование лексических цепочек для реферирования текстов

А. В. Пилинович

В статье [8] с целью реферирования текста строится модель в виде лексических цепочек. Реферирование включает четыре этапа: оригинальный текст делится на блоки (сегменты), строятся лексические цепочки, определяются сильные цепочки, извлекаются важные предложения.

Реферирование — это процесс сжатия исходного текста в более компактный, при сохранении информативности текста. Реферирование выполняется для решения разных задач - от обзорного анализа текстов какой-либо научной области до быстрого выделения главных тем текста. Создание качественной информативной аннотации произвольного текста является сложной задачей, требующей полного понимания текста. Легче создавать приблизительные, указательные аннотации (indicative summaries), позволяющие принять решение — стоит ли читать текст. В работе [8] описан метод создания указательных аннотаций по произвольным текстам.

Интуитивное понятие *cohesion* (связность, склеивание, слияние), введенное в [24], указывает на объединение разных частей (фрагментов) текста в одно целое, в то, что имеет значение, смысл. Одним из видов связности является *лексическая связность* (lexical cohesion) [29]. Лексическая связность формируется с помощью семантически связанных слов. Halliday и Hasan [24] выделили два способа формирования лексической связности: (1) с помощью категории повторений и (2) категории словосочетаний.

1. *Лексическая связность повторений* (reiteration category) достигается повтором слов, использованием синонимов и гипонимов.

2. *Лексическая связность словосочетаний* (collocation category) определена для слов, которые часто употребляются вместе, то есть встречаются в одних и тех же контекстах.

Слова и фразы, между которыми существует лексическая связность, формируют лексическую цепочку (lexical chains) [29]. Метод лексических цепочек, предложенный Барзилей и Эльхадад [8], основан на анализе совместной встречаемости слов и лексических связей между словами.

Алгоритм построения цепочек. Достоинство лексических цепочек в том, что их легко распознать и построить. Первая вычислительная модель для лексических цепочек была представлена в работе Морриса и Хирста [29]. Цепочки создавались путем взятия нового слова из текста и поиска родственной (связанной) цепочки для слова в соответствии с критериями родства. Недостатком подхода в [29] было то, что в одну цепочку могло входить слово с разными значениями (для многозначных слов). Таким образом, выбор подходящей цепочки для слова эквивалентен решению WSD-задачи.

Метод построения лексических цепочек включает шаги:

1. Выбирается набор слов-кандидатов (существительные и составные существительные). Это кандидаты на включение в цепочки.

2. Строится список всех значений для каждого слова-кандидата (по данным словаря).

3. Для каждого значения каждого слова-кандидата находится (вычисляется) отношение (расстояние) до каждого слова во всех уже построенных цепочках (слово в цепочке имеет строго определённое значение, задаваемые другими словами в той же цепочке). Между двумя словами есть отношение (будет указана связь в цепочке), если мало расстояние между этими словами в тексте (text distance) или между значениями этих слов существует путь в тезаурусе WordNet. Выделяют три вида отношений ([19], стр. 36):

   (a) *Extra-strong* отношение существует для слов, повторяющихся в тексте.

Повтор может быть на любом расстоянии от первого употребления слова.

(б) *Strong* отношение определено между словами, связанными отношением в WordNet. Два таких слова должны находиться в окне не более семи предложений.

(в) *Medium-strong* отношение указывается для слов, синсеты которых находятся на расстоянии больше одного в WordNet (но есть ещё и дополнительные ограничения на путь между синсетами). Слова в тексте должны находиться в пределах трех предложений.

4. Слово-кандидат добавляется в цепочки, со словами которых найдена связь. Смысловая неоднозначность устраняется, то есть в цепочку добавляется не просто слово, а его конкретное значение (благодаря выбору значения в словаре на шаге 2).

Для выбора приоритетной цепочки, (для вставки слова-кандидата) отношения упорядочены так: extra-strong, strong, medium-strong. В работе Хирста и Ст-Онж [28] предложен жадный алгоритм выбора цепочек. При этом слово-кандидат попадает ровно в одну цепочку и после этого выбор уже не может быть изменён, даже если последующий текст покажет ошибочность первоначального решения. В работе Барзилей и Эльхадад [8] предложена более сложная схема выбора «подходящего значения», требующая рассмотрения всех возможных цепочек. Таким образом, будут сформированы цепочки с учетом всех возможных значений слов с последующим выбором наилучшей цепочки. Эта более сложная схема и рассматривается далее.

Для иллюстрации метода приводится пример на отрывке текста, представленном ниже, и посмотрим, какое значение будет выбрано для слова *machine*. Во-первых, для слова *Mr.* создается узел [лексема "Mr. значение {mister, Mr.}]. Следующим по тексту существительным, представленным в тезаурусе WordNet, будет слово *person*, у него есть два значения: *"human being"* ($person_1$) и *"grammatical category of pronouns and verb forms"* ($person_2$). Наличие двух значений у слова *person* разбивает пространство цепочек на два множества интерпретаций: в первой интерпретации используется значение $person_1$, во второй — $person_2$, (Рис. 4).

