Наука *и* Образование
МГТУ им. Н.Э. Баумана
Сетевое научное издание
ISSN 1994-0408

Ссылка на статью:
// Наука и Образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2015. № 6. С. 4–19.
DOI: **10.7463/0815.9328000**
Дата: 4.06.2016

УДК 004.852

Графовая модификация метрических алгоритмов классификации

Васнецов А. Г.[1,*], Самарев Р. С.[1]
[*]vasnetsov93@gmail.com
[1]МГТУ им. Н.Э. Баумана, Москва, Россия

В статье рассматривается вопрос модификации метрических алгоритмов классификации, в частности — алгоритма k-Nearest Neighbours, для его применения к последовательностям данных. Предлагается метод обобщения метрических алгоритмов классификации, в рамках которого разработан алгоритм для решения задачи классификации и разметки последовательностей связанных данных. Рассматриваются преимущества разработанного метода классификации по сравнению с существующими. Приводится сравнение эффективности предлагаемого алгоритма с алгоритмом CRF в задаче распознавания частей речи на открытом наборе данных CoNLL2000.

Ключевые слова: машинное обучение; классификация последовательностей; алгоритм k-Nearest Neighbours

Введение

Классификация и разметка (labeling) последовательностей имеет широкий спектр практических применений. В исследованиях генома, классификации белковых последовательностей в существующие категории используется, чтобы узнать функции нового белка [1, 2]. Исследования последовательности событий обращения к Интернет-ресурсам может позволить отличить пользователя-человека от поискового робота [3]. Классификация и разметка последовательностей находят своё применение и в анализе естественных языков (Natural Language Processing, NLP) [4, 5].

В общем случае, последовательность - это упорядоченный список событий $S =< s_1, s_2, \ldots, s_n >$, где с событием s_i, в общем случае, могут быть связаны любые данные. Однако в данной работе будем предполагать, что события представлены в виде набора признаков фиксированной длинны, в котором признак может быть представлен строкой, действительным или целым числом, логической переменной или переменной перечислимого типа. Каждый элемент последовательности, помимо прочего, может иметь метку, в этом случае говорят о размеченных последовательностях.

Пусть L- множество меток, тогда на размеченной последовательности задана функция отображения событий $Label : s_i \to l, l \in L$. Помимо метки элементов последовательности, сама последовательность также может иметь класс: $Class : s_i \to c, c \in Cl$ где Cl - множество возможных классов.

Таким образом, при работе с последовательностями, возникает две задачи: классификация последовательностей (sequence classification), — то есть восстановление неизвестной функции *Class*, и разметка последовательностей (sequence labeling), то есть восстановление неизвестной функции T. В литературе [6] этот вид классификации также называется сильной классификацией последовательностей (strong sequence classification). В обоих задачах предполагается наличие обучающего множества - набора правильно классифицированных или размеченных последовательностей.

Для решения задачи классификации было предложено множество различных методов, которые можно разбить на следующие группы:

- Метод, основанный на наборах признаков фиксированной длины. Этот способ требует преобразования последовательности в набор признаков, и это преобразование играет основную роль в классификации.
- Метод на основе оценки расстояния между последовательностями. Примером такого расстояния может служить расстояние Левенштейна[7].
- Классификация с использованием статистических моделей.

Для решения сильной задачи классификации, однако, подходит лишь последняя группа методов, среди которых отметим[6]:

- Условные случайные поля (Condition Random Fields, CRF).
- Скрытые марковские цепи.
- Марковские SVM[8].
- Рекуррентные нейронные сети (RNN).

Данные методы основываются на статистическом моделировании каких-либо признаков элементов последовательности и предсказании их меток на основе этих моделей. В некоторых случаях такой подход может оказаться неэффективным, так как, например, несмотря на то, что метки признаков имеют разные категориальные значения, они имеют разную степень “близости”. Например буквы ‘К’ и ‘Г’ в некотором смысле (по типу звучания) ближе друг к другу чем буквы ‘А’ и ‘Р’[9], а слова “красный” и “синий” ближе друг к другу, чем слова “алгебра” и “табурет”. Такую “близость” часто удобно задавать с помощью функции расстояния. Для использования в процессе классификации такой функции расстояния необходимо использовать другой тип алгоритмов, – так называемые **метрические алгоритмы классификации**, например алгоритм k-ближайших соседей (k-Nearest Neighbours, kNN).

