УДК 004.048
DOI: 10.15827/0236-235X.030.1.085-099
Дата подачи статьи: 19.07.16
2017. Т. 30. № 1. С. 85–99

МЕТОДЫ АВТОМАТИЧЕСКОЙ КЛАССИФИКАЦИИ ТЕКСТОВ

Т.В. Батура, к.ф.-м.н., ведущий научный сотрудник, tatiana.v.batura@gmail.com (Новосибирский государственный университет, ул. Пирогова, 2, г. Новосибирск, 630090, Россия); старший научный сотрудник (Институт систем информатики им. А.П. Ершова СО РАН, просп. Лаврентьева, 6, г. Новосибирск, 630090, Россия)

Классификация текстов является одной из основных задач компьютерной лингвистики, поскольку к ней сводится ряд других задач: определение тематической принадлежности текстов, автора текста, эмоциональной окраски высказываний и др. Для обеспечения информационной и общественной безопасности большое значение имеет анализ в телекоммуникационных сетях контента, содержащего противоправную информацию (в том числе данные, связанные с терроризмом, наркоторговлей, подготовкой протестных движений или массовых беспорядков).

Данная статья представляет собой обзор методов классификации текстов, целями которого являются сравнение современных методов решения задачи классификации текстов, обнаружение тенденций развития данного направления, а также выбор наилучших алгоритмов для применения в исследовательских и коммерческих задачах.

Широко известный современный подход к классификации основывается на методах машинного обучения. В данной статье описываются наиболее распространенные алгоритмы построения классификаторов, проводимые с ними эксперименты и результаты этих экспериментов. Обзор подготовлен на основе выполненных за 2011–2016 гг. научных работ, находящихся в открытом доступе в сети Интернет и опубликованных в авторитетных журналах или в трудах международных конференций, высоко оцениваемых научным сообществом.

В статье произведены анализ и сравнение качества работы различных методов классификации по таким характеристикам, как точность, полнота, время работы алгоритма, возможность работы алгоритма в инкрементном режиме, количество предварительной информации, необходимой для классификации, независимость от языка.

Ключевые слова: *классификация текстов, анализ текстовой информации, обработка данных, машинное обучение, нейронные сети, качество классификации.*

Прогресс в области микроэлектроники и информационных технологий обусловил широкое распространение обработки в реальном времени больших потоков данных. Например, многие простые операции повседневной жизни, такие как использование кредитной карты или телефона, требуют автоматизированного создания, анализа и обработки различных данных. Поскольку эти операции часто выполняются большим числом участников, необходимы распределенные и массовые потоки данных. Точно так же социальные сети содержат большое количество специфических сетевых и текстовых потоков данных. Поэтому актуальна проблема создания моделей и алгоритмов, позволяющих эффективно обрабатывать большие потоки данных, особенно в условиях ограниченных временных и других ресурсов.

Для обеспечения информационной и общественной безопасности важное значение имеет анализ в телекоммуникационных сетях контента, содержащего противоправную информацию (в том числе данных, связанных с терроризмом, наркоторговлей, сетевым экстремизмом, подготовкой протестных движений или массовых беспорядков).

Целями данного обзора являются сравнение современных методов решения задачи классификации текстов, обнаружение тенденций развития данного направления, а также выбор наилучших алгоритмов для применения в исследовательских и коммерческих задачах.

Методы классификации текстов лежат на стыке двух областей – информационного поиска и машинного обучения. Их сходство состоит в способах представления самих документов и способах оценки качества алгоритмов. На сегодняшний день разработано большое количество методов и их различных вариаций для классификации текстов. Каждая группа методов имеет свои преимущества и недостатки, области применения, особенности и ограничения.

Особый интерес представляет случай, когда данные поступают в виде потока, например в телекоммуникационных сетях. Определенные трудности возникают из-за того, что обучение модели всегда основывается на совокупности свойств набора документов. Эти совокупные свойства могут изменяться с течением времени, и при построении потокового классификатора необходимо учитывать возможные изменения исходного распределения данных [1]. Желательно, чтобы выбранный метод мог поддерживать инкрементное обучение, то есть чтобы классификатор обучался на каждом отдельно взятом образце в режиме реального времени. При инкрементном обучении обучающие примеры поступают последовательно в процессе работы алгоритма, так что классификатор должен постоянно корректировать результаты обучения и дообучаться. При неинкрементном обучении вся обучающая выборка предоставляется сразу полностью. Ясно, что в случае инкрементного обучения поведение классификатора в процессе работы меняется, что уменьшает его предсказуемость и может осложнить настройку системы. В то же время инкрементное обучение делает систему гораздо более гибкой, адаптируемой к изменяющимся условиям.

