КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2012 Т. 4 № 4 С. 707–719

Ки&М

ЧИСЛЕННЫЕ МЕТОДЫ И ОСНОВЫ ИХ РЕАЛИЗАЦИИ

УДК: 004.852

Задачи и методы автоматического построения графа цитирований по коллекции научных документов

В. А. Полежаев

Лаборатория «РУКОНТ-ФизТех» ФУПМ МФТИ,
Россия, 141700, г. Долгопрудный, Институтский переулок, д. 9

E-mail: polezhaev@forecsys.ru

Получено 06 сентября 2012 г.

Задача автоматического построения графа цитирования по коллекции научных документов сводится к решению последовательности задач распознавания. Рассматриваются методы решения, их адаптация и объединение в технологическую цепочку, приводятся результаты вычислительных экспериментов для некоторых задач.

Ключевые слова: компьютерный анализ текстов, граф цитирований, библиография, метаописания, мэтчинг, связывание, разметка, сегментация

Automated citation graph building from a corpora of scientific documents

V. A. Polezhaev

RUKONT-PhysTech Laboratory, CMAM department, MIPT, 9 Institutskii per., Dolgoprudny, Moscow Region, 141700, Russia

Abstract. — In this paper the problem of automated building of a citation graph from a collection of scientific documents is considered as a sequence of machine learning tasks. The overall data processing technology is described which consists of six stages: preprocessing, metainformation extraction, bibliography lists extraction, splitting bibliography lists into separate bibliography records, standardization of each bibliography record, and record linkage. The goal of this paper is to provide a survey of approaches and algorithms suitable for each stage, motivate the choice of the best combination of algorithms, and adapt some of them for multilingual bibliographies processing. For some of the tasks new algorithms and heuristics are proposed and evaluated on the mixed English and Russian documents corpora.

Keywords: text mining, machine learning, information extraction, citation graph, bibliography, matching, record linkage, labeling, segmentation, conditional random fields

Citation: *Computer Research and Modeling*, 2012, vol. 4, no. 4, pp. 707–719 (Russian).

Работа выполнена при поддержке Министерства образования и науки РФ (Государственный контракт 07.524.11.4002).

Введение

Рост объемов научной литературы, доступной в электронном виде, приводит к необходимости автоматического выделения ссылок и построения графа цитирований для нужд информационного поиска и наукометрии. Широко известные системы агрегации научного контента CiteSeerX, Google Scholar, MS Academic Search решают эти задачи все еще с недостаточной точностью. Кроме того, в них слабо представлен русскоязычный контент.

В данной работе представлена технология автоматического построения графа цитирования по коллекции научных документов, разрабатываемая в рамках проекта web-сервиса для интеллектуального поиска, классификации и агрегации научной информации в пополняемых мультидисциплинарных коллекциях текстовых документов.

Предлагаемая технология обработки документа включает шесть основных этапов: предварительная обработка, выделение метаописания (если оно присутствует в тексте документа), выделение списков библиографии, разбиение каждого списка на записи, разбиение каждой записи на поля, связывание записей-дубликатов.

Методы решения некоторых задач по отдельности описаны в литературе, но не рассматривались вместе в рамках единой технологии. Целью данной работы является обзор существующих методов, выбор и адаптация наилучшего сочетания методов. Для некоторых подзадач предлагаются новые методы и эвристики и проводятся эксперименты на коллекции русскоязычных и англоязычных текстов.

Задачи анализа текстов

В области анализа текстов можно выделить три класса задач, возникающих при построении графа цитирований. Это задачи разметки, сегментации текста и связывания объектов.

Разметка. Пусть заданы входной и выходной алфавиты $\mathbb{A}$ и $\mathbb{B}$ соответственно. Задача разметки заключается в построении поэлементного отображения последовательности входных символов $a_1 a_2 \ldots a_n$, $a_i \in \mathbb{A}$ на последовательность выходных символов $b_1 b_2 \ldots b_n$, $b_i \in \mathbb{B}$.

Под символом входного алфавита может пониматься некоторый элемент, который описывает объект предметной области, например буква, слово или вектор признаков. Под символом выходного алфавита обычно понимается метка класса.

Таким образом, задача разметки может рассматриваться как задача классификации, где объектами являются последовательности входных элементов, а результатом классификации является последовательность меток классов.

В простейшем случае каждый элемент может быть классифицирован независимо от других с помощью известных методов классификации, предполагающих независимость объектов в выборке. Но в таком случае не учитываются взаимосвязи между элементами в последовательности, что, как правило, критично для решения задачи разметки.