Правильность определения метки элемента часто зависит от контекста, в котором этот элемент встретился. Так например, значение слова часто зависит от контекста, в котором оно употребляется. Метрические алгоритмы же в качестве входных данных

принимают вектор признаков фиксированной длинны и не имеют внутреннего состояния, поэтому их прямое применение к задаче разметки последовательности мало эффективно.

В статье предлагается способ модификации метрических алгоритмов классификации, позволяющий алгоритму учитывать контекст элемента последовательности, тем самым улучшая точность классификации.

Обобщение алгоритмов классификации для последовательностей

Для обоснования возможности модификации метрических алгоритмов классификации рассмотрим взаимосвязи между существующими алгоритмами классификации, и методы, применённые к ним для обеспечения возможности работы с последовательностями. Одним из перспективных направлений решения задачи сильной классификации являются рекуррентные нейронные сети (Recurrent neural network, RNN), которые отличаются от обычных нейронных сетей тем, что в них имеется обратная связь. При этом под обратной связью подразумевается связь от логически более удалённого элемента к менее удалённому. Наличие обратных связей позволяет запоминать и воспроизводить целые последовательности реакций на один стимул. В связи с тем, что алгоритм kNN не имеет никакого внутреннего состояния, использовать метод обобщения, эквивалентный применённому в RNN не представляется возможным.

На рисунке 1 приведена диаграмма взаимосвязи между алгоритмами HMM, CRF, Naive Bayes и Logit Regression[10]. Из диаграммы следует, что алгоритм HMM является обобщением наивного байесовского классификатора на последовательности входных данных. Действительно, в процессе работы алгоритма HMM для каждой отдельной вершины, фактически, применяется алгоритм Байесовской классификации со своим набором исходных данных. Для классификации в этом случае используется алгоритм Витерби, находящий последовательность переходов между вершинами графа, максимизирующую вероятность появления последовательности.

На основе рассмотрения существующих алгоритмов классификации последовательностей предложим метод обобщения метрических алгоритмов классификации для работы с последовательностями данных. Метод включает в себя алгоритм для построения графовой модели, выбор оптимизируемой функции, модифицированный алгоритм Витерби. Рассмотрим предлагаемый метод на примере алгоритма kNN.

Алгоритм kNN в процессе своей работы опирается на гипотезу о компактности: близкие объекты, как правило, лежат в одном классе, где под близкими понимаются объекты, имеющие наименьшее и всех прочих расстояние в некоторой, заранее определённой метрике. Эта гипотеза естественным образом обобщается на задачу классификации последовательностей.