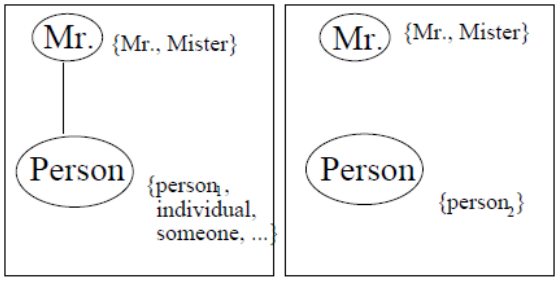


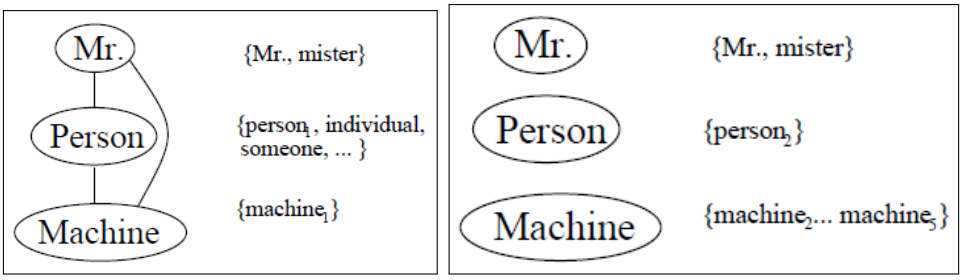
Рис. 4: Шаг 1, интерпретация 1 и 2

Mr. *Kenny is the* ***person*** *that invented an anesthetic* ***machine*** *which uses* ***micro-computers*** *to control the rate at which an anesthetic is pumped into the blood. Such* ***machines*** *are nothing new. But his* ***device*** *uses two* ***micro-computers*** *to achieve much closer monitoring of the* ***pump*** *feeding the anesthetic into the patient.*

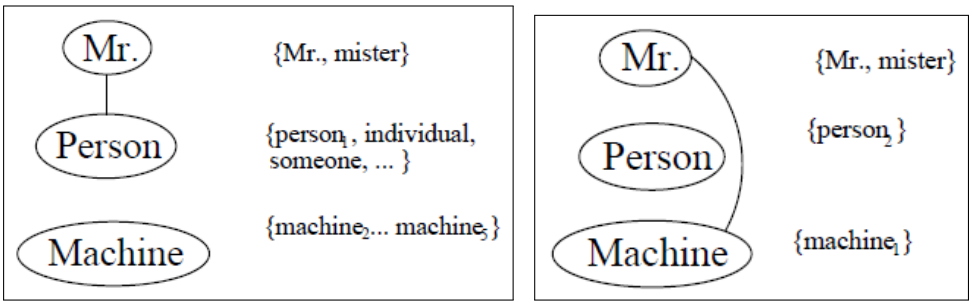
Компонентой в [8] называют список взаимоисключающих интерпретаций. Именно посредством компонент выбор одного из значений слов ведёт к выбору соответствующей интерпретации, а, следовательно, к невозможности других интерпретаций из этой компоненты. Интерпретации 1 и 2 на Рис. 4 являются компонентой.

Следующее слово *anesthetic* не связано со словами из первой компоненты, поэтому для него создается компонента с одним значений (то есть новая компонента содержит ровно одну интерпретацию).

Следующее слово *machine* имеет 5 значений: от $machine_1$ до $machine_2$. В первом значении $machine_1$ [лексема "machine значение {an efficient person}] слово связано со значениями слов *person* и *Mr.*, поэтому слово *machine* вставляется в первую компоненту. После этой вставки изображение первой компоненты становится таким, как показано на Рис. 6. Если продолжить этот процесс и вставить слова *micro-computer, device* и *pump,* то количество альтернативных вариантов значительно увеличивается. Самые сильные интерпретации представлены на Рис. 7. При условии, что текст связный, лучшей интерпретацией считается та, которая имеет больше всего связей. В данном случае в конце шага 3 выбрана другая интерпретация $machine_4$ [лексема "machine значение {*any mechanical or electrical device that performs or assists in the performance*}], что верно отражает значение слова *machine* в этом контексте.