Особенности процесса классификации в потоке связаны еще с тем, что не всегда удается контролировать скорость поступления данных. Некоторые классы документов могут встречаться в потоке только время от времени. Обнаружить этот редкий класс бывает непросто, и классификация текстов в таких случаях становится чрезвычайно сложной задачей.

Сравнение методов построения классификаторов является довольно сложной задачей по причине того, что разные входные данные могут приводить к различным результатам. Поэтому необходимо осуществить их программную реализацию и вычисление эффективности на одинаковых наборах документов для обучения и тестирования.

Формальная постановка задачи классификации текстов

Следует отличать классификацию от кластеризации. При классификации документов категории определены заранее, при кластеризации они не заданы и даже информация об их количестве может отсутствовать.

Формально постановку задачи классификации можно записать следующим образом.

Имеются множество документов $D = \{d_1, \ldots, d_{|D|}\}$ и множество возможных категорий (классов) $C = \{c_1, \ldots, c_{|C|}\}$. Неизвестная целевая функция Ф: $D \times C \to \{0, 1\}$ задается формулой

$$\Phi\left(d_j, c_i\right) = \begin{cases} 0, \text{ если } d_j \notin c_i, \\ 1, \text{ если } d_j \in c_i. \end{cases} \quad (1)$$

Требуется построить классификатор Ф′, максимально близкий к Ф.

В такой постановке задачи следует отметить, что о категориях и документах нет никакой дополнительной информации, кроме той, которую можно извлечь из самого документа.

Если классификатор выдает точный ответ:

$$\Phi': D \times C \to \{0, 1\}, \quad (2)$$

то классификация называется точной.

Если классификатор определяет степень подобия (Categorization Status Value) документа:

$$CSV: D \to [0, 1], \quad (3)$$

то классификация называется пороговой.

В общем случае процесс обучения с учителем (обучение по прецедентам, supervised learning) заключается в следующем. Системе предъявляется набор примеров, связанных с какой-либо заранее неизвестной закономерностью. Этот набор иногда называют обучающей выборкой L. Ее используют для обучения классификатора и определения значения его параметров, при которых классификатор выдает лучший результат. Далее в системе вырабатываются решающие правила, с помощью которых происходит разделение множества примеров на заданные классы. Качество разделения проверяется тестовой выборкой примеров T. При этом необходимо, чтобы выполнялись условия

$$L \cap T = \varnothing, \quad (4)$$

$$\Omega = L \cup T \subset C \times D. \quad (5)$$

Для множества примеров Ω известны значения целевой функции Ф.

Если в задаче каждому документу $d \in D$ может соответствовать только одна категория $c \in C$, то имеет место однозначная классификация, а если произвольное количество категорий, то многозначная классификация.

Частным случаем однозначной классификации является бинарная классификация, когда коллекцию документов нужно разбить на две непересекающиеся категории. Например, задача определения тональности высказываний (положительная или отрицательная окраска) или задача обнаружения спама (является сообщение спамом или нет) решается при помощи бинарного классификатора.

Решение задачи классификации состоит из четырех последовательных этапов:

– предобработка и индексация документов;
– уменьшение размерности пространства признаков;
– построение и обучение классификатора с помощью методов машинного обучения;
– оценка качества классификации.

При выборе конкретного алгоритма классификации следует учитывать особенности каждого из них. По-прежнему остается нерешенным вопрос определения набора классифицирующих признаков, их количества и способов вычисления весов. В алгоритмах глубокого обучения точность классификации сильно зависит от наличия обучающей выборки подходящего размера. Подготовка такой выборки – очень трудоемкий процесс. До сих пор остается также открытой проблема подбора параметров некоторых алгоритмов на этапе обучения.

Далее подробно рассмотрен каждый из этапов, описаны различные алгоритмы построения классификаторов, проводимые с ними эксперименты и результаты этих экспериментов.

Описание методов классификации

На рисунке 1 представлена общая схема процесса классификации. Рассмотрим каждый из его этапов.

Предобработка и индексация документов. Предварительная обработка текста включает в себя токенизацию, удаление функциональных слов (семантически нейтральных слов, таких как союзы, предлоги, артикли и пр.). Далее осуществляется морфологический анализ (производятся разметка по частям речи и стемматизация). Это позволяет значительно сократить размерность пространства. В результате в качестве признаков документа выступают все значимые слова, встречающиеся в документе.