Простым способом учета последовательной структуры входных данных является использование скользящего окна при построение признакового описания. При этом признаковое описание расширяется контекстными признаками, значения которых соответствуют значениям базовых признаков для соседних объектов.

Такой прием позволяет использовать известные методы SVM [Kudoh, Matsumoto, 2000], Winnow [Zhang, 2001] или решающие деревья для отдельной классификации каждого элемента входной последовательности.

Сложности возникают, когда помимо признаков требуется учитывать метки классов объектов-соседей, которые известны на момент обучения, но не известны на момент классификации. В таком случае применяются эвристики голосования или сглаживания.

Более совершенные методы основаны на математическом аппарате графических моделей. Графические модели описывают вероятностные распределения со сложными зависимости между переменными, которые представляются в виде графа и позволяют естественным образом учитывать структурные ограничения.

В терминах задачи разметки, каждой позиции i входной последовательности ставится в соответствии переменная x_i, область значений которой совпадает с множеством значений $\mathbb{A}$. Аналогично, для каждой позиции i выходной последовательности ставится в соответствие переменная y_i с областью значений $\mathbb{B}$. Переменные x_i называются наблюдаемыми, y_i — скрытыми. Значения скрытых и наблюдаемых переменных, соответствующие некоторым входным и выходным последовательностям образуют конфигурации наблюдаемых и скрытых переменных соответственно.

Задача разметки состоит в поиске наиболее вероятной конфигурации скрытых переменных при условии заданной конфигурации наблюдаемых переменных.

Первыми широкое применение получили скрытые марковские модели (Hidden Markov Models, HMM [Rabiner, 1989]), однако они позволяют учитывать лишь простейшие зависимости между переменными и ограничивают в использовании признакового описания. Были разработаны некоторые обобщения, которые частично устранили эти недостатки [Skounakis at al., 2003; Yu, 2010].

На данный момент для решения задачи разметки в анализе текстов наибольшую популярность приобрели модели условных случайных полей (Conditional Random Fields, CRF [Lafferty, 2001]). В CRF удалось решить характерные для графических моделей проблемы вывода, что позволило использовать полноценное признаковое описание и учитывать сложные зависимости между переменными.

Сегментация. Задача сегментации схожа с задачей разметки, но анализируемые последовательности имеют характерную блочную структуру. Выходная последовательность $b_1 b_2 \dots b_n$ разбивается на части-блоки, внутри каждого блока все символы совпадают. Также, в отличие от разметки, методы решения задачи сегментации обычно используют кластеризацию. Общие для большинства методов сегментации черты заключаются в определении функции схожести на блоках и задании функционала качества сегментации. В [Beeferman et al., 1999; Choi et al., 2001; Brants, 2002] описаны методы сегментации текстов, основанные на представлении блоков как «мешка слов».

Связывание. В текстах на естественном языке один и тот же объект, понятие или сущность могут называться по-разному. Например, в названиях журналов, конференций могут использоваться сокращения, а в именах авторов может быть изменен порядок слов или использованы инициалы.

Задача связывания (Record Linkage, RL) — это задача идентификации различных представлений одного объекта или поиска дубликатов представлений. В литературе также упоминается как «matching». Связывание можно выполнять для объектов самой разной природы, в данном случае будет рассматриваться случай, когда объекты представлены в виде структур с текстовыми полями (например, метаописание документа с полями «автор», «название» и т. д.).

Обзор техник связывания дан в [Elmagarmid et al., 2007]. Стандартной техникой для определения дубликатов представлений является решение задачи классификации, в которой объектами распознавания являются пары структур, классов два — «дубликаты», «не дубликаты». В качестве классификатора обычно используется SVM.

В качестве признаков рассматриваются различные функции сравнения двух строк. Так как объекты представлены в виде структур с текстовыми полями, то при заданных K функциях сравнения строк, применяя каждую из функций к каждому полю, можно составить признаковое описание размерности $M \cdot K$, где M — число полей.

Согласно [Cohen et al., 2003; Bilenko et al., 2003] выделяются следующие типы функций сравнения строк.

1. Статические.
   (a) Character-based. Основаны на посимвольном сравнении. Расстояние Левенштейна, расстояние Смита–Ватермана, функция Джаро и другие.
   (b) Token-based. Основаны на сравнении строк как «мешка слов». Функция Монжа–Элкана, схожесть Джаккарда, TF-IDF.
   (c) Hybrid. Двухуровневые. Представляют собой token-based функции, но схожесть токенов определяется по character-based метрике второго уровня. Адаптированная функция Монжа–Элкана, Soft-TFIDF.
   (d) Предикаты. Являются некоторым логическим утверждением для пары строк, например, когда в строках одинаковое число слов.
2. Обучаемые. Основаны на применении алгоритмов машинного обучения для лучшей настройки параметров метрики (например веса операций в расстоянии Левенштейна) под конкретные данные.