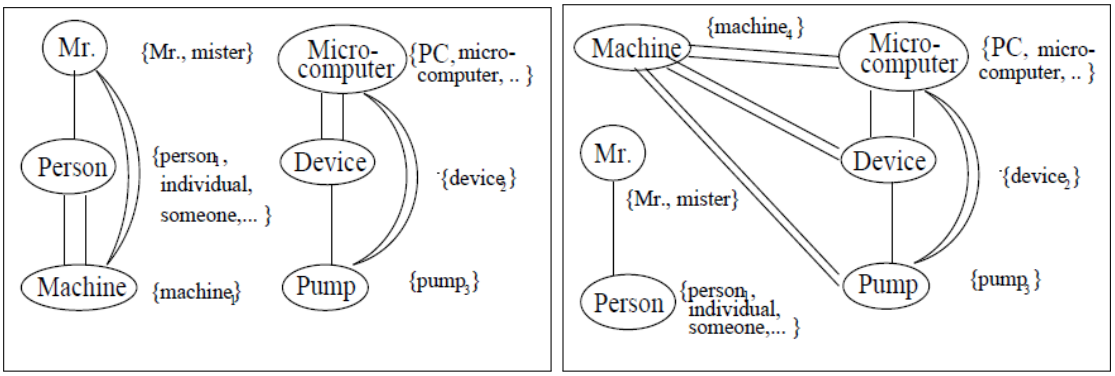


(a) Интерпретация 1 (b) Интерпретация 2



(a) Интерпретация 3 (b) Интерпретация 4

Рис. 6: Четыре интерпретации на втором шаге



(a) Интерпретация 1 (b) Интерпретация 2

Рис. 7: Две самые сильные интерпретации, полученные на третьем шаге

Оценка интерпретации определяется как сумма оценок ее цепочек. Оценка цепочки определяется количеством и весом отношений между участниками цепочки. В эксперименте авторы зафиксировали следующий вес: повторения и синонимы — 10, антонимы — 7, гиперонимы и гипонимы — 4. Описанный алгоритм вычисляет все возможные интерпретации, не допуская противоречий между ними. Когда число возможных интерпретаций превышает определенный порог, слабые интерпретации удаляются, это необходимо для предотвращения экспоненциального роста использования памяти.

Объединение цепочек из разных сегментов. Текст предварительно разбивается на сегменты (несколько предложений или абзац). Пример выше (*Mr. Kenny...*) соответствует одному сегменту. Цепочки строятся для каждого сегмента на основе найденных отношений между словами (extra-strong, strong, medium-strong). На следующем этапе объединяются цепочки из разных сегментов, но для объединения нужно, чтобы выполнялось еще более жесткое условие: две цепочки объединяются, если они содержат одно и то же слово в одном и том же значении. Поскольку есть прямая связь между цепочками и смысловы-

ми блоками текста, постольку с помощью лексических цепочек можно решать и обратную задачу — разбиение текста на сегменты [8].

Вычисление оценок цепочек. Для того чтобы использовать лексические цепочки для построения аннотации, в первую очередь следует выявить сильнейшие цепочки среди всех тех, которые создаются описанным выше алгоритмом. Барзилей и Эльхадад в [8] предложили эмпирическую методику для оценки силы цепочки. Они разработали среду, чтобы вычислить и графически визуализировать лексические цепочки, чтобы оценить экспериментально, насколько хорошо идентифицируются (определяются) основные темы текстов. Авторы собрали данные из 30 текстов, выбранных случайным образом из популярных журналов (например, "The Economist", "Scientific American"). Для каждого текста вручную выполнили ранжирование цепочек по степени соответствия основным темам текста.

Из множества параметров, которые можно измерить (длина цепочки; объем текста, покрываемого цепочкой; плотность; диаметр слов цепочки в графе тезауруса; число повторений), Барзилей и Эльхадад [8] опытным путем нашли следующие показатели значимости цепочек для построение реферата:

- *Длина (Length):* число употреблений в тексте элементов цепочки.
- *Индекс однородности (HomogeneityIndex):* 1 — количество различных употреблений в тексте элементов цепочки, деленное на длину (*Length*).

Таким образом, значимость цепочек оценивается так:

$$Score(Chain) = Length \times HomogeneityIndex$$

При ранжировании цепочек в соответствии с этой оценкой, было найдено, что для построения реферата нужны цепочки, удовлетворяющие «критерию прочности (силы)»:

$$Score(Chain) > Average(Scores) + 2 \times StandardDeviation(Scores)$$

где *Average* — это средняя оценка по всем цепочкам, *StandardDeviation* — среднеквадратическое отклонение.

Извлечение важных предложений. После того как сильные цепочки отобраны, выполняется поиск соответствующих им предложений и извлечение этих предложений целиком из исходного текста.

Для каждой сильной цепочки на основе разработанных эвристик выбирается ровно одно предложение для включения в текст реферата:

Эвристика 1: Для каждой цепочки для включения в реферат выбрать то предложение, которое содержит первое появление члена цепочки в тексте.

Эвристика 2: Для каждой цепочки для включения в реферат выбрать предложение, которое содержит первое появление *показательного* элемента цепочки в тексте. *Показательные* слова (representative words), служащие представителями цепочки, — это такие слова цепочки, которые встречаются в цепочке не реже, чем в среднем по всем словам цепочки.