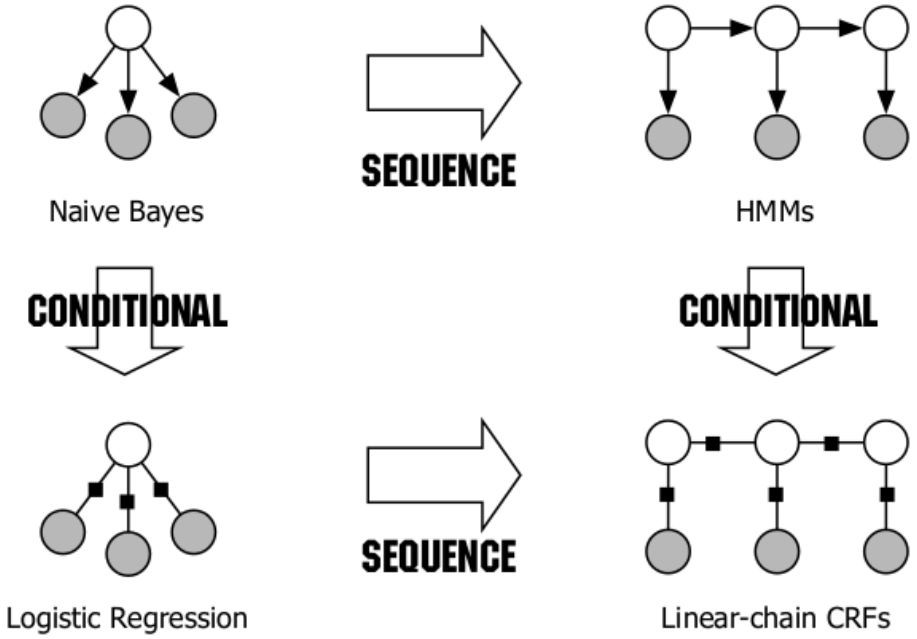

Рис. 1. Диаграмма взаимосвязи между алгоритмами классификации.

Гипотеза о компактности последовательностей: каждый элемент последовательности, как правило, принадлежит тому классу, который минимизирует суммарное расстояние всех элементов данной последовательности до известных элементов соответствующего класса.

Таким образом, алгоритм классификации последовательностей, являющийся обобщением kNN, будет заключаться в поиске последовательности классов, минимизирующих суммарное расстояние каждого элемента классифицируемой последовательности до ближайшего элемента соответствующего класса обучающей выборки. Так как переходы между классами, в соответствии с обучающей выборкой, образуют некоторую графовую структуру, то назовём описанный алгоритм Structured k-Nearest Neighbours (SkNN). Взаимосвязь алгоритмов отражена на рисунке 2.

Обозначения в Structured k-Nearest Neighbours

Для рассмотрения модели SkNN введём следующие обозначения:

1. k - количество рассматриваемых ближайших элементов.
2. L - множество возможных меток классов элементов последовательности.
3. Cl - множество возможных классов последовательностей.
4. S - множество элементов последовательностей.
5. X - множество последовательностей обучающей выборки. $X = \{(s)_n : n \in \mathbb{N}, s \in S\}$
6. $G = (V, E)$ - граф, описывающий структуру классификатора.

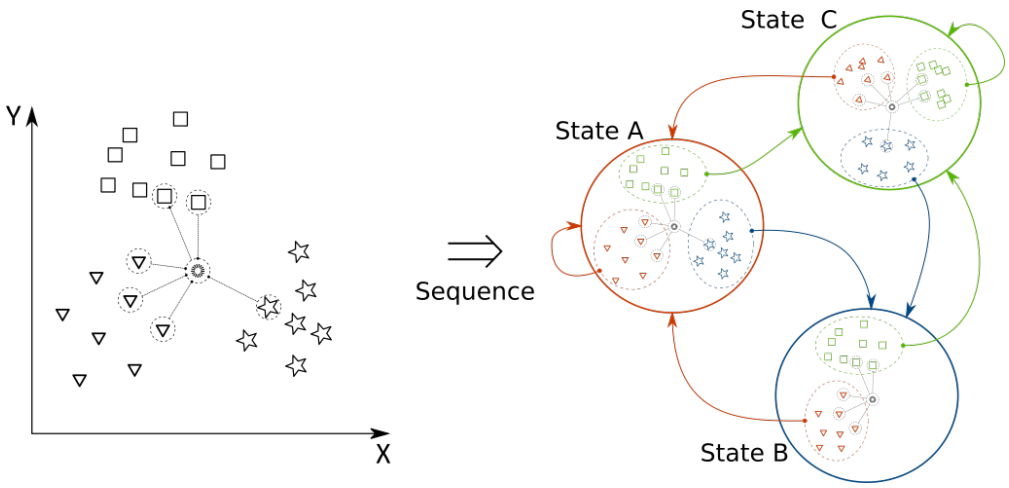

Рис. 2. Structured k-Nearest Neighbours

7. $f_{vl} : V \mapsto L$ - функция, определяющая соответствие вершины метке. Функция является инъективной.
8. $f_{sl} : S \mapsto L$ - функция, определяющая то, какую метку класса имеет элемент обучающей выборки.
9. $f_{sv} : S \mapsto V$ - функция, определяющая то, какой вершине принадлежит элемент обучающей выборки. Все первые элементы последовательностей принадлежат V_{init}.
10. $C \in V$ - текущее состояние автомата. В момент времени $\tau = 0, C = V_{init}$.