Применение обучаемых функций требует дополнительных затрат на подготовку обучающих данных и сам процесс обучения. В то же время показано, что использование только статических функций позволяет добиться приемлемого качества.

Обычно требуется искать объекты-дубликаты внутри множества или на паре множеств. Простейший способ — классификация всех возможных пар объектов — вычислительно не эффективен. Обычно используются методы, позволяющие заранее отбрасывать пары, не являющиеся дубликатами. Обзор методов дан в [Christen, 2012].

Предварительная обработка

Перед решением основной задачи выделения и анализа списков библиографии необходимо очистить входные данные для повышения качества распознавания. Предполагается, что входными являются неформатированные тексты, выделяемые из файлов распространенных форматов, таких как pdf, doc, html, ps, djvu, rtf. В результате выделения текста из форматов, имеющих физическую, а не логическую разметку (pdf, ps), а также полученных в результате сканирования, могут появляться различные артефакты — колонтитулы, номера страниц, фрагменты формул, таблиц и графиков, мешающие правильному распознаванию списков библиографии.

Удаление номеров страниц. Сделаем несколько предположений: (1) каждый колонтитул с номером страницы представляется в тексте отдельной строкой, (2) в одном документе номера страниц оформлены в одном стиле, (3) номера страниц могут окружаться некоторым текстом, например «= 12 =» или «Page 12 of 16», длина которого не превышает заданного порога ℓ_0.

Введем три вспомогательные функции от двух строк s, t, содержащих номера страниц.

$L(s,t)$ — модифицированное расстояние Левенштейна [Cohen et al., 2003], в котором замены букв штрафуются больше, чем замены цифр, и каждая следующая замена штрафуется больше, чем предыдущая.

$S(s,t)$ — функция штрафа за различия чисел в строках.

$$S(s,t) = \begin{cases} C_1(\sigma(t) - \sigma(s)), & \sigma(t) > \sigma(s), \\ C_2, & \sigma(t) \le \sigma(s), \end{cases}$$

где $\sigma(t)$ — сумма всех целых чисел в строке t.

$F(s,t)$ — функция штрафа за большое удаление строк в тексте друг от друга.

$$F(s,t) = \begin{cases} 0, & C_3 < N(t) - N(s) < C_4, \\ C_5, & \text{иначе}, \end{cases}$$

где $N(t)$ — порядковый номер строки t в тексте, $C_4 > C_3 \geq 0$.

Наконец, определим *функцию порядка* $P(s,t) = L(s,t) + S(s,t) + F(s,t)$, значение которой тем меньше, чем более правдоподобно, что s и t — две последовательные строки с номерами. Параметры функции $P(s,t)$ подбираются экспериментально.

Для удаления строк с номерами страниц сначала выбираются все *строки-кандидаты*, содержащие цифры и имеющие длину не более $\ell_0 = 20$. Затем строятся всевозможные *последовательности-кандидаты* из строк-кандидатов. Строится матрица A, $A_{ij} = P(s_i, s_j)$, где s_i — строка-кандидат с номером i. Если значение A_{ij} больше заданного порога, то, будем считать, что строка s_i заведомо не может располагаться перед s_j, и полагать $A_{ij} = N$, где N — специальная метка. В дальнейшем при построении последовательностей, метки N будут игнорироваться, что уменьшит вычислительные затраты. Последовательности-кандидаты строятся по матрице A путем добавления новых элементов таким образом, чтобы минимизировать значение функции порядка от последнего элемента в последовательности и добавляемого элемента. В качестве первых элементов последовательностей выбираются те, которым соответствуют столбцы матрицы A, состоящие только из меток N.

На практике оказалось, что данный алгоритм, помимо последовательности номеров страниц, находит также последовательности подписей к рисункам и таблицам. Поэтому целесообразно удалять все найденные последовательности без выделения какой-либо одной из них.

Удаление колонтитулов. Как и в случае с номерами страниц, будем предполагать, что колонтитулы представляются отдельными строками в тексте. Необходимо учесть, что колонтитулы в одном тексте могут незначительно варьироваться, чередоваться и изменяться.

Определим бинарную *функцию сходства* строк $B(s,t)$. Если длины строк s, t отличаются более чем на 5 символов, то $B = 0$. Если строки полностью совпадают, то $B = 1$. Если у строк не совпадают ни префиксы, ни суффиксы, $B = 0$. Иначе вычисляется значение меры сходства Джаро–Винклера [Cohen et al., 2003], и если оно превышает 0.95, то $B = 1$, иначе — $B = 0$.

