УДК 004.8

Ле Мань Ха

Московский физико-технический институт (государственный университет)

Оптимизация алгоритма KNN для классификации текстов

Рассмотрены два подхода повышения быстродействия алгоритма KNN для классификации текстов: уменьшение количества потенциальных ближайших соседей и использование двоичной кучи при поиске K ближайших соседей.

Ключевые слова: KNN, классификация текстов, двоичная куча

Le Manh Ha

Moscow Institute of Physics and Technology (State University)

Optimization of the algorithm KNN for text classification

In this paper, the main purpose is to consider two ways of optimization of the algorithm KNN for text classification: decrease the number of potential nearest neighbours and use binary heap in searching K nearest neighbours.

Key words: KNN, text classification, binary heap

1. Введение

KNN – K ближайших соседей – один из самых используемых методов классификации. Основным принципом метода K ближайших соседей является то, что объект относится к тому классу, которому принадлежат больше всего его ближайших соседей. Другими словами, для объекта рассмотрим его K ближайших соседей, дальше для каждого класса считаем, сколько из этих K соседей принадлежат данному классу, объект относится к классу с наибольшим количеством соседей [1].

KNN широко используется для классификации текстов. Для начала считаем координаты текстов в пространстве. Размер пространства есть количество терминов в корпусе (объем словаря). Считая TF-IDF для всех текстов в корпусе, получаем представления текста в виде векторов, каждый компонент вектора – «важность» соответствующего слова для данного текста. Координаты текстов используются для решения различных задач, в том числе классификация [2].

Один из недостатков KNN – медленная скорость, для классификации нового текста нужно вычислять расстояния между этим текстом со всеми текстами в корпусе, а их количество может быть миллионы. Для уменьшения количества вычислительных операций будем сравнивать только тексты, имеющие общие термины, другими словами, нет смысла сравнивать тексты, которые не имеют никаких связей. Кроме того, использование двоичной кучи тоже помогает повысить быстродействие алгоритма [3].

2. Оптимизация алгоритма KNN

Подход. Уменьшение количества потенциальных ближайших соседей

Для каждого термина храним id текстов, содержащих этот термин. Для быстрого поиска id текста сортируем координаты этих текстов по этому термину по возрастанию. Итак, для каждого термина t есть список (id, x), где x – значение координаты по данному термину: $X(t) = [(id_1, x_1), (id_2, x_2)...(id_m, x_m)]$, где $x_1 <= x_2... <= x_m$.

Три шага оптимизированного алгоритма KNN для классификации нового текста:

Шаг 1. Вычисляем координаты (TF-IDF) этого текста.

Шаг 2. Для каждого термина $(t,\ x)$ ищем ближайшее к x значение на $X(t) = [(id_1, x_1), (id_2, x_2)...(id_m, x_m)]$, используя метод двоичного поиска [3].

Шаг 2-1. $i = 1;\ j = m$.

Шаг 2-2. Пока $i < j$.

Шаг 2-2-1. $k = (i + j)\text{div}2$.

Шаг 2-2-1. Если $x_k < x$, то $i = k + 1$.

Шаг 2-2-2. Если $x_k >= x$, то $j = k$.

После шага 2-2 x_k есть ближайшее значение к x на $X(t)$.

Шаг 3. Рассмотрим 2*K текстов вокруг (id_k, x_k) на $X(t)$, считая расстояние между текстами, проведем сравнение и обновление K ближайших соседей данного текста.

Итак, сложность оптимизированного алгоритма KNN: O(K*log(N)*|L|), а KNN: O(N*|L|), где N – количество текстов в корпусе, |L| – средняя длина текста в корпусе.

Очевидно, что оптимизированный алгоритм KNN имеет преимущество по скорости работы при больших корпусах текстов.

Подход 2. Использование двоичной кучи

Двоичная куча – это двоичное дерево, для которого выполнены 3 условия [3].

1. Значение в любой вершине не меньше (или не больше), чем значения её потомков.

2. Глубина листьев (расстояние до корня) отличается не более чем на 1 слой.

3. Последний слой заполняется слева направо.

Для каждого текста будем хранить его ближайшие соседи в двоичной куче, корень которой есть самый ближайший сосед.

Процедура поиска K ближайших соседей текста t_0.

Шаг 1. Создание пустой двоичной кучи.

Шаг 2. При рассмотрении текста t_i, если размер кучи меньше, чем K, то просто добавляем t_i в кучу, а если размер кучи равен K и расстояние между t_i и t_0 меньше, чем расстояние между корнем кучи и t_0, то обменяем корень кучи с t_i и восстанавливаем свойства кучи.

Сложность процедуры поиска K ближайших соседей одного текста с использованием двоичной кучи: O(log(K)), а без него: O(K).

3. Тестирование

Было проведено тестирование оптимизированного алгоритма KNN на корпусе Wikinews на английском и русском языках и сравнивание его скорости, а также точности по сравнению с другими алгоритмами классификации текстов. Для оценки алгоритмов был использован метод K-Fold Cross-validation с параметром K = 10 [4].

Скорость и точность алгоритма KNN до и после оптимизации

Корпус и алгоритм		Время тестирования (с)	Точность (%)
English	KNN	48 257	87.3
	Оптимизированный KNN	13 795	88.9
Russian	KNN	9168	72.5
	Оптимизированный KNN	4866	71.7

Из таблицы видно, что оптимизированный KNN работает существенно быстрее, чем KNN, а точность классификации почти совпадает.

Литература

1. *Manning C.D., Raghavan P.* Hinrich Schutze An Introduction to Information Retriveval. Cambridge University Press, 2009.
2. *Jurafsky D., Martin J.H.* Speech and Language Processing. Prentice-Hall Inc., 2000.
3. *Cormen T.H., Leiserson C.E., Rivest R.L., Stein C.* Introduction to algorithms. The MIT Press, 2009.
4. Hastie, Tibshirani Cross-validation and bootstrap. SLDM III., 2009.

References

1. *Manning C.D., Raghavan P.* Hinrich Schutze An Introduction to Information Retriveval. Cambridge University Press, 2009.
2. *Jurafsky D., Martin J.H.* Speech and Language Processing. Prentice-Hall Inc., 2000.
3. *Cormen T.H., Leiserson C.E., Rivest R.L., Stein C.* Introduction to algorithms. The MIT Press, 2009.
4. Hastie, Tibshirani Cross-validation and bootstrap. SLDM III., 2009.

Поступила в редакцию 31.12.2015