

Московский государственный университет имени М.В. Ломоносова

Факультет Вычислительной математики и кибернетики

Кафедра Алгоритмических языков

**Курсовая работа**

**Семантические классы прилагательных**

**Выполнил:**

студент 324 группы

Бондарев Ярослав Алексеевич

**Научный руководитель:**

профессор, доктор физико-математических наук

Мальковский Михаил Георгиевич

Москва, 2018

Содержание

1. Введение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3

2. Детальное описание задачи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4

3. Описание введенных структур \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5

4. Исследование и построение решения задачи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7

5. Обзор используемых инструментов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 8

6. Изложение программной реализации алгоритма \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 9

7. Описание сценария работы с системой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

8. Возможные улучшения алгоритма \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_15

9. Заключение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_16

10. Список литературы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 17

Приложение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 18

1. Введение

Информация - первостепенный ресурс для любого члена социума и тем более в наши дни, когда знание во многом является определяющей силой. Для современного человека автоматическая обработка данных – это уже не новость, однако, не всей информацией одинаково удобно манипулировать. Особое место занимает обработка естественного языка, ибо одной из основных целей информационных технологий является организация максимального комфорта для пользователя при работе с данными. Примером могут послужить различные системы: интернет-собеседники, морфологические анализаторы, программы генерации и распознавания текстов.

На самом деле, описать отдельное слово – достаточно простая задача, ведь существует огромное количество словарей различного направления, но работа с текстом как с набором взаимосвязанных слов становится достаточно нетривиальной, так как необходимо учитывать внушительный набор параметров – от синтаксиса и до прагматики.

В данной работе затрагивается задача определения семантики слова. Важность этого вопроса не подлежит сомнению, однако, приведем пример из области генерации текста. Необходимо создавать текст, наиболее похожий на написанный человеком, а для этого нужно использовать неограниченный лексикон, что для машины, вообще говоря, невозможно. Однако мы можем достаточно искусно имитировать эту неограниченность, используя синонимы, гипонимы, гиперонимы, а уместность их использования обеспечивается с помощью некоторых правил.

Таким образом, можно чётко сформулировать цель моей работы – реализация системы определения семантических классов прилагательных. Данные структуры описаны более подробно в следующей главе.

Моя работа является скорее учебной, однако, я считаю, что возможно усовершенствовать предложенные мной идеи с целью использовать на практике. Изучая данную тему, я ставил перед собой цель разработать некую классификацию прилагательных для перехода между референтами в словосочетаниях. Насколько мне известно, эта тема оригинальна, поэтому вместо обзора существующих решений будут подробно описаны предложенные мной структуры.

2. Детальное описание задачи

Предлагается рассматривать конструкции вида:

<субъект> - <прилагательное> <существительное>

В зависимости от семантических параметров прилагательного на основании описываемых такими предложениями фактов строятся разные (с точки зрения их истинности) следствия из этих фактов.

Рассмотрим пример:

1. Пусть в «базе знаний» описаны следующие правила вывода:

(x) [адвокат (x) художник (x)]

(x) [кот (x) млекопитающее (x)]

2.В зависимости от класса прилагательных истинность следствий различна:

Лебедев – питерский адвокат Лебедев – питерский художник истинно

Лебедев – выдающийся адвокат Лебедев – выдающийся художник истинно/ложно

Мурзик – черный кот Мурзик – черное млекопитающее истинно

Мурзик – крупный кот Мурзик – крупное млекопитающее ложно

Таким образом, можно выделить три класса:

* [питерский, чёрный]
* [выдающийся]
* [крупный]

Необходимо:

1. Описать эти классы
2. Найти другие прилагательные каждого класса
3. Рассмотреть несколько аналогичных примеров
4. Написать программу, которая находит в тексте предложения рассматриваемого типа, предлагает пользователю определить класс очередного прилагательного и формирует словарь таких прилагательных (с информацией об их классе)
5. Написать программу, которая на основе построенного словаря и правил вывода просматривает текст и строит (опять же, для предложений рассматриваемого типа) истинные следствия

3. Описание введённых структур

Итак, рассмотрим более подробно классы, упомянутые в постановке задачи:

1. Следствие истинно – это значит, что возможно осуществить переход от одного существительного к другому, сохранив семантику словосочетания.
2. В данном случае мы не можем установить истинность перехода, т. к. не позволяет прагматика. В примере «адвокат художник» зависимость от прагматики очевидна.
3. Следствие ложно и, следовательно, при замене существительного нарушается семантика (кот не является большим млекопитающим – см. пример из предыдущего раздела)

Также, в процессе работы был составлен словарь, с помощью которого происходит вывод правил. Для его построения было необходимо более детально описать прилагательные русского языка. Насколько известно, существуют два больших подкласса – качественные и относительные прилагательные. Предлагается следующее разбиение:

* Качественные:
  + Форма
  + Цвет
  + Свойство
  + Температура
  + Звук
  + Общая оценка
  + Размер
  + Вкус
  + Запах
* Относительные:
  + Место
  + Отношение к другому предмету
  + Время
  + Единица измерения

Будем называть объекты данного разбиения «метапараметрами». Вообще говоря, данная классификация может быть произвольной, более того, такие метапараметры как «оценка», «свойство» и «отношение» целесообразно разделить для улучшения качества вывода.

Перейдем к рассмотрению словаря. Его структура достаточно проста:

|  |  |  |
| --- | --- | --- |
| Прилагательное | Метапараметр | Одушевленность |

Под «одушевленностью» подразумевается возможность употребления данного прилагательного с одушевленным существительным. Этот параметр позволяет исключить вывод таких словосочетаний, как, например, «спелый кот».

4. Исследование и построение решения

Машинная обработка естественного языка на уровне семантики является весьма сложным вопросом, так как приходится производить глубокий анализ слов и их взаимосвязей с другими словами языка. Однако, специфика рассматриваемой задачи позволяет работать с семантикой более поверхностно.

Предлагается работать с ограниченным набором пар существительных (расширение производится вручную), в частности:

* Кот млекопитающее
* Мост сооружение
* Художник адвокат
* Ложка прибор
* Замок механизм
* Урок занятие
* Компьютер машина
* Менеджер сотрудник
* Вирус программа
* Ножницы инструмент

Вообще говоря, изменение данного набора не потребует модификации алгоритма.

Для каждой пары предлагается рассмотреть набор из следствий по количеству метапараметров – по одному прилагательному для каждого метапараметра в словосочетании. Прилагательное выбирается случайно, но с учётом одушевленности и, разумеется, метапараметра.

Затем производится обучение классификатора: каждому следствию ставится в соответствие один из описанных выше классов. Причем каждому прилагательному с данным метапараметром соответствует один и тот же класс. Если полученное следствие смысла не имеет, то данный метапараметр и все соответствующие ему прилагательные отбрасываются.

Следующим шагом является обработка целевого текста – нахождение предложений рассматриваемого типа в нем и построение различных следствий с указанием их истинности.

5. Обзор используемых инструментов

Программная реализация предложенного алгоритма производилась с помощью языка программирования Python версии 3.6 и СУБД SQLite версии 3.

Отдельно необходимо рассмотреть морфологический анализатор pymorphy2. С его помощью можно:

* приводить слово к нормальной форме (например, “люди человек”)
* ставить слово в нужную форму. Например, ставить слово во множественное число, менять падеж слова
* возвращать грамматическую информацию о слове

При работе используется словарь OpenCorpora, а для незнакомых слов строятся гипотезы.

Рассмотрим пример:

>>> morph.parse("фиолетовый")

[Parse(word='фиолетовый', tag=OpencorporaTag('ADJF,Qual masc,sing,nomn'), normal\_form='фиолетовый', score=0.5, methods\_stack=((<DictionaryAnalyzer>, 'фиолетовый', 238, 0),)), Parse(word='фиолетовый', tag=OpencorporaTag('ADJF,Qual inan,masc,sing,accs'), normal\_form='фиолетовый', score=0.5, methods\_stack=((<DictionaryAnalyzer>, 'фиолетовый', 238, 4),))]

Результатом вызова метода является список возможных разборов слова. Элементом списка является объект, в состав которого входят тег, нормальная форма, вероятность, информация о словаре. Тег состоит из грамматических характеристик слова.