Аналогично удалению номеров страниц, для поиска последовательностей строк колонтитулов сначала отбираются строки-кандидаты, длина которых превышает заданный порог. Затем вычисляется значение функции сходства на всех парах строк-кандидатов и множество всех строк-кандидатов разбивается на группы, содержащие вместе с каждым объектом как минимум один объект, схожий с ним.

На практике оказалось, что данный алгоритм, помимо группы строк колонтитулов, находит и другие подпоследовательности схожих строк, в частности группы похожих формул. Поэтому целесообразно удалять все найденные группы.

Выделение метаописания

При обработке документов, размещенных в открытом доступе в Интернете, *метаописание документа*, содержащее библиографические данные (заголовок, список авторов, название журнала, том, страницы и т. д.) не всегда может быть получено из того контекста, где была обнаружена ссылка на данный документ. Эти данные могут содержаться внутри самого документа: для статьи — в заголовке или в колонтитуле; в случае книги, отчета, диссертации — на первых страницах.

Задача выделения метаописания состоит в формировании структуры с библиографической информацией из текста самого документа.

В [Giles et al., 2003] предложен метод выделения полей метаописания из текста документа, основанный на решении задачи классификации, в которой объектами являются строки, классами — типы полей (заголовок, список авторов и т. д.), признаками — характеристики строк (номер строки в тексте, доля цифр в строке, число слов в строке, число слов из контекстного словаря и т. д.). Контекстный словарь составляется для каждого типа полей и содержит слова, часто встречающиеся в полях данного типа.

Для каждого класса строится отдельный классификатор SVM. Таким образом, каждая строка может быть отнесена к нескольким классам. Обучающая выборка формируется из документов, в которых метаописания размечены вручную. Процесс классификации является итерационным: на каждой итерации выполняется контекстная классификация, учитывающая классификацию на предыдущем шаге. Для строк, отмеченных несколькими классами, применяются специальные эвристические алгоритмы для нахождения частей строк, соответствующих конкретным классам.

Выделение списков библиографии

Списки библиографии выделяются для дальнейшего их разбиения на библиографические записи и анализ.

В общем случае тексты могут иметь самую разную логическую структуру, например список библиографии может находиться не в конце документа, если текст выделен из сборника, то списков может быть несколько. Сами же библиографические ссылки могут быть и обычно оформлены в разных стилях.

Это обуславливает использование методов машинного обучения.

Определение типа текста. Если текст получен из документа с физической разметкой (например, pdf), то он разбит на строки по ширине страницы или колонки. Библиографические записи обычно оформляются как абзацы, но в данном случае абзацы оказываются разрезанными на несколько частей-строк. Если же структура абзацев сохранена, то можно считать, что каждая библиографическая запись представлена одной строкой. Задача состоит в определении типа текста — сохранена ли структура абзацев.

Факт разбиения абзацев влияет также на способ генерации признакового описания для задачи выделения списков библиографии, что будет показано далее.

Предлагается эвристический алгоритм, использующий только информацию о длинах строк текста. Он основан на нескольких наблюдениях: (1) если текст разбит по строкам, то большинство строк в тексте относительно короткие и примерно одинаковой длины; (2) однако могут встречаться и заметно более длинные строки, вызванные артефактами извлечения; (3) если текст разбит по абзацам, то распределение длин строк имеет дисперсию, сравнимую с квадратом средней длины абзаца; (4) строки малой длины (меньше 30) можно игнорировать.

Определение типа документа осуществляется с помощью решающего правила: если дисперсия длин строк менее 1000, то считать, что текст разбит по строкам, иначе — по абзацам. В экспериментах это правило показало 99 % точности.

Алгоритм выделения списка библиографии. Задача выделения списков библиографии ставится как задача разметки или сегментации на строках текста с классами «принадлежит библиографии», «не принадлежит библиографии».

Рассматривались два алгоритма — стандартный алгоритм классификации и алгоритм, основанный на графических моделях. В первом случае в качестве классификатора было выбрано решающее дерево C4.5, как показавшее высокое качество вместе с хорошей скоростью работы, во втором — CRF.

Использовался набор из 33 признаков, из которых наиболее информативными оказались: число знаков препинания в строке; начинается ли строка с цифры; число слов, начинающихся с заглавных букв; число двух подряд идущих слов, начинающихся с заглавных букв; число инициалов; число цифр; число точек; число заглавных букв; число ключевых слов («С.», «no», «vol.» и др.) число четырехзначных чисел; длина строки относительно средней длины библиографической ссылки (220 символов); нормированное расстояние до предыдущего ключевого слова, например «References» или «Литература».