Эвристика 3: Для каждой цепи найти блок текста, где есть высокая концентрация цепочки (то есть много употреблений элементов из этой цепочка). Извлечь предложение с первого появления цепочки в этом блоке. Концентрация вычисляется как число появлений членов цепи в сегменте, разделенное на количество существительных в сегменте. Цепочка имеет высокую концентрацию, если ее концентрация является максимальной из всех цепочек. Кластер представляет собой группу последовательных сегментов, таких, что каждый сегмент содержит какие-либо элементы цепочки.

Эксперименты в [8] показали значительное преимущество алгоритма на основе лексических цепочек (точность 47-61% и полнота 64-67%) по сравнению с программой Microsoft Summarizer, доступной в Word'97 (точность 32-33% и полнота 37-39%). Эти результаты указывают на большой потенциал лексических цепочек в задаче реферирования.

Построение сочетаемостных ограничений на основе байесовских сетей для разрешения многозначности

И. А. Сихонина

В статье [11] представлена байесовская модель, применяемая для разрешения лексической многозначности глаголов. Авторы рассматривают такое понятие, как сочетаемостные ограничения (selectional preferences). *Сочетаемостные ограничения* (далее SP) — это закономерности использования глагола относительно семантического класса его параметров (субъект, объект (прямое дополнение) и косвенное дополнение).

Модели автоматического построения SP важны сами по себе и имеют приложения в обработке естественного языка. Сочетаемостные ограничения глагола могут применяться для получения возможных значений неизвестного параметра при известных глаголах; например, из предложения *«Осенние хххх жужжали и бились на стекле»* легко определить, что "хххх" — мухи. При построении предложения SP позволяют отранжировать варианты и выбрать лучший среди них. Исследование SP могло бы помочь в понимании структуры ментального лексикона.

Системы обучения SP без учителя обычно комбинируют статистические подходы и подходы, основанные на знаниях. Компонент базы знаний (здесь WordNet [37]) — это обычно база данных, в которой слова сгруппированы в классы.

Статистический компонент состоит из пар предикат-аргумент, извлеченных из неразмеченного корпуса. В тривиальном алгоритме можно было бы получить список слов (прямых дополнений глагола), и для тех слов, которые есть в WordNet, вывести их семантические классы. В работе [11] семантическим классом называется *синсет* (от англ. ***syn**onym **set***, группа синонимов) тезауруса WordNet, то есть класс соответствует одному из значений слова. Таким образом, в тривиальном алгоритме на основе данных WordNet можно выбрать классы (значения слов), с которыми употребляются (встречаются в корпусе) глаголы.

Например, если в исходном корпусе текстов глагол *ползать* употребляется со словом *ящерица*, принадлежащим классу РЕПТИЛИИ, то в модели построения SP будет записано, что «глагол *ползать* употребляется со словами из класса РЕПТИЛИИ». Если слово *крокодил*, во-первых, также встречается в тексте с глаголом *ползать*, во-вторых, слово *крокодил* принадлежит сразу двум классам: РЕПТИЛИЯ и ВЕРТОЛЕТ, то из этого следует, что модель SP будет расширена информацией о том, что «глагол *ползать* употребляется со словами из классов и РЕПТИЛИЯ, и ВЕРТОЛЕТ».

В ранее разработанных моделях (Резник (1997) [46], Абни и Лайт (1999) [6]) было обнаружено, что главная трудность в таком тривиальном алгоритме — это наличие неоднозначных слов в обучающих данных. В тех же работах ([46], [6]) были предложены более сложные модели, в которых предполагается, что все значения многозначных слов появляются с одинаковой частотой.

Байесовские сети или байесовские сети доверия (БСД) состоят из множества переменных (вершин) и множества ориентированных ребер, соединяющих эти переменные. Такой сети соответствует ориентированный ациклический граф. Каждая переменная может принимать одно из конечного числа взаимоисключающих состояний. Пусть все переменные будут бинарного типа, то есть принимают одно из двух значений: истина или ложь. Любой переменной A с родителями $B1, \ldots, Bn$ соответствует таблица условных вероятностей (conditional probability table, далее CPT).

Например, построим SP для глагола *ползать* и сеть на Рис. 8 будет базой знаний.

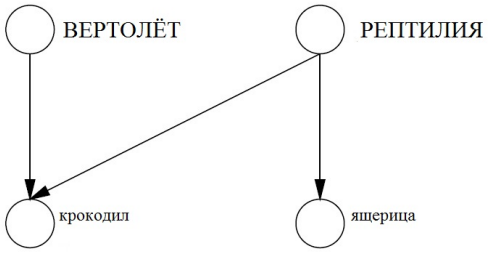


Рис. 8: Байесовская сеть для многозначного существительного *крокодил*

Глагол *ползать* употребляется со словами *крокодил* и *ящерица*. Переменные ВЕРТОЛЕТ и РЕПТИЛИЯ соответствуют более общим абстрактным значениям, переменные *крокодил* и *ящерица* являются более узкими, конкретными значениями. Переменная РЕПТИЛИЯ может принимать одно из двух значений, соответствующих словам *крокодил* и *ящерица*, именно эту задачу определения значения и нужно решить.