Также необходимо указать точность анализатора:

* случайно выбранный разбор (из допустимых) верен примерно в 66% случаев;
* первый по словарю разбор верен примерно в 72% случаев;

Данный инструмент корректно работает с буквой «ё» и использует некоторые нестандартные граммемы:

* Слово состоит из латинских букв
* Знаки пунктуации
* Числа
  + Целое число
  + Вещественное число
  + Римское число

6. Изложение программной реализации алгоритма

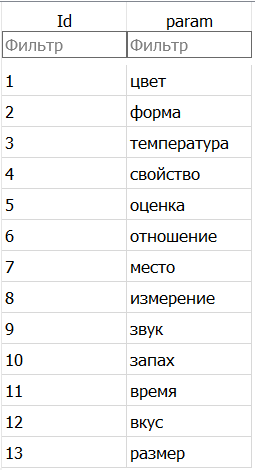
Решение задачи представлено в виде двух программ – обучение классификатора и, собственно, сам классификатор. Также, для хранения данных было создано несколько таблиц в базе данных. Более конкретно:

* Dict – словарь (рисунок 1):
  + word – прилагательное
  + params – метапарметр
  + anim – «одушевленность»

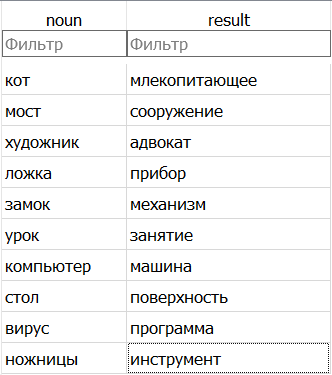


**Рис. 1**

* Metaparams – список метапараметров (рисунок 2)

 **Рис. 2**

* Nouns – результат работы обучающей программы:
  + noun – существительное
  + param – метапарметр
  + class – класс (1-3)
* Rules – набор пар существительных (рисунок 3)



**Рис.3**

Перейдем, непосредственно, к описанию алгоритма:

0. Подготовка к работе

На данном этапе правится набор существительных, для которых строятся следствия, изменяется набор метапараметров и соответствующим образом корректируется словарь. Необходимо отметить, что данный шаг является необязательным и выполняется только по необходимости.

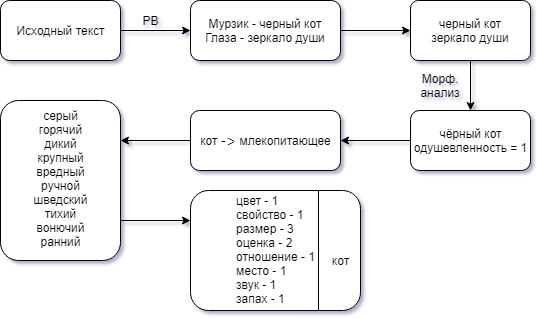
1. Обучение классификатора

С помощью регулярных выражений производится поиск в тексте предложений нужного вида. Затем рассматриваются два слова после символа “-”. Производится определение части речи с помощью морфологического анализатора. Если это прилагательное и существительное, причем последнее есть в Rules, тогда ставим в нормальную форму и запоминаем данное существительное. Под нормальной формой существительного подразумевается форма единственного числа, именительного падежа. Также, из тега наиболее вероятного разбора извлекается информация об одушевленности.

Далее для каждого существительного строится список из словосочетаний. Прилагательные выбираются из таблицы Dict по метапараметру и одушевленности случайно.

Затем на экран выводятся строки вида «прилагательное старое\_существительное прилагательное новое\_существительное», где «новое\_существительное» берется из Rules. Пользователю предлагается поставить цифру от 0 до 3. Нулю соответствует отсутствие смысла словосочетаний, 1-3 – классы истинности следствий. Если словосочетание смысла не имеет, то данный метапараметр не участвует в дальнейшей классификации.

Полученные от пользователя ответы записываются в таблицу Nouns. Краткая схема работы представлена на рисунке 4



**Рис. 4**

2. Собственно, классификация

Из целевого текста извлекаются предложения рассматриваемого типа с помощью алгоритма, использованного в первой программе. Результатом его работы является список существительных.

Из базы данных извлекается информация по найденным существительным.