В основе алгоритма SkNN лежит ориентированный граф $G = (V, E)$, каждая вершина которого соответствует либо началу последовательности (условно назовём такую вершину $V_{init} \in V$), либо возможному классу элемента последовательности $V_l, l \in L$, либо концу последовательности $V_{end} \in V$.

При этом должны выполняться следующие условия:

Условие 1. Ребро $E_{inti,x}$ присутствует в графе G тогда и только тогда, когда в обучающей выборке имеются 2 следующих друг за другом элемента $\alpha \in S$ и $\beta \in S$, таких что классы элементов $\alpha : L_\alpha = x$ и $\beta : L_\beta = y$.

Условие 2. Ребро $E_{inti,x}$ присутствует тогда и только тогда, когда в обучающей выборке присутствует последовательность, первый элемент которой имеет класс x.

Условие 3. Ребро $E_{x,end}$ присутствует тогда и только тогда, когда в обучающей выборке присутствует последовательность, последний элемент которой имеет класс x.

Построение модели SkNN из обучающей выборки выполняется алгоритмом 1.

Для разметки последовательностей с использованием алгоритма SkNN можно использовать модифицированный алгоритм Витерби, который отличается от оригинального алгоритма тем, что минимизирует суммарное расстояние. Приведём модифицированный алгоритм Витерби 2.

Функция $n\_dist(inst, X, k)$ находит в наборе X k ближайших в некоторой метрике элемент к $inst$ и возвращает средневзвешенное расстояние до них.

Исходные параметры: Обучающая выборка X

Результат: Построение модели SkNN.

for $seq \in Dataset$ **do**
  $C = V_{init}$
  for $inst \in seq, inst \in S$ **do**
    $l := f_{sl}(inst)$
    $L.add(l)$
    $f_{sv}(inst) := C;$
    $newC := f_{vl}(l);$
    $E.add(E_{C,newC});$
    $C := newC;$
  end
  $E.add(E_{C,V_{end}});$
end

Алгоритм 1: Построение модели SkNN из обучающей выборки.

Типы признаков в SkNN

В текущей реализации модели, в качестве входных данных для алгоритма SkNN можно использовать последовательности элементов, каждый из которых состоит из некоторого фиксированного числа признаков.

В общем случае, требования к типу и составу признаков элемента накладываются исключительно функций расстояния (метрикой), которая должна быть способна определить расстояние любыми двумя элементами.

Это является отличительной чертой и преимуществом алгоритма SkNN, так как остальные алгоритмы, такие как HMM, CRF, StructuredSVM, RNN способны работать лишь с описанием элемента в виде кортежа признаков числовых и перечислимых типов.

Для SkNN, однако, возможно использование признаков произвольного типа, при этом семантика признака не теряется, но может быть представлена в виде функции расстояния. Так например возможно ввести метрику, работающую с элементами онтологии. В этом случае функция расстояния может возвращать, например, минимальный путь от одного концепта до другого согласно онтологии. С помощью взвешенного суммирования разных признаков возможно создать метрику, одновременно учитывающую признаки произвольного типа, например признаков, полученных из онтологии и вектора весов слов, полученных с помощью алгоритма word2vec [11].

Исходные параметры: Модель SkNN, классифицируемая последовательность $sequence$

Результат: Классификация последовательности.

for $v_i \in V$ **do**
  $T1[i,1] \leftarrow 0$
  $T2[i,1] \leftarrow V_{init}$
end
for $inst_i \in sequence$ **do**
  for $v_j \in V$ **do**
    $T1[j,i] \leftarrow \min\limits_{k}(T1[k,i-1] + n\_dist(inst_i, inst : f_{sv}(inst) = v_j, K))$
    $T2[j,i] \leftarrow \operatorname{argmin}\limits_{k}(T1[k,i-1] + n\_dist(inst_i, inst : f_{sv}(inst) = v_j, K))$
  end
end
$T \leftarrow sizeof(sequence)$
$Z_T \leftarrow \operatorname{argmin}\limits_{k}(T1[k,T])$
$y_i \in Y$
for $i \leftarrow \{T, T-1, ..., 2\}$ **do**
  $Z_{i-1} \leftarrow T2[z_i, i]$
  $y_{i-1} \leftarrow V_{z_{i-1}}$
end
return Y

Алгоритм 2: Модифицированный алгоритм Витерби.

