УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
май–июнь 2017 Том 17 № 3 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
May–June 2017 Vol. 17 No 3 ISSN 2226-1494 http://ntv.ifmo.ru/en

УДК 004.85

ПОСТРОЕНИЯ НАБОРОВ ДАННЫХ ДЛЯ ЗАДАЧИ БИНАРНОЙ КЛАССИФИКАЦИИ ПО ИХ ХАРАКТЕРИСТИЧЕСКОМУ ОПИСАНИЮ

А.С. Забашта[a], А.А. Фильченков[a]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
Адрес для переписки: azabashta@corp.ifmo.ru

Информация о статье
Поступила в редакцию 30.03.17, принята к печати 28.04.17
doi: 10.17586/2226-1494-2017-17-3-498-505
Язык статьи – русский

Ссылка для цитирования: Забашта А.С., Фильченков А.А. Построения наборов данных для задачи бинарной классификации по их характеристическому описанию // Научно-технический вестник информационных технологий, механики и оптики. 2017. Т. 17. № 3. С. 498–505. doi: 10.17586/2226-1494-2017-17-3-498-505

Аннотация

Предмет исследования. Представлен метод построения экземпляров данных для задачи классификации по заданному характеристическому описанию в виде вектора мета-признаков для задачи классификации. Предложен наивный метод для решения той же задачи, используемый в качестве референтного. Исследовано характеристическое пространство экземпляров задач классификации, а также методы обхода этого пространства. **Метод.** Предложенный метод основывается на генетическом алгоритме, где в качестве минимизируемой целевой функции используется расстояние в характеристическом пространстве от вектора описания построенного экземпляра задачи классификации до заданного. Для работы генетического алгоритма разработаны операторы кроссовера и мутации экземпляров задачи классификации, использующие операции добавления или удаления признаков и объектов. **Основные результаты.** Для проверки предложенного метода выбраны нетривиальные двухмерные мета-признаковые пространства, построенные над статистическими, информационно-теоретическими и структурными характеристиками экземпляров задачи. Для сравнения использован наивный метод, не учитывающий характеристического описания. При равных ограничениях в 6500 рассмотренных экземпляров задачи классификации предложенный в работе метод обошел наивный на всех тестах. Погрешность уменьшена в среднем в 30 раз. **Практическая значимость.** Предложенный метод построения наборов данных для задачи классификации по их характеристическому описанию позволяет получить неизвестные экземпляры задачи классификации, которые нужны для оценки работы классификаторов в определенных областях мета-признакового пространства при построении систем автоматического выбора алгоритмов.

Ключевые слова
машинное обучение, мета-обучение, задача классификации, эволюционные вычисления, генетический алгоритм

Благодарности
Работа выполнена при финансовой поддержке Правительства Российской Федерации, грант 074-U01 и РФФИ, грант 16-37-60115-мол_а_дк.

GENERATING DATASETS FOR THE BINARY CLASSIFICATION TASK BASED ON THEIR CHARACTERISTIC DESCRIPTIONS

A.S. Zabashta[a], A.A. Filchenkov[a]

[a] ITMO University, Saint Petersburg, 197101, Russian Federation
Corresponding author: azabashta@corp.ifmo.ru

Article info
Received 30.03.17, accepted 28.04.17
doi: 10.17586/2226-1494-2017-17-3-498-505
Article in Russian

For citation: Zabashta A.S., Filchenkov A.A. Generating datasets for the binary classification task based on their characteristic descriptions. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2017, vol. 17, no. 3, pp. 498–505 (in Russian). doi: 10.17586/2226-1494-2017-17-3-498-505

Abstract

Subject of Study. We present a method for generating instances of the binary classification task based on their characteristic descriptions in the form of a meta-feature vector. We propose a naïve method for the same problem solution to be used as a referral one. We study the characteristic space of the binary classification task instances, as well as the methods for this space traversal. **Method**. The proposed method is based on genetic algorithm, where the distance in the characteristic space from the description vector of the generated instance for the binary classification task to the specified one is used as the minimized objective function. We developed the crossover and mutation operators for the genetic algorithm. These operators are based

on such transformations as addition or removal of features and objects from datasets. **Main Results.** In order to validate the proposed method, we chose several non-trivial two-dimensional meta-feature spaces that were generated from statistical, information-theoretical and structural characteristics of classification task instance. We used the baseline method to evaluate the relative error of the proposed method. Both methods used the same number of classification tasks instances. The proposed method outperformed the naïve method and reduced average error by 30 times. **Practical Relevance**. The proposed method for generating instances for classification task based on their characteristic description allows obtaining unknown instances that are required to evaluate the performance of classifiers in certain areas of the meta-features space for design of automatic algorithm selection systems.

Keywords
machine learning, meta-learning, classification problem, evolutionary computation, genetic algorithm

Acknowledgements
This work was financially supported by the Government of the Russian Federation, Grant 074-U01, and the Russian Foundation for Basic Research, Grant 16-37-60115 mol_a_dk.