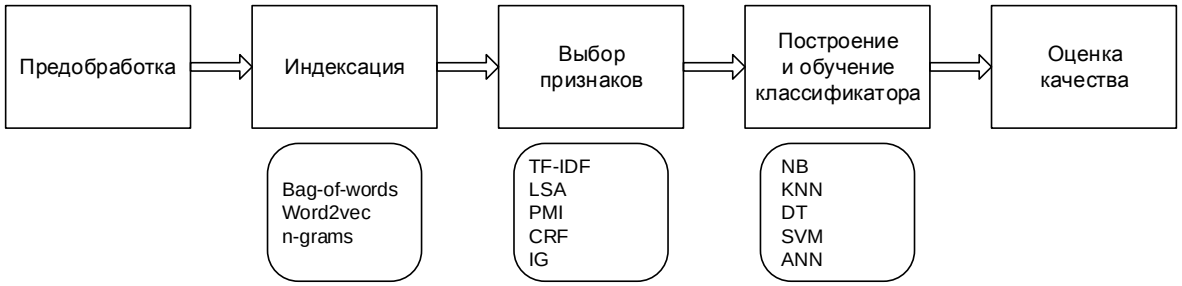

Рис. 1. Этапы процесса автоматической классификации текстов

Fig. 1. The steps of text automatic classification process

Индексация документов – это построение некоторой числовой модели текста, которая переводит текст в удобное для дальнейшей обработки представление.

Например, модель «мешка слов» (bag-of-words) позволяет представить документ в виде многомерного вектора слов и их весов в документе [2]. Другими словами, каждый документ – это вектор в многомерном пространстве, координаты которого соответствуют номерам слов, а значения координат – значениям весов.

Другая распространенная модель индексации – Word2vec [3]. Она представляет каждое слово в виде вектора, который содержит информацию о контекстных (сопутствующих) словах.

Еще одна модель индексации основана на учете *n*-грамм [2], то есть последовательностей из соседних символов.

Очевидно, что для обучающих и тестовых документов должен применяться один и тот же метод индексации.

Уменьшение размерности пространства признаков. Вычислительная сложность различных методов классификации напрямую зависит от размерности пространства признаков. Поэтому для эффективной работы классификатора часто прибегают к сокращению числа используемых признаков (терминов).

За счет уменьшения размерности пространства терминов можно снизить эффект переобучения – явление, при котором классификатор ориентируется на случайные или ошибочные характеристики обучающих данных, а не на важные и значимые. Переобученный классификатор хорошо работает на тех экземплярах, на которых он обучался, и значительно хуже на тестовых данных. Чтобы избежать переобучения, количество обучающих примеров должно быть соразмерно числу используемых терминов. В некоторых случаях сокращение размерности пространства признаков в 10 раз (и даже в 100) может приводить лишь к незначительному ухудшению работы классификатора.

Существуют несколько способов определения веса признаков документа. Наиболее распространенный – вычисление функции TF-IDF [2, 4, 5]. Его основная идея состоит в том, чтобы больший вес получали слова с высокой частотой в пределах конкретного документа и с низкой частотой употреблений в других документах.

Вычисляется частота термина TF (term frequency) – оценка важности слова в пределах одного документа d по формуле

$$TF = n_{t,d} / n_d, \quad (6)$$

где $n_{t,d}$ – количество употреблений слова t в документе d; n_d – общее число слов в документе d.

Обратная частота документа IDF (inverse document frequency) – инверсия частоты, с которой слово встречается в документах коллекции. IDF уменьшает вес общеупотребительных слов по формуле

$$IDF = \log(|D| / D_t), \quad (7)$$

где $|D|$ – общее количество документов в коллекции; D_t – количество всех документов, в которых встречается слово t.

Итоговый вес термина в документе относительно всей коллекции документов вычисляется по формуле

$$V_{t,d} = TF \cdot IDF. \quad (8)$$

Следует отметить, что по формуле (8) оценивается значимость термина только с точки зрения частоты вхождения в документ, без учета порядка следования терминов в документе и их лексической сочетаемости.

Для уменьшения размерности пространства терминов также применяют латентно-семантический анализ (LSA), использующий сингулярное разложение матриц [3, 6], поточечную взаимную информацию (PMI) [6, 7] (разновидность ассоциативной меры), условные случайные поля (CRF) [8] (обобщение скрытой марковской модели). Встречаются исследования [4, 9], в которых применяются статистические критерии и относительная энтропия для вероятностных распределений, называемая коэффициентом усиления информации, или дивергенцией Кульбака–Лейблера.

Построение и обучение классификатора с помощью методов машинного обучения. Можно выделить следующие методы классификации:

- вероятностные (например NB [4, 6]);
- метрические (например KNN [9]);
- логические (например DT [6, 10]);

– линейные (например SVM [4, 5, 6, 9]; логистическая регрессия [2, 8, 10]);

– методы на основе искусственных нейронных сетей (например FFBP [4, 10], RNN [8], DAN2 [9], CNN [2]).