Получение графовой структуры с помощью кластеризации

В большом количестве существующих задач классификации последовательности требуется не классифицировать каждый отдельных элемент последовательности, а всю последовательность целиком.

В этом случае, обычно, в исходных данных не присутствуют сведения о классе каждого отдельного элемента и алгоритм SkNN сводится к обычному алгоритму kNN, выполняемому для каждого элемента отдельно.

Очевидно, такой подход не даёт сколько-нибудь приемлемых результатов, поэтому необходимо автоматически выделить структуру в исходных данных.

Так как в основе SkNN лежит сравнение элементов в некоторой метрике, то становится возможным использовать кластеризацию для автоматического выделения графовой структуры в последовательностях. Для получения такой структуры можно выполнить следующие действия:

- Пусть дана начальная структура, в которой каждый класс последовательностей представлен единственной вершиной. С вершиной ассоциированы элементы последовательностей, входящих в соответствующий класс.
- Для каждой вершины проведём кластеризацию относящихся к ней элементов последовательностей.
- Каждую кластеризованную вершину разделим на несколько вершин, таким образом, что каждая новая вершина соответствует одному кластеру старой вершины и содержит в себе элементы соответствующего кластера. При этом, в граф добавляются ребра в соответствии с условием 1.

Стоит заметить, что такой подход не обязательно использовать исключительно совместно с SkNN, однако, в связи с тем, что входом для алгоритмов кластеризации является матрица расстояний между объектами (или функция расстояния / схожести), то алгоритм SkNN, использующий ту же функцию расстояния, будет лучше других «совместим» с полученной кластеризацией.

Классификация последовательностей

Как было сказано выше, многие реальные задачи требуют в качестве решения не классификацию отдельных элементов последовательности, а классификацию последовательности в целом.

Алгоритм формирования SkNN может быть применен и в этом случае без дополнительных модификаций. Для этого достаточно выполнить следующие действия:

1. Для последовательностей каждого класса (типа) применить алгоритм кластеризации и получить графовую структуру.
2. Объединить полученные на предыдущем этапе графы в один, для этого:
   (a) Добавить вершины init(старт) и end(конец).
   (b) Добавить ребра из вершины init к начальным вершинам графов, полученных на первом этапе, и ребра из конечных вершин к вершине end.

В результате получим граф, состоящий из не смежных подграфов, общий вид которого представлен на рисунке 3.

Применяя полученный граф в алгоритме SkNN для классификации последовательностей, в силу того, что в графе не существует пути из подграфа одного класса в подграф другого класса, все получаемые метки класса будут принадлежать одному подграфу, а следовательно и одному исходному классу последовательностей. Таким образом, для классификации последовательностей достаточно определить какому из исходных классов принадлежат полученные метки.