Введение

Под **задачей** будем понимать класс определенного вида заданий, каждое из которых является **экземпляром** этой задачи. Например, сформулированная в общем виде задача коммивояжера является в нашей терминологии задачей, а конкретный граф с заданными на ребрах весами – ее экземпляром. Таким образом, экземпляры задачи являются входными данными для алгоритмов, которые решают соответствующую задачу.

Анализ некоторых алгоритмов основан на изучении их работы на различных экземплярах задач, чтобы определить модель поведения и возможные ограничения алгоритма для дальнейшего его улучшения. Для проведения такого анализа требуется обширный набор экземпляров конкретной задачи. Однако во многих случаях число реальных экземпляров ограничено, и мы не можем быть уверены в их разнообразии. Для решения этой проблемы требуется разработка методов генерации новых экземпляров.

Исторически сложилось так, что в первых работах, посвященных генерации экземпляров, внимание уделялось лишь трудным случаям, на которых конкретный алгоритм работает хуже всего. Данные экземпляры требовались для выявления недостатков алгоритма. Изначально методы генерации трудных экземпляров были предложены для задачи выполнимости булевых формул [1–4], а позже и для других задач, таких как оптимизация [5], задача нахождения кратчайшего вектора [6], или даже для точных алгоритмов [7, 8]. В других работах предлагалось генерировать не только самые трудные, но и самые простые экземпляры задач [9].

Алгоритмы, перечисленные выше, опираются на синтез данных с экстремальными значениями качества работы алгоритмов. Альтернативой является генерация данных с определенными характеристиками. Первым исследованием в рамках этой постановки является работа [10], которая появилась так же давно, как и исследования общей проблемы генерации. Тем не менее, достаточный импульс для развития данной области был получен лишь после применения принципов мета-обучения в работах Смит-Майлз и соавторов [11, 12].

Мета-обучение [13, 14] решает задачу автоматического выбора алгоритма для данного экземпляра. Оно сводит ее к задаче предсказания, использующего свойства экземпляра, называемые мета-признаками. Так как мета-признаки используются для предсказания эффективности алгоритмов на этом экземпляре, они имеют важное значение при анализе их производительности и, более широко, поведения. В то время как Смит-Майлз и ее соавторы следуют начальной постановке и используют мета-признаки для более описательных целей, мы фокусируемся на задаче генерации экземпляров для заданного мета-признакового описания. Это поможет исследователям обнаружить влияние одного мета-признака или их подмножества на производительность алгоритма.

Классификация является одной из наиболее популярных задач машинного обучения. В настоящее время исследователи разработали множество различных алгоритмов классификации, но среди них нет и не может быть одного универсального алгоритма [15, 16]. Для любого алгоритма, не учитывающего свойства экземпляра решаемой задачи, найдется экземпляр, на котором этот алгоритм уступит любому другому. Таким образом, относительная точность алгоритмов классификации зависит от самих данных.

Насколько нам известно, не существует какого-либо исследования по синтезу экземпляров для задачи классификации. Можно указать две возможные причины их отсутствия. В первую очередь, экземпляр задачи классификации гораздо сложнее синтезировать по сравнению с экземплярами задач выполнимости и оптимизации. Во-вторых, в сообществе исследователей распространено мнение, что имеется достаточное число наборов данных, которое, однако, рассеивается при взгляде на проблему с точки зрения мета-обучения.

Целью настоящей работы является разработка метода синтеза наборов данных для задачи классификации по их мета-признаковым описаниям. Полученный метод должен быть в состоянии заполнить определенные области пространства экземпляров. В данной работе эта проблема решается за счет сведения ее к задаче оптимизации, которая, в свою очередь, решается с помощью эволюционных вычислений.

Задача классификации

Задача классификации относится к классу задач обучения с учителем [17, 18]. Чаще всего экземпляр задачи классификации представляется в виде матрицы «объекты–признаки». Каждая ее строка – определенный объект, который описывается вектором признаков, характеризующих его свойства, и меткой класса, к которому он принадлежит. В данной работе в качестве признаков рассматриваются вещественные числа, а в качестве классов – булево множество {0; 1}. Следует отметить, что все модели классификаторов позволяют обрабатывать данные, представленные в таком виде.

Данные с известными классами подаются на вход классификатору, который должен распознать в них скрытую зависимость между признаками и меткой класса. Затем классификатор должен аппроксимировать эту зависимость, построив алгоритм определения класса объекта на основе его признакового описания.

Для исследования классификаторов в рамках мета-обучения требуется характеристическое описание наборов данных – вектор мета-признаков, которые описывают свойства всех возможных экземпляров задачи классификации. В качестве таких признаков используют размерность набора данных (число признаков, число представителей и число классов), статистические и теоретико-информационные метрики признаков и классов или состояние дерева принятия решений (статистика числа листьев, ветвей, глубины или ширины).