Далее обобщенно описываются эти методы, указываются преимущества и недостатки каждого из них.

Метод Байеса (Naive Bayes, NB) относится к вероятностным методам классификации.

Пусть $P(c_i|d)$ – вероятность того, что документ, представленный вектором $d = (t_1, \ldots, t_n)$, соответствует категории c_i для $i = 1, \ldots, |C|$. Задача классификатора заключается в том, чтобы подобрать такие значения c_i и d, при которых значение вероятности $P(c_i|d)$ будет максимальным:

$$CSV(d) = \arg\max_{c_i \in C} P(c_i \mid d). \qquad (9)$$

Для вычисления значений $P(c_i|d)$ пользуются теоремой Байеса:

$$P(c_i \mid d) = \frac{P(c_i)P(d \mid c_i)}{P(d)}, \qquad (10)$$

где $P(c_i)$ – априорная вероятность того, что документ отнесен к категории c_i; $P(d \mid c_i)$ – вероятность найти документ, представленный вектором $d = (t_1, \ldots, t_n)$, в категории c_i; $P(d)$ – вероятность того, что произвольно взятый документ можно представить в виде вектора признаков $d = (t_1, \ldots, t_n)$.

По сути $P(c_i)$ является отношением количества документов из обучающей выборки L, отнесенных в категорию c_i, к количеству всех документов из L.

$P(d)$ не зависит от категории c_i, а значения $t_1, \ldots, t_n$ заданы заранее, поэтому знаменатель – это константа, не влияющая на выбор наибольшего из значений $P(c_i|d)$.

Вычисление $P(d \mid c_i)$ затруднительно из-за большого количества признаков $t_1, \ldots, t_n$, поэтому делают «наивное» предположение о том, что любые две координаты, рассматриваемые как случайные величины, статистически не зависят друг от друга. Тогда можно воспользоваться формулой

$$P(d \mid c_i) = \prod_{k=1}^{n} P(t_k \mid c_i). \qquad (11)$$

Далее все вероятности подсчитываются по методу максимального правдоподобия.

Преимущества метода:

– высокая скорость работы;

– поддержка инкрементного обучения;

– относительно простая программная реализация алгоритма;

– легкая интерпретируемость результатов работы алгоритма.

Недостатки метода: относительно низкое качество классификации и неспособность учитывать зависимость результата классификации от сочетания признаков.

Метод k ближайших соседей (k Nearest Neighbors, KNN) относится к метрическим методам классификации. Чтобы найти категорию, соответствующую документу d, классификатор сравнивает d со всеми документами из обучающей выборки L, то есть для каждого $d_z \in L$ вычисляется расстояние $\rho(d_z, d)$. Далее из обучающей выборки выбираются k документов, ближайших к d. Согласно методу k ближайших соседей, документ d считается принадлежащим тому классу, который является наиболее распространенным среди соседей данного документа, то есть для каждого класса c_i вычисляется функция ранжирования:

$$CSV(d) = \sum_{d_z \in L_k(d)} \rho(d_z, d) \cdot \Phi(d_z, c_i), \qquad (12)$$

где $L_k(d)$ – ближайшие k документов из L к d; $\Phi(d_z, c_i)$ – известные величины, уже расклассифицированные по категориям документы обучающей выборки.

Преимущества метода:

– возможность обновления обучающей выборки без переобучения классификатора;

– устойчивость алгоритма к аномальным выбросам в исходных данных;

– относительно простая программная реализация алгоритма;

– легкая интерпретируемость результатов работы алгоритма;

– хорошее обучение в случае с линейно неразделимыми выборками.

Недостатки метода:

– репрезентативность набора данных, используемого для алгоритма;

– высокая зависимость результатов классификации от выбранной метрики;

– большая длительность работы из-за необходимости полного перебора обучающей выборки;

– невозможность решения задач большой размерности по количеству классов и документов.

Метод деревьев решений (Decision Trees, DT) относится к логическим методам классификации.

Деревом решений называют ациклический граф, по которому производится классификация объектов (в нашем случае текстовых документов), описанных набором признаков. Каждый узел дерева содержит условие ветвления по одному из признаков. У каждого узла столько ветвлений, сколько значений имеет выбранный признак. В процессе классификации осуществляются последовательные переходы от одного узла к другому в соответствии со значениями признаков объекта. Классификация считается завершенной, когда достигнут один из листьев (конечных узлов) дерева. Значение этого листа определит класс, которому принадлежит рассматриваемый объект. На практике обычно используют бинарные деревья решений, в которых принятие решения перехода по ребрам осуществляется простой проверкой наличия признака в документе. Если значение признака

меньше определенного значения, выбирается одна ветвь, если больше или равно, другая.