Таблица 7: Условные вероятности переменных *крокодил* и *ящерица* в зависимости от значений переменных ВЕРТОЛЕТ и РЕПТИЛИЯ, где (В, Р, к, я — это аббревиатуры слов ВЕРТОЛЕТ, РЕПТИЛИЯ, *крокодил* и *ящерица*)

	$(X = x \mid Y_1 = y_1, Y_2 = y_2)$			
	В,Р	В,¬Р	¬В,Р	¬В, ¬Р
к = *true*	0,99	0,99	0,99	0,01
к = *false*	0,01	0,01	0,01	0,99
я = *true*	0,99	0,99	0,01	0,01
я = *false*	0,01	0,01	0,99	0,99

При построении Табл. 7 условных вероятностей (CPT), учтем следующие предположения:

- вероятность, что выбираем какой-либо из концептов (ВЕРТОЛЕТ и РЕПТИЛИЯ) очень мала, то есть $P(B{=}true) = P(P{=}true) = 0{,}01$, следовательно, велика вероятность, что концепты не выбраны: $P(B{=}false) = P(P{=}false) = 0{,}99$;

- если какой-либо из концептов истинен (B, P), то «выпадает» слово *крокодил*;

- если концепт РЕПТИЛИЯ истинен, то растут шансы встретить слово *ящерица*;

Из Табл. 7 вероятности появления слов следует вывод, что использование разу двух значений слова *крокодил (рептилия и вертолет МИ-24)* маловероятно. Вероятность использования значения РЕПТИЛИЯ намного больше чем значения ВЕРТОЛЕТ. Таким образом гипотеза «вертолет» «отброшена» ("explaining away").

Байесовские сети для построения SP. Иерархия существительных в WordNet представлена в виде ориентированного ациклического графа. Синсет узла принимает значение «истина», если глагол «выбирает» существительное из набора синонимов. Априорные вероятности задаются на основе двух предположений: во-первых, маловероятно, что глагол будет употребляться только со словами какого-то конкретного синсета, и во-вторых, если глагол действительно употребляется только со словами из данного синсета (например, синсет ЕДА), тогда должно быть правомерным употребление этого глагола с гипонимами этого синсета (например, ФРУКТ).

Те же предположения (что для синсетов) верны и для употреблений слов с глаголами:

1. слово, вероятно, является аргументом глагола в том случае, если глагол употребляется с каким-либо из значений этого слова;

2. отсутствие связки глагол-синсет говорит о малой вероятности того, что слова этого синсета употребляются с глаголом.

Словам «вероятно» и «маловероятно» должны быть приписаны такие числа, сумма которых равна единице.

Находкой работы [11] является разъяснение стратегии "explaining away", то есть отбрасывание маловероятных значений слов при построении сочетаемостных ограничений. Такая стратегия является неотъемлемым свойством байесовских сетей и байесовского вывода, полезным свойством при разрешении лексической многозначности.

Заключение

Разрешение лексической многозначности - это задача выбора между разными значениями слов и словосочетаний в словаре в зависимости от контекста. Задача разрешения лексической многозначности является открытой проблемой, то есть крайне интересной и привлекательной с научной точки зрения.

В статье представлен краткий обзор методов и алгоритмов разрешения лексической многозначности. Во-первых, методы, основанные на машинном обучении. Во-вторых, методы, не использующие никаких размеченных корпусов для различения значений слов. В-третьих, методы, использующие внешние словарные источники информации (машиночитаемые словари, тезаурусы, онтологии).

Работа Старковой В.Г. поддержана грантом РГНФ (проект № 15-04-12029), работа Кириллова А.Н. и Чирковой Ю.В. поддержана грантом РГНФ (проект № 15-04-12006). Работа Крижановского А. А. выполнена при частичной финансовой поддержке Программы фундаментальных исследований Секции литературы и языка ОИФН РАН «Язык и информационные технологии» 2015-2017 (проект "Корпус вепсского языка: разработка и формирование морфологической базы электронного ресурса").

Литература

1. *А. Н. Аверин.* Разработка сервиса поиска биграмм // Труды международной конференции «Корпусная лингвистика–2006. СПб., С.Петерб. ун-та., 2006.

2. *Дж. О. Ким, Ч. У. Мьюллер, У. Р. Клекка.* Факторный, дискриминантный и кластерный анализ. «Финансы и статистика», Москва, Россия. 1989. Стр.172.