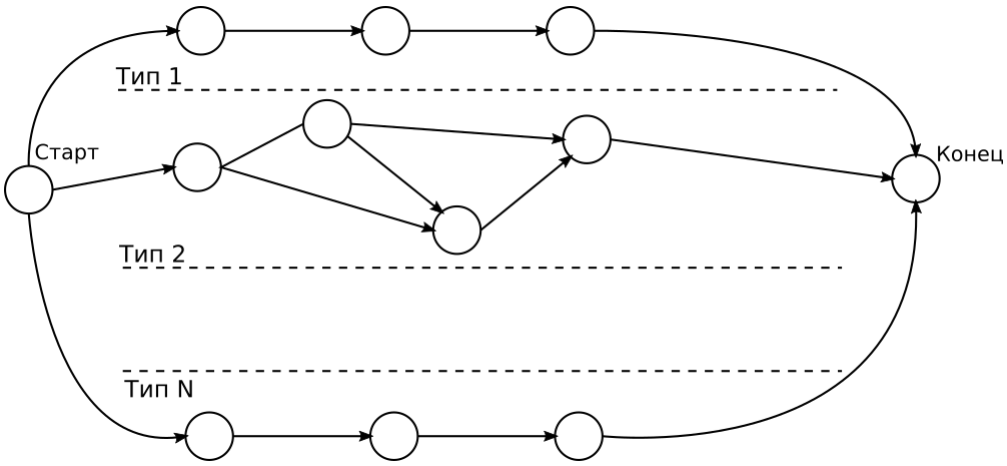

Рис. 3. Общий вид графа, применяемого для классификации последовательностей.

Эксперименты

Для проверки работы алгоритма и его сравнения с существующими популярными на практике алгоритмами, была проведена серия экспериментов с использованием открытых наборов исходных данных[12]. Отметим, что область возможного применения алгоритма SkNN достаточна велика: в качестве эталонных наборов данных использовались наборы из таких областей как распознавание рукописных символов (набор UJI Pen Characters[13]) и обработки текстов на естественном языке (набор CoNLL200 [14]).

Распознавание рукописных символов

Для первого эксперимента был выбран набор данных, представляющих собой рукописные символы. Этот набор данных был выбран, так как содержит данные простейшего типа - последовательность двумерных координат на плоскости. Простота работы с такими данными позволяет легко их визуализировать, что упрощает отладку алгоритма. Для испытания работы алгоритма была составлена обучающая выборка, состоящая из 200 последовательностей рукописного написания цифр 8 разных авторов. Тестовая выборка состояла из 40 последовательностей 3 других авторов.

Для эксперимента использовалось нормализованное Евклидово расстояние, как простейший пример метрики.

Эксперимент показал точность алгоритма SkNN равную 92.5%, что существенно превышает случайное угадывание (10% в данном случае).

Стоит заметить, что точность алгоритма SkNN хоть и не превышает точности алгоритмов, специализированных для решения именно этой задачи [15], однако показывает результат сравнимый с ними.

Распознавание групп слов в предложении (chunking)

Второй экспериментальный набор **CoNLL200** данных представляет собой последовательности слов в предложениях на английском языке с соответствующими частями речи, полученными (POS-tag) с помощью программы Brill tagger[16], и меткой группы (chunk tag), полученной из корпуса WSJ[17]. Метки групп представляют собой такие части предложения как глагольные группы и т.п.

В проведённом эксперименте точность алгоритма SkNN сравнивалась с точностью алгоритма CRF, реализованного в программной утилите CRF++. Этот алгоритм является одним из наиболее точных и популярных алгоритмов на сегодняшний день в области анализа текстов на естественном языке. Параметры набора данных: размер обучающего корпуса — 9 тыс. предложений, 220 тыс. слов, размер тестовой выборки — 400 предложений, 10 тыс. слов. В качестве гиперпараметра для обоих алгоритмов выступает размер окна контекста, который добавляется к текущему элементу последовательности. Точность алгоритмов сравнивалась с одинаковыми значениями этого гиперпараметра.

Другим параметром для алгоритма SkNN является используемая для вычисления расстояния метрика. Эксперимент производился с использованием метрик, наиболее успешно применяющихся на практике в области анализа текста на естественном языке [18]. Среди используемых метрик:

1. Modified Value Difference Metric (MVDM) [9].
2. Пересечение наборов (Overlap).
3. Взвешенное пересечение (Weighted Overlap).
   (a) С помощью Information Gain[19].
   (b) С помощью Information Gain Ratio[20].

Результаты экспериментов приведены в таблице 1.