Для учета структурных ограничений задачи в случае использования дерева C4.5 [Quinlan, 1993] можно воспользоваться схемой скользящего окна при формировании признакового описания либо использовать контекстную классификацию.

Контекстная классификация выполняется на результатах базовой классификации, полученной решающим деревом C4.5 по правилу: строка относится к тому классу, к которому отнесено большинство из k соседних строк. Экспериментально было найдено оптимальное значение параметра $k = 10$.

В качестве модели CRF использовалась линейная модель второго порядка с признаками, построенными для окна размера 5. Несмотря на то, что модель CRF учитывает последовательную структуру данных, использование дополнительно контекстной классификации позволяет улучшить качество.

Допустим, в результате определения типа текста оказалось, что абзацы разбиты по ширине страницы или колонки, то есть библиографические записи оказываются разделены на несколько частей.

Рассмотрим реальный пример:

Mohri, M. (2000). Minimization algorithms for sequential transducers. Theoretical Computer Science, 234, 177–201.

Вторая строка вне контекста слабо отличима от обычного текста, следовательно, в поставленной задаче классификации на строках признаковое описание таких строк слабо отличается от строк вне списков библиографии. Это означает пересечение классов и ухудшение качества распознавания.

Для решения этой проблемы можно опять же воспользоваться скользящим окном при генерации признаков, либо использовать суммирование значений соответствующих признаков соседних строк из скользящего окна ширины S, определяемой для каждого текста индивидуально.

В экспериментах полагалось $S = [L/W]$, где $L = 220$ — средняя длина записи, W — средняя длина строки, оцененная с помощью робастного алгоритма.

Для оценки средней длины строки предлагается следующий алгоритм.

Строится гистограмма длин строк текста $Y(X)$, которая затем нормируется на отрезок $X \in [0; 100]$. После этого выполняется сглаживание и домножение значения гистограммы $Y(X)$ в каждой точке на значение X. В качестве искомого значения выбирается $\arg\max\ Y(X)$ с учетом выполненной ранее нормировки на отрезок $[0; 100]$.

Этот алгоритм на практике оказался устойчивым к различным выбросам и артефактам извлечения текста.

Эксперименты. Эксперименты выполнялись на размеченной коллекции из 70 текстов, включающих англоязычные тексты по computer science и русскоязычные авторефераты, по схеме

Таблица 1. Качество (F_1-мера) выделения списка библиографии для текстов, разбитых по строкам.

	C4.5				CRF	
	S	M	+S	+M	+S	+MS
сред.	0.811	0.833	0.886	**0.937**	0.924	**0.948**
мин.	0.144	0.147	0.199	0.500	0.295	0.504
макс.	0.967	1.000	1.000	1.000	1.000	1.000
откл.	0.161	0.145	0.086	0.093	0.137	0.109

Таблица 2. Качество (F_1-мера) выделения списка библиографии для текстов с сохраненной структурой абзацев.

	C4.5 S	C4.5 +S	CRF +S
сред.	0.943	**0.974**	**0.982**
мин.	0.838	0.925	0.926
макс.	0.992	0.993	1.000
откл.	0.046	0.019	0.017

скользящего контроля leave-one-out. Каждый из N текстов по очереди рассматривался как контрольный, на объектах из остальных $N - 1$ текстов происходило обучение. Оценка качества вычислялась как среднее по всем контрольным текстам значение F_1-меры для класса «принадлежит блоку библиографии».

Результаты экспериментов приведены в таблицах 1, 2. Метки S и M обозначают способы генерации признаков: S — скользящее окно, M — суммирование значений; + означает использование контекстной классификации.

Эксперименты показали, что CRF лучше решает задачу, чем C4.5, но требует больше времени для обучения и классификации. Метод, не использующий скользящее окно и суммирование значений при генерации признаков, работает существенно хуже, значение F_1-меры ниже 70 %.

Разбиение списков на записи

Подзадача разбиения списка библиографии на библиографические записи актуальна только для текстов, разбитых по строкам, в противном случае сами абзацы и будут записями.

Задача ставится как задача разметки, в которой объектами являются строки, классов два — «первая строка записи» и «не первая строка». Это минимальный набор классов, который позволяет по разметке разбить списки на записи. Рассматривались и некоторые избыточные случаи (например добавление класса «последняя строка записи»), но выигрыша в качестве они не дали. Для решения использовались те же методы C4.5, CRF и тот же набор признаков. Контекстная классификация, скользящее окно и суммирование значений не использовались.