3. *А. С. Епрев.* Применение контекстных векторов в классификации текстовых документов. 2010. http://jre.cplire.ru/iso/oct10/1/text.html.

4. *Н. В. Лукашевич.* Тезаурусы в задачах информационного поиска. Издательство МГУ, 2011. 495 с.

5. *Д. Ю. Турдаков.* Методы и программные средства разрешения лексической многозначности терминов на основе сетей документов: диссертация ... кандидата физико-математических наук: 05.13.11. — Москва, 2010. — 138 с.

6. *S. Abney and M. Light.* Hiding a semantic hierarchy in a markov model. In Proceedings of the Workshop on Unsupervised Learning in Natural Language Processing, ACL. 1999.

7. *A. Azzini, C. da Costa Pereira, M. Dragoni, and A. G. B. Tettamanzi.* Evolving Neural Networks for

Word Sense Disambiguation // 8-th International conference on hybrid intelligent systems. Spain. Barcelona. pp. 332–337. DOI: 10.1109/HIS.2008.88

8. *R. Barzilay and M. Elhadad.* Using lexical chains for text summarization. In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization (Madrid, Spain). 1997. pp. 10–17.

9. *M. Berry, T. Do, G. O'Brien, V. Krishna, and S. Varadhan.* SVDPACK (version 1.0) user's guide. Technical Report CS-93-194, University of Tennessee at Knoxville, Computer Science Department, April 1993.

10. *R. Bruce and J. Wiebe.* Word-sense disambiguation using decomposable models. In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, 1994, pp. 139–146. DOI: 10.3115/981732.981752

11. *M. Ciaramita and M. Johnson.* Explaining away ambiguity: Learning verb selectional preference with Bayesian networks. 2000.

12. *G. W. Cottrell and S. L. Small.* A connectionist scheme for modelling word sense disambiguation – Cognition and brain theory. 1983. № 6. pp. 89–120.

13. *G. W. Cottrell.* A connectionist approach to word sense disambiguation. Pitman, London, 1989.

14. *Charles X. Ling, M. Marinov.* Answering the connectionist challenge: A symbolic model of learning the past tenses of English verbs, Cognition, Elsevier, 1993.

15. *D. Lin.* Automatic Retrieval and Clustering of Similar Words.Proceedings of the 17th international conference on Computational linguistics-Volume 2. – Association for Computational Linguistics, Department of Computer Science University of Manitoba Winnipeg, Manitoba, Canada, 1998, pp. 768-774. DOI: 10.3115/980432.980696

16. *D. Lin.* Principle-based parsing without overgeneration. In Proceedings of ACL-93, Columbus, Ohio, 1993, pp. 112-120. DOI: 10.3115/981574.981590

17. *D. Lin.* Using syntactic dependency as local context to resolve word sense ambiguity. In Proceedings of ACL/EACL-97, Madrid, Spain, July, 1997, pp. 64-71. DOI: 10.3115/979617.979626

18. *D. Hindle.* Noun classification from predicate-argument structures. In Proceedings of ACL-90, Pittsburg, Pennsylvania, June, 1990, pp. 268-275.

19. *D. T. Duong.* Automated text summarization. Graduation Thesis. Hanoi University. 2011.

20. *Eugene A. Nida.* Componential Analysis of Meaning. The Hague, Mouton. 1975.

21. *Y. Freund, R. E. Schapire.* A Short Introduction to Boosting//AT&T Labs Research, Shannon Laboratory. - 1999.

22. *Y. Freund, R. E. Schapire.* Game theory, on-line prediction and boosting. In Proceedings of the Ninth Annual Conference on Computational Learning Theory, 1996. pp. 325-332.

23. *Y. Freund, R. E. Schapire.* A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences. 1997. pp. 119–139. DOI: 10.1006/jcss.1997.1504

24. *M. Halliday, and R. Hasan.* Cohesion in English. London: Longman. 1976.

25. *Z. Harris.* Distributional structure. In: Katz, J. J. (ed.) The Philosophy of Linguistics. New York: Oxford University Press. 1985. pp. 26–47

26. *M. Hearst.* Multi-paragraph segmentation of expository text. In Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics, 9–16. Las Cruces, New Mexico: Association for Computational Linguistics. 1994. DOI: 10.3115/981732.981734

27. *G. E. Hinton, J. L. McClelland, D. E. Rumelhart.* Distributed representations// In Parallel Processing: explorations in the microstructure of cognition. MIT Press, Cambridge, MA, 1986. pp. 5–44.

28. *G. Hirst and D. St-Onge* .Lexical chains as representations of context for the detection and correction of malapropisms. WordNet: An electronic lexical database, 1998. pp. 305-332.

29. *M. Hoey.* Patterns of Lexis in Text. Oxford: Oxford University Press. 1991.