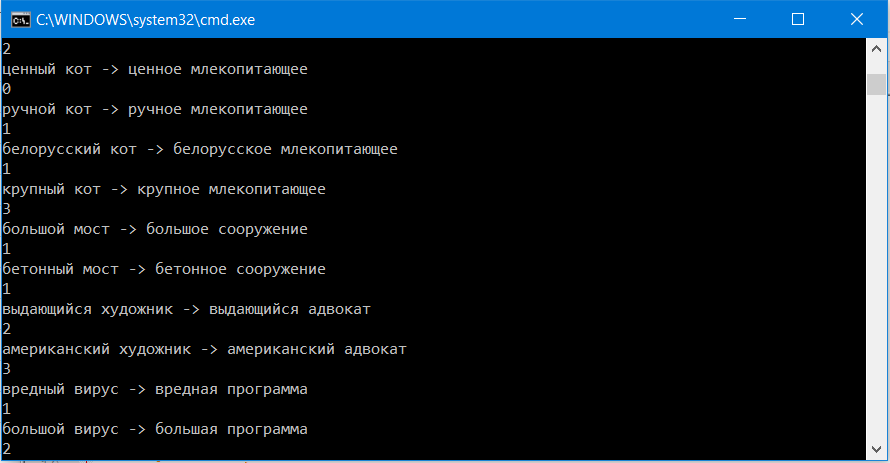
После этого на экран выводятся всевозможные следствия, и указывается их истинность.

7. Описание сценария работы с системой

Запуск происходит с помощью команды:

*>python3 train.py ../data/intext.txt*

* *train.py –* название программы, которая выполняет этап обучения классификатора
* *../data/intext.txt –* файл, из которого производится выборка словосочетаний для обучения



**Рис. 5**

На скриншоте (рисунок 5) представлен интерфейс взаимодействия с пользователем. Программа выводит на экран следствие и предлагает пользователю ввести число для определения класса.

В результате обучения заполняется таблица базы данных, содержащая информацию о классификации (рисунок 6).

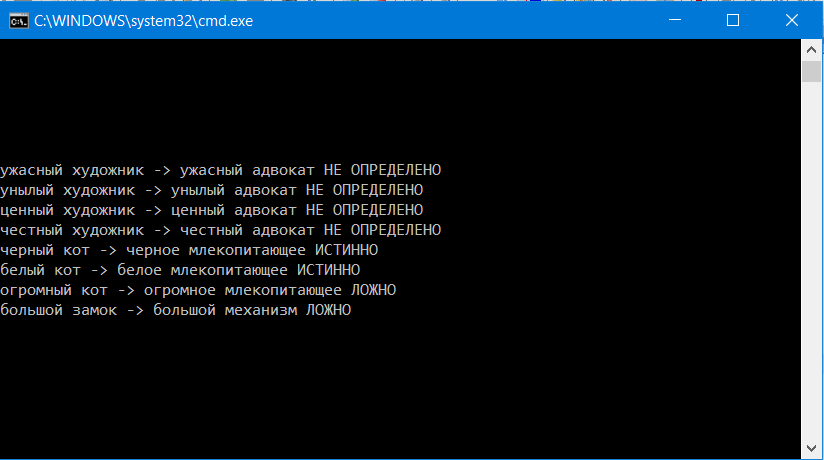


**Рис. 6**

Затем, необходимо запустить программу-классификатор:

*>python3 classify.py ../data/target.txt*

* *classify.py –* название программы
* *../data/target.txt –* текст, который необходимо обработать и проанализировать найденные словосочетания



**Рис. 7**

На рисунке 7 показан результат работы классификатора.

8. Возможные улучшения алгоритма

Как было описано во введении, данная работа является учебной. Однако, с помощью некоторых модификаций предложенное решение можно в некоторой степени эффективно использовать на практике. Итак, рассмотрим возможные изменения алгоритма, которые приведут к более качественной его работе:

* Использование методов, описанных в статье «Enriching a Text by Semantic Disambiguation for Information Extraction». В данном исследовании затрагивается тема поиска в документе. Предлагается наполнять документ словами, которые будут являться синонимами в рамках контекста. Это позволяет повысить точность и полноту результатов поиска.

Используемый алгоритм поиска контекстуальных синонимов можно адаптировать для использования в моей работе. В частности, на этапе обучения классификатора будет весьма полезно подобрать синонимы к текущему прилагательному. Также можно расширить набор существительных с помощью предложенного метода.