Эксперименты выполнялись по аналогичной схеме. Оценкой качества являлась средняя по классам F_1-мера. В таблице 3 приведены результаты экспериментов на всей коллекции. Теперь классификатор C4.5 показал лучшее качество по сравнению с CRF.

Стандартизация

На этапе стандартизации выделенные на предыдущем этапе текстовые строки, представляющие библиографические записи, необходимо преобразовать в структуры с библиографическими полями «автор», «название», «издание», «том», «номер», «организация», «конференция», «издатель», «редактор», «место», «год», «страницы».

Таблица 3. Качество (F_1-мера) выделения записей из списка библиографии.

	C4.5	CRF
сред.	**0.972**	0.854
мин.	0.877	0.543
макс.	1.000	1.000
откл.	0.031	0.092

Стандартизация является задачей разметки, где объектами распознавания являются выделенные текстовые последовательности, в которых элементами являются слова и символы пунктуации, а метки классов представлены библиографическими полями. Основная трудность задачи стандартизации связана с многообразием форматов библиографических записей.

Первые системы стандартизации представляли собой вручную составленные экспертные правила, позднее стал активно применяться аппарат графических моделей. Сначала использовались простые системы на основе одной модели HMM [Christen et al., 2002], затем появились более сложные системы, использующие композицию из нескольких HMM [Borkar et al., 2001; Agichtein, Ganti, 2004]. На данный момент наилучшими считаются следующие два метода — на основе CRF и FLUX-CiM [Cortez et al., 2007; Councill et al., 2008].

Среди графических моделей наилучшим методом стандартизации библиографических записей показал себя алгоритм, основанный на CRF. Модель CRF второго порядка обеспечивает качество выделения большинства ключевых полей порядка 97 % на тестовых данных [Councill et al., 2008].

Как правило, в подобных задачах используется модель CRF, обучаемая с учителем, то есть требующая вручную размеченных обучающих данных. На практике качество распознавания существенно зависит от того, насколько много форматов представления библиографических записей содержится в анализируемых текстах и насколько эти форматы различны.

Метод FLUX-CiM не требует размеченных вручную данных. Вместо этого предлагается собрать для каждого извлекаемого поля словарь, которые когда-либо встречались в этом поле. Например, для поля «автор» — это, очевидно, будет коллекция из имен и фамилий. Также хранится информация о числе вхождений каждого из слов. Во время распознавания решение о принадлежности слова полю принимается на основе этой информации.

Для создания такой базы можно использовать внешние источники (библиографические базы DBLP, PubMed, словари названий журналов, конференций, издательств и т. д.), главное требование — возможность выделить из данных соответствующие поля.

На известных текстовых коллекциях FLUX-CiM показал лучшее качество [Cortez et al., 2007], чем алгоритм, основанный на CRF. Однако качество работы FLUX-CiM напрямую зависит от полноты его базы, создание которой является организационно-технической проблемой.

Построение графа цитирования

В результате решения предыдущих подзадач для каждого поступившего текста выполнено выделение метаописания, выделены и разобраны по полям библиографические записи. На последнем этапе выполняется построение графа цитирования.

Граф цитирования — это направленный граф, вершины которого соответствуют документам, ребра — отношению цитирования.

Для формирования графа цитирования необходимо выполнить связывание метаописаний документов с выделенными в пристатейных списках литературы библиографическими записями.

Как структура с полями метаописание является расширением библиографической записи, а значит метаописания и записи можно сравнивать по общему подмножеству полей. Таким образом, для построения графа цитирования можно пользоваться стандартным способом связывания объектов, описанным в разделе 2.3.

Тем не менее есть ряд особенностей, которые требуют адаптации базового алгоритма связывания. Во-первых, процесс построения графа цитирования инкрементный, то есть при поступлении новых документов в граф должны добавляться новые вершины и ребра. Во-вторых, обновление графа при добавлении пакета документов может быть реализовано эффективнее, чем при добавлении документов по одному.

В качестве базового алгоритма обновления графа цитирования предлагается использовать модификацию метода адаптивного мэтчинга (Adaptive matching, AM) и метода вычислительной оптимизации Canopies, описанных в [McCallum et al., 2000; Cohen, Richman, 2002].

Адаптивный мэтчинг. Рассмотрим два множества $\mathbb{A}$, $\mathbb{B}$, их подмножества $A \subset \mathbb{A}$, $B \subset \mathbb{B}$. Объектами являются пары $X \subseteq A \times B$, а решением — множество пар $Y \subseteq X$.

Адаптивный мэтчинг выполняет связывание элементов a и b множеств A, B, в общем случае разной природы, чтобы минимизировать заданную функцию потерь. Связанные элементы обозначаются парой (a, b).