30. *A. Jain and R. Dubes.* Algorithms for Clustering Data. Prentice-Hall, Inc., Upper Saddle River, NJ, 1988.

31. *A. Jain, M. Murthy, and P. Flynn*Data clustering: a review. ACM Computing Surveys, 31(3):264-323, September 1999. DOI: 10.1145/331499.331504

32. *C. Leacock, G. Towell, and E. Voorhees.* Corpus-based statistical sense resolution. In Proceedings of the ARPA Workshop on Human Language Technology, March. 1993, pp. 260–265.

33. *M. Lesk.* Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone – Proceedings of the 5th SIGDOC. New York. 1986. pp. 24–26. DOI: 10.1145/318723.318728

34. *D. Lin and P. Pantel.* Induction of semantic classes from natural language text. In Proceedings of SIGKDD-01. San Francisco, CA. 2001. pp. 317–322. DOI: 10.1145/502512.502558

35. *C. D. Manning and H. Schütze.* Foundations of Statistical Natural Language Processing. MIT Press. 1999.

36. *C. J. Merz and P. M. Murphy.* UCI repository of machine learning databases, 1998. www.ics.uci.edu/ mlearn/MLRepository.html.

37. *G. Miller.* Wordnet: An on-line lexical database. International Journal of Lexicography, 3(4). 1990.

38. *R. J. Mooney.* Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning, Department of Computer Sceinces, University of Texas, Austin, TX 78712-1188, 1996.

39. *R. J. Mooney, M. E. Califf.* Induction of First-Order Decision Lists: Results on Learning the Past Tense of English Verbs, Department of Computer Sceinces, University of Texas, Austin, TX 78712-1188, 1995.

40. *J. Morris, and G. Hirst*Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Computational Linguistics 17(1):21–43. 1991.

41. *R. Navigli.* Word sense disambiguation: A survey. ACM Computing Surveys (CSUR) 41, no. 2 (2009): 10. DOI: 10.1145/1459352.1459355

42. *P. Pantel, D. Lin.* Discovering Word Senses from Text. University of Alberta. Department of Computing Science Edmonton, Alberta T6H 2E1 Canada, 2002. DOI: 10.1145/775047.775138

43. *T. Pedersen.* A Simple Approach to Building Ensembles of Naive Bayesian Classi?ers for Word Sense Disambiguation// Department of Computer Science, University of Minnesota Duluth.–2000.

44. *T. Pedersen and R. Bruce.* Distinguishing word senses in untagged text. Proc.EMNLP.Providence, RI, 1997.

45. *A. Purandare and T. Pedersen.* Improving word sense discrimination with gloss augmented feature vectors // Workshop on Lexical Resources for the Web and Word Sense Disambiguation. – 2004. – pp. 123-130.

46. *P. Resnik.* Selectional preference and sense disambiguation. In Proceedings of the ANLP-97 Workshop: Tagging Text with Lexical Semantics: Why, What, and How? 1997.

47. *J. R. Quinlan.* C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

48. *G. Savova.* Resolving ambiguities in biomedical text with unsupervised clustering approaches. University of Minnesota Supercomputing Institute Research Report, 2005.

49. *R. E. Schapire and Y. Singer.* Improved boosting a predictions. In Proceedings of the Eleventh Annual Confere Theory, 1998. pp. 80–91.

50. *R. E. Schapire.* Using output codes to boost multiclass learning problems. InMachin Learning: Proceedings of the Fourteenth International Conference, 1997. pp. 313–321.

51. *J. Veronis and N. Ide.* Word sense disambiguation with very large neural networks extracted from machine readable dictionaries – Proceedings of the 13th International Conference on Computational Linguistics. Helsinki. 1990. pp. 389–394. DOI: 10.3115/997939.998006

52. *SenseClusters.* http://senseclusters.sourceforge.net

53. *H. Schutze.* Automatic Word Sense Discrimination. Computational Linguistics, vol. 24, number 1., 1998.

54. *UMLS Terminology Services (UTS).* http://umlsks.nlm.nih.gov/kss/servlet/Turbine/template

55. *D. L. Waltz and J. B. Pollack.* Massively parallel parsing: a strongly interactive model of natural language interpretation – Cognitive science. 1985. № 9. pp. 51–74. DOI: 10.1207/s15516709cog0901_4

56. *M. Weeber, J. Mork, A. Aronson.* Developing a test collection for biomedical word sense disambiguation. Proc. AMIA., 2001.