Таблица 1. Результаты экспериментов

Размер контекста	**Алгоритм**	**Метрика**	**Достигнутая точность**
Только текущей элемент	SkNN	MVDM	0.71
	SkNN	Overlap	0.7
	SkNN	IG Weighted Overlap	0.76
	CRF	-	0.85
Окно размером 2 до и после текущего элемента.	SkNN	Overlap	0.91
	SkNN	IG WeightedOverlap	0.6
	SkNN	IGRatio Weighted Overlap	0.88
	SkNN	**MVDM**	**0.93**
	CRF	-	0.87

Заключение

В статье исследована возможность модификации метрических алгоритмов классификации на примере алгоритма k-Nearest Neighbours для их применения к последовательностям. Научная новизна заключается в том, что предложен новый алгоритм классификации последовательностей SkNN и продемонстрирована его эффективность в задачах классификации и разметки последовательностей в области распознавания и обработки текстов на естественном языке. Сравнение алгоритма с существующими методами показывает, что при данной точности алгоритма целесообразно продолжать работу в направлении эффективной реализации и, например, реализации данного алгоритма для вычислительных кластеров, что позволит существенно увеличить скорость его работы.

Список литературы

1. Kocsor A. et al. Application of compression-based distance measures to protein sequence classification: a methodological study // Bioinformatics. – 2006. – Т. 22. – №. 4. – С. 407-412.
2. Sonego P., Kocsor A., Pongor S. ROC analysis: applications to the classification of biological sequences and 3D structures // Briefings in bioinformatics. – 2008. – Т. 9. – №. 3. – С. 198-209.
3. Feily M., Shahrestani A., Ramadass S. A survey of botnet and botnet detection // Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on. – IEEE, 2009. – С. 268-273.

4. Zhou G. D., Su J. Named entity recognition using an HMM-based chunk tagger // proceedings of the 40th Annual Meeting on Association for Computational Linguistics. – Association for Computational Linguistics, 2002. – С. 473-480.
5. Lafferty J., McCallum A., Pereira F. C. N Conditional random fields: Probabilistic models for segmenting and labeling sequence data. – 2001.
6. Nguyen N., Guo Y. Comparisons of sequence labeling algorithms and extensions // Proceedings of the 24th international conference on Machine learning. – ACM, 2007. – С. 681-688.
7. Левенштейн В. И. Двоичные коды с исправлением выпадений и вставок символа 1 // Проблемы передачи информации. – 1965. – Т. 1. – №. 1. – С. 12-25.
8. Altun Y. et al. Hidden markov support vector machines //ICML. – 2003. – Т. 3. – С. 3-10.
9. Liu F. et al. A modified value difference metric kernel for context-dependent classification tasks // Machine Learning and Cybernetics, 2006 International Conference on. – IEEE, 2006. – С. 3432-3437.
10. Sutton C., McCallum A. An introduction to conditional random fields for relational learning // Introduction to statistical relational learning. – 2006. – С. 93-128.
11. Mikolov T. et al. Distributed representations of words and phrases and their compositionality // Advances in neural information processing systems. – 2013. – С. 3111-3119.
12. Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
13. Llorens D. et al. The UJIpenchars Database: a Pen-Based Database of Isolated Handwritten Characters //LREC. – 2008.
14. Tjong Kim Sang E. F., Buchholz S. Introduction to the CoNLL-2000 shared task: Chunking // Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7. – Association for Computational Linguistics, 2000. – С. 127-132.
15. Ramos-Garijo R. et al. An input panel and recognition engine for on-line handwritten text recognition // FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS. – 2007. – Т. 163. – С. 223.
16. Acedański S. A morphosyntactic Brill tagger for inflectional languages // Advances in Natural Language Processing. – Springer Berlin Heidelberg, 2010. – С. 3-14.
17. Paul D. B., Baker J. M. The design for the Wall Street Journal-based CSR corpus // Proceedings of the workshop on Speech and Natural Language. – Association for Computational Linguistics, 1992. – С. 357-362.

18. Daelemans W., Van den Bosch A. Memory-based language processing. – Cambridge University Press, 2005.
19. Kent J. T. Information gain and a general measure of correlation //Biometrika. – 1983. – Т. 70. – №. 1. – С. 163-173.
20. Mori T. Information gain ratio as term weight: the case of summarization of ir results // Proceedings of the 19th international conference on Computational linguistics-Volume 1. – Association for Computational Linguistics, 2002. – С. 1-7.