Для предлагаемого решения Y^* и правильного решения Y функцию потерь можно ввести как мощность симметрической разности множеств Y^* и Y.

Алгоритм [Cohen, Richman, 2002] основан на использовании стандартной техники — бинарного классификатора на объектах парах $(a, b) \mid a \in A, b \in B$. Классификатор принимает решение о связывании (a, b) элементов a и b. То есть решение Y для объектов X может быть получено классификацией всех возможных пар $(a, b) \in A \times B$.

В данном случае признаковое описание составлено из функций сравнения пары строк, как описано в разделе 2.3.

Если представить объекты множеств A и B точками на плоскости, а пару (a, b) — ребром, соединяющим точки a, b, то результатом работы алгоритма будет двудольный граф.

Основное отличие, которое требуется для применения адаптивного мэтчинга к задаче построения графа цитирования заключается в необходимости связывания элементов не только между двумя множествами A и B, но и внутри одного из множеств, пусть для определенности B.

Несмотря на то что связываются объекты разной природы (метаописания и библиографические записи), их представление как структур идентично, то есть $\mathbb{A} = \mathbb{B}$. Тем не менее объединять A и B в одно множество и искать дубликаты внутри него в задаче построения графа цитирований неэффективно, что будет показано далее.

Предлагаемый алгоритм адаптивного мэтчинга описан в Алгоритме 1, модификация вспомогательного алгоритма Canopies — в Алгоритме 2.

Первое отличие от базового алгоритма в том, что модификация Canopies (Алгоритм 2) строит пары-кандидаты не только как подмножество $A \times B$. Для каждого элемента a множества A выбираются близкие к нему элементы b из множества B. Далее для каждого выбранного элемента b выбирается множество близких к нему элементов из того же множества B. Выбранные таким образом элементы образуют результирующее множество пар, подмножество $(A \times B) \cup (B \times B)$.

Второе отличие заключается в добавлении пар, которые не были отмечены классификатором непосредственно. Считается, что внутри каждой компоненты связности любая пара объектов является дубликатами.

Как будет показано далее, мэтчинг между множествами A и B соответствует мэтчингу уже находящихся в базе объектов и новых, а мэтчинг на множестве B соответствует поиску дубликатов среди новых объектов.

Алгоритм 1 Модифицированный алгоритм адаптивного мэтчинга

TRAIN

Вход: $Y^* = \{(a,b) \mid a \in A, b \in B\}$ — объекты пары (решение) для обучения;
$LearningSet = \varnothing$;
$SetOfPairs = Canopies(A,B)$;
для $(a,b) \in SetOfPairs$
$(a,b).Label = \begin{cases} \text{«+»}, if\ (a,b) \in Y^* \\ \text{«—»}, otherwise \end{cases}$
$LearningSet.Add((a,b))$;
return $TrainClassifier(LearningSet)$;

MATCH

Вход: A, B — элементы для связывания, Alg — обученный классификатор;
$SetOfPairs = Canopies(A,B)$;
$G = (V,E) \mid V = \{v \mid v \in A \cup B\}, E = \varnothing\}$;
для $(a,b) \in SetOfPairs$
если $Alg(a,b) =$ «+» **то**
$E.Add((a,b))$;
$C = Components(G)$; — вернуть компоненты связности, внутри каждой компоненты любая пара объектов является дубликатами
return C;

Алгоритм 2 Модифицированный алгоритм Canopies

Вход: множества A, B, вычислительно не трудоемкая функция сравнения строк $Distance$, параметр T_{loose}, определяемый эвристическим путем;
$CandidatePairs = \varnothing$;
$PossibleCentres = A$;
пока $PossibleCentres \neq \varnothing$
$a = PickRandom(PossibleCentres)$;
$Canopy(a) = \{(a,b) \mid b \in B\ \ Distance(a,b) \leq T_{loose}\}$;
$CandidatePairs.AddAll(Canopy(a))$;
для $(a,b) \in Canopy(a)$
$PairsToAdd = \{(b,x) \mid x \in B)\ \ Distance(b,x) \leq T_{loose}\}$;
$CandidatePairs.AddAll(PairsToAdd)$;
$PossibleCenters.Remove(a)$;
return $CandidatePairs$;

Алгоритм обновления графа цитирований. Наконец, опишем процесс обновления графа цитирования по коллекции документов.

Пусть O — объекты-метаописания, уже находящиеся в базе, S — библиографические записи в базе, для которых не были найдены соответствующие документы (не удалось найти подходящие метаописания), N — новые библиографические записи, выделенные из поступивших документов, I — новые метаописания, выделенные из поступивших документов.

Обозначим процедуру адаптивного мэтчинга (Алгоритм 1) на множествах A, B через $AM(A,B)$.