* Как упоминалось выше, возможно увеличение количества метапараметров путём разбиения существующих. Данная идея будет весьма эффективна для текстов узкой направленности.
* На данный момент для обучения классификатора используется метод опроса пользователя, однако гораздо более оптимально будет внедрение некоторого алгоритма машинного обучения на примерах.
* Также, возможна разработка и реализация дополнительного алгоритма, который будет строить необходимый для работы системы словарь из произвольных словарей различной направленности и, в частности, викисловаря.

9. Заключение

Сегодня задача определения семантики слова является достаточно важной проблемой. А генерация текстов - составляющей разных аспектов жизни современного человека. Например, составление рекламных объявлений или разработка тестов для учащихся. Вполне возможно, что в будущем большинство таких задач будут решаться машиной, использующей различные алгоритмы для определения семантики или расширения словарного запаса.

Итак, можно подвести итог проделанной работы:

* Разработана оригинальная классификация прилагательных;
* Построен алгоритм логического вывода следствий для словосочетаний вида «прилагательное существительное»;
* Данный алгоритм реализован программно;
* Предложены возможные улучшения данного алгоритма с целью использования на практике.

10. Литература

1. Korobov M.: Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts, 2015. — P. 320-332

2. Гращенков П. В., Кобозева И. М.: Семантические классы и управление прилагательных // Компьютерная лингвистика и интеллектуальные технологии: по материалам международной конференции «Диалог 2017» — С. 134-159

3. Морфологический анализатор pymorphy2 [HTML] (<https://pymorphy2.readthedocs.io>) (16.05.2018)

4. Самоучитель python // Pythonworld [HTML] (<https://pythonworld.ru/samouchitel-python>) (16.05.2018)

5. Редактор диаграмм // Draw.io [HTML] ( <https://www.draw.io>) (16.05.2018)

6. Ожегов С. И., Шведова Н. Ю. Толковый словарь русского языка: 80 000 слов и фразеологических выражений // Российская академия наук. Институт русского языка им. В. В. Виноградова. — 4-е изд., дополненное. — М.: Азбуковник, 1999.

7. Русский язык // Энциклопедия русского языка [HTML] (<http://russkiyyazik.ru>) (16.05.2018)

8. Y. Gotoh, S. Renals: Information Extraction from Broadcast News // Philosophical Transactions: Mathematical, Physical and Engineering Sciences, Vol. 358, No. 1769, Computers, Language and Speech: Formal Theories and Statistical Data (Apr. 15, 2000), pp. 1295-1310

9. Bernard Jacquemin, Caroline Brun, Claude Roux. Enriching a Text by Semantic Disambiguation for Information Extraction. de Loupy, Claude. Proceeding of the Workshop on Using Semantics for Information Retrieval and Filtering: State of the Art and Future Research (LREC 2002), May 2002, Las Palmas, Canary Islands, Spain. pp.45-51, 2002

Приложение

* Функция определения части речи и постановки в нормальную форму слова

def wordToNormalForm**(**word, POS**)**: #str word – слово, str POS – часть речи

morph = pymorphy2.MorphAnalyzer()

tagsWord **=** morph.parse**(**word**)** #разбор слова – список тегов

partsOfSpeech **=** **{}**

for el in tagsWord:

if partsOfSpeech.get**(**el.tag.POS**)** is None: # Заполняем словарь вида **{ч\_речи: вероятность}**

partsOfSpeech**[**el.tag.POS**]** **=** el.score

else:

partsOfSpeech**[**el.tag.POS**]** **+=** el.score

if not partsOfSpeech.get**(**POS**)**: partsOfSpeech**[**POS**]** = 0

POS\_score **=** partsOfSpeech.pop**(**POS**)**

if partsOfSpeech.values**()**:

if POS\_score **<** reduce**(**lambda **x, y:** x **+** y, partsOfSpeech.values**())**:

“““Если вероятность того, что слово является частью речи POS меньше вероятности противоположного события, то возвращается False”””

del morph

return False

for el in tagsWord**:**

if POS in el.tag**:**

del morph

return el.normal\_form # Возвращаем нормальную форму