Science & Education
of the Bauman MSTU
Electronic journal
ISSN 1994-0408

Science and Education of the Bauman MSTU, 2015, no. 6, pp. 4–19.
DOI: **10.7463/0815.9328000**
Date: 4.06.2016

Graphical modification of metric classification algorithms

Vasnetsov A. G.[1,*], **Samarev R. S.**[1] [*]vasnetsov93@gmail.com

[1]Bauman Moscow State Technical University, Russia

Keywords: machine learning; sequence labeling; k-Nearest Neighbours algorithm

This article deals with the question of modifying metric classification algorithms, in particular - k-Nearest Neighbours algorithm for ability to apply it to sequential data. Method of generalisation of metric algorithms is proposed. As part of method, algorithms for sequence labeling and classification are developed. Advantages and disadvantages of developed algorithm regarding to the existing ones is shown. comparison of effectivity of CRF and developed algorithm is shown in task of sentence chunking using open dataset CoNLL2000.

References

1. Kocsor A. et al. Application of compression-based distance measures to protein sequence classification: a methodological study // Bioinformatics. – 2006. – Vol. 22. – # 4. – pp. 407-412.
2. Sonego P., Kocsor A., Pongor S. ROC analysis: applications to the classification of biological sequences and 3D structures // Briefings in bioinformatics. – 2008. – Vol. 9. – # 3. – pp. 198-209.
3. Feily M., Shahrestani A., Ramadass S. A survey of botnet and botnet detection // Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on. – IEEE, 2009. – pp. 268-273.
4. Zhou G. D., Su J. Named entity recognition using an HMM-based chunk tagger // proceedings of the 40th Annual Meeting on Association for Computational Linguistics. – Association for Computational Linguistics, 2002. – pp. 473-480.
5. Lafferty J., McCallum A., Pereira F. C. N Conditional random fields: Probabilistic models for segmenting and labeling sequence data. – 2001.

6. Nguyen N., Guo Y. Comparisons of sequence labeling algorithms and extensions // Proceedings of the 24th international conference on Machine learning. – ACM, 2007. – pp. 681-688.

7. Levenstein V. I. Binary codes with correction of dropping out and appearance of symbols // Ploblems of information transmission. – 1965. – Vol. 1. – # 1. – pp. 12-25.

8. Altun Y. et al. Hidden markov support vector machines //ICML. – 2003. – Vol. 3. – pp. 3-10.

9. Liu F. et al. A modified value difference metric kernel for context-dependent classification tasks // Machine Learning and Cybernetics, 2006 International Conference on. – IEEE, 2006. – pp. 3432-3437.

10. Sutton C., McCallum A. An introduction to conditional random fields for relational learning // Introduction to statistical relational learning. – 2006. – pp. 93-128.

11. Mikolov T. et al. Distributed representations of words and phrases and their compositionality // Advances in neural information processing systems. – 2013. – pp. 3111-3119.

12. Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

13. Llorens D. et al. The UJIpenchars Database: a Pen-Based Database of Isolated Handwritten Characters //LREC. – 2008.

14. Tjong Kim Sang E. F., Buchholz S. Introduction to the CoNLL-2000 shared task: Chunking // Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7. – Association for Computational Linguistics, 2000. – pp. 127-132.

15. Ramos-Garijo R. et al. An input panel and recognition engine for on-line handwritten text recognition // FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS. – 2007. – Vol. 163. – pp. 223.

16. Acedański S. A morphosyntactic Brill tagger for inflectional languages // Advances in Natural Language Processing. – Springer Berlin Heidelberg, 2010. – pp. 3-14.

17. Paul D. B., Baker J. M. The design for the Wall Street Journal-based CSR corpus // Proceedings of the workshop on Speech and Natural Language. – Association for Computational Linguistics, 1992. – pp. 357-362.

18. Daelemans W., Van den Bosch A. Memory-based language processing. – Cambridge University Press, 2005.

19. Kent J. T. Information gain and a general measure of correlation //Biometrika. – 1983. – Vol. 70. – # 1. – pp. 163-173.

20. Mori T. Information gain ratio as term weight: the case of summarization of ir results // Proceedings of the 19th international conference on Computational linguistics-Volume 1. – Association for Computational Linguistics, 2002. – pp. 1-7.