В отличие от остальных подходов, представленных ранее, подход, использующий деревья решений, относится к символьным (то есть нечисловым) алгоритмам.

Алгоритм построения бинарного дерева решений состоит из следующих шагов.

Создается первый узел дерева, в который входят все документы, представленные всеми имеющимися признаками. Размер вектора признаков для каждого документа равен n, так как $d = (t_1, \ldots, t_n)$.

Для текущего узла дерева выбираются наиболее подходящий признак t_k и его наилучшее пограничное значение v_k.

На основе пограничного значения выбранного признака производится разделение обучающей выборки на две части. Далее выбранный признак не включается в описание фрагментов в этих частях, то есть фрагменты в частях представляются вектором с размерностью $n - 1$.

Образовавшиеся подмножества обрабатываются аналогично до тех пор, пока в каждом из них не останутся документы только одного класса или признаки для различения документов.

Когда говорят о выборе наиболее подходящего признака, как правило, подразумевают частотный признак, то есть любой признак текста, допускающий возможность нахождения частоты его появления в тексте. Лучшим для разделения является признак, дающий максимальную на данном шаге информацию о категориях. Таким признаком для текста может являться, например, ключевое слово. С этой точки зрения любой частотный признак можно считать переменной. Тогда выбор между двумя наиболее подходящими признаками сводится к оценке степени связанности двух переменных. Поэтому для выбора подходящего признака на практике применяют различные критерии проверки гипотез, то есть критерии количественной оценки степени связанности двух переменных, поставленных во взаимное соответствие, где 0 соответствует полной независимости переменных, а 1 – их максимальной зависимости.

Для исследования связи между двумя переменными удобно использовать представление совместного распределения этих переменных в виде таблицы сопряженности (факторной таблицы, или матрицы частот появления признаков). Она является наиболее универсальным средством изучения статистических связей, так как в ней могут быть представлены переменные с любым уровнем измерения. Таблицы сопряженности часто используются для проверки гипотезы о наличии связи между двумя признаками при помощи различных статистических критериев: критерия Фишера (точного теста Фишера), критерия согласия Пирсона (критерия хи-квадрат), критерия Крамера, критерия Стьюдента (t-критерия Стьюдента) и пр.

Преимущества метода:

– относительно простая программная реализация алгоритма;

– легкая интерпретируемость результатов работы алгоритма.

Недостатки метода: неустойчивость алгоритма по отношению к выбросам в исходных данных и большой объем данных для получения точных результатов.

Метод опорных векторов (Support Vector Machine, SVM) является линейным методом классификации. В настоящее время этот метод считается одним из лучших. Рассмотрим множество документов, которые необходимо расклассифицировать. Сопоставим ему множество точек в пространстве размерности $|D|$.

Выборку точек называют линейно разделимой, если принадлежащие разным классам точки можно разделить с помощью гиперплоскости (в двухмерном случае гиперплоскостью является прямая линия). Очевидный способ решения задачи в таком случае – провести прямую так, чтобы по одну сторону от нее лежали все точки одного класса, а по другую – все точки другого класса. Тогда для классификации неизвестных точек достаточно будет посмотреть, с какой стороны прямой они окажутся.

В общем случае можно провести бесконечное множество гиперплоскостей (прямых), удовлетворяющих нашему условию. Ясно, что лучше всего выбрать прямую, максимально удаленную от имеющихся точек. В методе опорных векторов расстоянием между прямой и множеством точек считается расстояние между прямой и ближайшей к ней точкой из множества. Именно такое расстояние и максимизируется в данном методе. Гиперплоскость, максимизирующая расстояние до двух параллельных гиперплоскостей, называется разделяющей (на рисунке 2 обозначена буквой L). Ближайшие к параллельным гиперплоскостям точки называются опорными векторами (рис. 2), через них проходят пунктирные линии. Другими словами, алгоритм работает в предположении, что, чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора, так как максимизация зазора между классами способствует более уверенной классификации.

На практике структура данных зачастую бывает неизвестна и очень редко удается построить разделяющую гиперплоскость, а значит, невозможно гарантировать линейную разделимость выборки. Могут существовать такие документы, которые алгоритм отнесет к одному классу, а в действительности они должны относиться к противоположному. Такие данные называются выбросами, они создают погрешность метода, поэтому было бы лучше их игнорировать. В этом заключается суть проблемы линейной неразделимости.