57. *Y. Zhao and G. Karypis.* Evaluation of hierarchical clustering algorithms for document datasets. In Proceedings of the 11th International Conference on Information and Knowledge Management, McLean, VA, 2002. pp. 515-524. DOI: 10.1145/584792.584877

СВЕДЕНИЯ ОБ АВТОРЕ:

Каушинис Татьяна Викторовна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: merilstreet@mail.ru

Кириллов Александр Николаевич
доктор физико-математических наук
доцент
Институт прикладных математических исследований Карельского научного центра РАН
ул. Пушкинская, 11, Петрозаводск, Республика Карелия, Россия, 185910
эл. почта: kirillov@krc.karelia.ru
тел.: (8142) 766312

Kaushinis, Tatiana
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: merilstreet@mail.ru

Kirillov, Alexander
Institute of Applied Mathematical Research, Karelian Research Centre, Russian Academy of Sciences
11 Pushkinskaya St., 185910 Petrozavodsk, Karelia, Russia
e-mail: kirillov@krc.karelia.ru
tel.: (8142) 766312

Коржицкий Никита Иванович
Студент
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: nikita@nikita.tv

Крижановский Андрей Анатольевич
кандитат технических наук
Институт прикладных математических исследований Карельского научного центра РАН
ул. Пушкинская, 11, Петрозаводск, Республика Карелия, Россия, 185910
эл. почта: andew.krizhanovsky@gmail.com
тел.: (8142) 766312

Пилинович Александр
Студент
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: alexander.pilinovich@yandex.ru

Сихонина Ирина Александровна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: syawenka@mail.ru

Спиркова Анна Михайловна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: annspirkova@gmail.com

Старкова Валентина Геннадьевна
старший инженер-программист
Институт прикладных математических исследований КарНЦ РАН
ул. Пушкинская, 11, Петрозаводск, Республика Карелия, Россия, 185910
тел.: (8142) 766312
эл. почта: stark_val@mail.ru

Степкина Татьяна Владимировна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: hogdp@mail.ru

Ткач Станислав Сергеевич
Студент
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел.: (8142) 711078
эл. почта: tkachkras@gmail.com

Чиркова Юлия Васильевна
кандидат физико-математических наук
Институт прикладных математических исследований КарНЦ РАН
ул. Пушкинская, 11, Петрозаводск, Республика Карелия, Россия, 185910
тел.: (8142) 766312 эл. почта: julia@krc.karelia.ru

Korzhitsky, Nikita
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: nikita@nikita.tv

Krizhanovsky, Andrew
Institute of Applied Mathematical Research, Karelian Research Centre, Russian Academy of Sciences
11 Pushkinskaya St., 185910 Petrozavodsk, Karelia, Russia
e-mail: andew.krizhanovsky@gmail.com
tel.: (8142) 766312

Pilinovich, Aleksander
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: alexander.pilinovich@yandex.ru

Sikhonina, Irina
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: syawenka@mail.ru

Spirkova, Anna
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: annspirkova@gmail.com

Starkova, Valentina
Institute of Applied Mathematical Research, Karelian Research Centre, Russian Academy of Sciences
11 Pushkinskaya St., 185910 Petrozavodsk, Karelia, Russia
tel.: (8142) 766312
e-mail: stark_val@mail.ru

Stepkina, Tatiana
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: hogdp@mail.ru

Tkach, Stanislav
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: tkachkras@gmail.com

Chirkova, Julia
Institute of Applied Mathematical Research, Karelian Research Centre, Russian Academy of Sciences
11 Pushkinskaya St., 185910 Petrozavodsk, Karelia, Russia
tel.: (8142) 766312 e-mail: julia@krc.karelia.ru

Чухарев Алексей Леонидович
старший инженер-программист
Институт прикладных математических исследований КарНЦ РАН
ул. Пушкинская, 11, Петрозаводск, Республика Карелия, Россия, 185910
тел.: (8142) 766312 эл. почта: chuharev@krc.karelia.ru

Chuharev, Alexey
Institute of Applied Mathematical Research, Karelian Research Centre, Russian Academy of Sciences
11 Pushkinskaya St., 185910 Petrozavodsk, Karelia, Russia
tel.: (8142) 766312 e-mail: chuharev@krc.karelia.ru

Шорец Дарья Сергеевна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел: (8142) 711078
эл. почта: da_sha1078@mail.ru

Shorets, Daria
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: da_sha1078@mail.ru

Ярышкина Екатерина Александровна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел:(8142) 711078
эл. почта: kate.rysh@gmail.com

Yaryshkina, Ekaterina
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: kate.rysh@gmail.com

Янкевич Дарья Юрьевна
Студентка
Математический факультет
Петрозаводский государственный университет
пр-кт Ленина, 33, Петрозаводск, Республика Карелия
тел: (8142) 711078
эл. почта: dyankevic@gmail.com

Yankevich, Daria
Petrozavodsk State University
33, Lenin Str., 185910, Petrozavodsk, Republic of Karelia, Russia
tel.: (8142) 711078
e-mail: dyankevic@gmail.com

Таблица 8: aaaaa $a_1(m)/m$ kkkkk m

m	1	2	3	4	5	6	7	8	9	10
$\frac{a_1(m)}{m}$	1	1	3	5	7	23	49	52	76	99