На первом шаге выполняется $AM(O, I)$. Это соответствует проверке на наличие в базе документов, которые добавляются. Данный шаг можно заменить просто «жестким» сравнением, однако использование адаптивного мэтчинга является более гибким.

По результатам первого шага выполняется включение новых документов в граф.

Затем выполняется $AM(O \cup S, N)$. Связываются новые выделенные библиографические записи (N) с теми, что уже есть в базе. Это или сами документы (O), или если не получается, то другие ссылки, для которых не было найдено документов (S). Если соответствий не найдено, они добавляются в S.

Наконец, выполняется $AM(I, S)$. Теперь связываются те записи, которые находятся в хранилище (которые не удалось соотнести с метаописаниями) (S) с новыми метаописаниями (I).

По результатам перечисленных шагов происходит обновление графа цитирований. Ключевой особенностью является возможность применять один и тот же алгоритм связывания многократно для решения однотипных подзадач.

Заключение

В статье описан процесс построения графа цитирований по коллекции научных документов как решение последовательности задач распознавания, дан обзор существующих методов, обоснован выбор лучших методов, предложены их адаптации.

Проведенные в статье эксперименты, а также результаты, полученные другими исследователями, позволяют рассчитывать на успешное решение конечной задачи построения графа цитирований с достаточно высокой точностью.

Благодарности

Автор выражает благодарность своему научному руководителю профессору, д.ф.-м.н. К. В. Воронцову за советы и конструктивные замечания.

Список литературы

Agichtein E. and Ganti V. Mining reference tables for automatic text segmentation. *In proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*, pages 20–29. ACM Press, 2004.

Beeferman D., Berger A., and Lafferty J. Statistical models for text segmentation. In *Machine Learning*, pages 177–210, 1999.

Bilenko M., Mooney R., Cohen W., Ravikumar P., and Fienberg S. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18:16–23, September 2003.

Borkar V., Deshmukh K., and Sarawagi S. Automatic segmentation of text into structured records, 2001.c

Brants T. Topic-based document segmentation with probabilistic latent semantic analysis. *In Proceedings of CIKM (McLean*, pages 211–218. ACM Press, 2002.

Choi F. Y. Y., Wiemer-Hastings P., and Moore J. Latent semantic analysis for text segmentation. *In Proceedings of EMNLP*, pages 109–117, 2001.

Christen P. A survey of indexing techniques for scalable record linkage and deduplication. *IEEE Transactions on Knowledge and Data Engineering*, 24:1537–1555, 2012.

Christen P., Churches T., and Zhu J. X. Probabilistic name and address cleaning and standardisation, 2002.

Cohen W. W. and Richman J. Learning to match and cluster large high-dimensional data sets for data integration, 2002.

Cohen W. W., Ravikumar P., and Fienberg S. E. A comparison of string distance metrics for name-matching tasks. Pages 73–78, 2003.

Cortez E., da Silva A. S., Gonçalves M. A., Mesquita F., and de Moura E. S. Flux-cim: flexible unsupervised extraction of citation metadata. In *JCDL '07: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, pages 215–224, New York, NY, USA, 2007. ACM.

Councill I. G., Giles C. L., and Kan M. y. Parscit: An open-source crf reference string parsing package. In *International language resources and evaluation*. European Language Resources Association, 2008.

Elmagarmid A. K., Ipeirotis P. G., and Verykios V. S. Duplicate record detection: A survey. *Transactions on knowledge and data engineering*, page 2007, 2007.

Giles H. H. C. L., Manavoglu E., Zha H., Zhang Z., and Fox E. A. Automatic document metadata extraction using support vector machines. *In JCDL '03: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 37–48, 2003.

Kudoh T. and Matsumoto Yu. Use of support vector learning for chunk identification. *In Proceedings of CoNLL-2000 and LLL-2000*, pages 142–144, 2000.

Lafferty J. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282–289. Morgan Kaufmann, 2001.

McCallum A., Nigam K., and Ungar L. H. Efficient clustering of high-dimensional data sets with application to reference matching, 2000.

Quinlan J. R. C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

Rabiner L. R. A tutorial on hidden markov models and selected applications in speech recognition. 77(2):257–286, 1989.

Skounakis M., Craven M., and Ray S. Hierarchical hidden markov models for information extraction. *In Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 427–433. Morgan Kaufmann, 2003.

Yu S.-Z. Hidden semi-markov models. *Artificial Intelligence*, 2010.

Zhang T., Damerau F., and Johnson D. Text chunking based on a generalization of winnow. *Journal of Machine Learning Research*, 2:615–637, 2001.