Описанные характеристики наборов данных можно использовать для определения областей компетенции алгоритмов классификации, т.е. тех областей, в которых эти алгоритмы превосходят все остальные. Идея мета-обучения состоит в том, что при правильном выборе пространства алгоритмы будут работать относительно похожим образом на близких экземплярах. Например, если один классификатор оказался лучше других на одном наборе данных, то он же будет лучше остальных на другом наборе, достаточно похожем на первый.

Описание предлагаемого подхода

Построение набора данных для классификации по его характеристическому описанию в данной работе решается как задача минимизации. Так как характеристическое описание является вещественным вектором, то и объекты, описываемые этими признаками, можно представить как точки в многомерном вещественном пространстве. Следовательно, для того чтобы построить экземпляр из интересующей нас области пространства, необходимо синтезировать набор данных, характеристическое описание которого, будучи представленным в виде точки, минимизирует расстояние до заданной точки.

Одним из методов поиска минимума являются эволюционные вычисления [19]. Основная идея заключается в том, чтобы, начиная с некоторой начальной популяции – множества наборов данных, в процессе эволюции искать и сохранять наилучшие объекты для генерации следующей популяции. В качестве метода такой минимизации в данной работе используется генетическое программирование [20].

Для работы генетического алгоритма требуется задать начальную популяцию, фитнесс-функцию (функцию приспособленности), а также операторы кроссовера и мутации. В качестве начальной популяции для генетического алгоритма использовались существующие наборы данных для задачи бинарной классификации. Для реализации оператора мутации и кроссовера требуются дополнительные операторы удаления и добавления признаков и объектов в набор данных. Как уже было отмечено ранее, экземпляр задачи классификации можно представить как матрицу – двухмерный набор данных, где каждая строка представляет объект, а колонка – признак. Таким образом, добавление нового объекта в набор данных можно представить как добавление новой строки в матрицу.

Предположим, что мы хотим добавить новый объект v класса t в набор данных D, все объекты которого, помимо метки класса, содержат n признаков. Обозначим за x_i значение i-го признака объекта x. Изначально значения всех признаков нового объекта v помечены особым значением «□» как пустые $\forall\, i \in \{1 \ldots n\}, v_i = \square$. Пусть D^t – подмножество D, содержащее объекты класса t. Пока не заданы значения всех признаков ($\exists\, i, v_i = \square$), повторяется следующая процедура:

1. выбирается случайное подмножество F из $\{1 \le i \le n \mid v_i = \square\}$ множества признаков, значения которых еще отсутствуют у объекта v;
2. выбирается случайный объект q из множества D^t;
3. $\forall\, i \in F, v_i \leftarrow q_i$ – копирование выбранных значений признаков F из q в v.

Предположим, что мы хотим добавить новый, (n+1)-й признак в D. Для этого создается случайная функция $f{:}D \to \mathbb{R}$ и применяется ко всем объектам $\forall\, x \in D, x_{n+1} = f(x)$. Эта функция строится рекурсивно. Если g и h – две другие уже построенные случайные функции, то f – случайная функция, построенная по одной из следующих стратегий, выбранной случайно:

– $f(x) = x_i$, где признак i выбирается равновероятно при создании f;
– $f(x) = -1$, если у объекта x отрицательный класс, иначе +1;
– $f(x) = c$, где c – определенная константа, выбранная из нормального распределения при создании f;

- $f(x) = r$, где r – случайная величина, выбранная из нормального распределения при каждом расчете f;
- $f(x) = |g(x)|$;
- $f(x) = \sin(g(x))$;
- $f(x) = g(x) + h(x)$;
- $f(x) = g(x) \cdot h(x)$.

В работе использовались случайные функции глубины не больше четырех. Если они выходили за этот предел, то на новом шаге их построения рассматривались только первые четыре нерекурсивные варианта построения.

Оператор мутации для генетического алгоритма должен принимать на вход набор данных и возвращать новый модифицированный набор с похожими свойствами. Для этого выбиралась одна из двух случайных стратегий: изменить в наборе число признаков или изменить число объектов. Затем выбиралось случайное число m из нормального распределения и округлялось до ближайшего целого, не равного нулю. Если значение было положительным, то в набор добавлялось $|m|$ признаков или объектов в зависимости от выбранной ранее стратегии, иначе $|m|$ признаков или объектов удалялись.

Оператор кроссовера должен принимать и возвращать два набора данных со свойствами, частично похожими на первый или второй набор. Пример кроссовера двух экземпляров задачи классификации приведен на рис. 1. Входные наборы данных A и B разбивались на множества объектов с положительными и отрицательными классами A^p, A^n, B^p и B^n (рис. 1, а). Наборы с равными классами уравнивались путем удаления или добавления объектов (рис. 1, б). Затем случайные пары объектов из множеств A^p и B^p естественно объединялись и образовывали новое множество C^p, а из A^n и B^n получалось C^n. После этого C^p и C^n объединялись в множество C (рис. 1, в). Для получения результата кроссовера множество объектов C делилось на два меньших набора D и E по случайному множеству признаков (рис. 1, г).

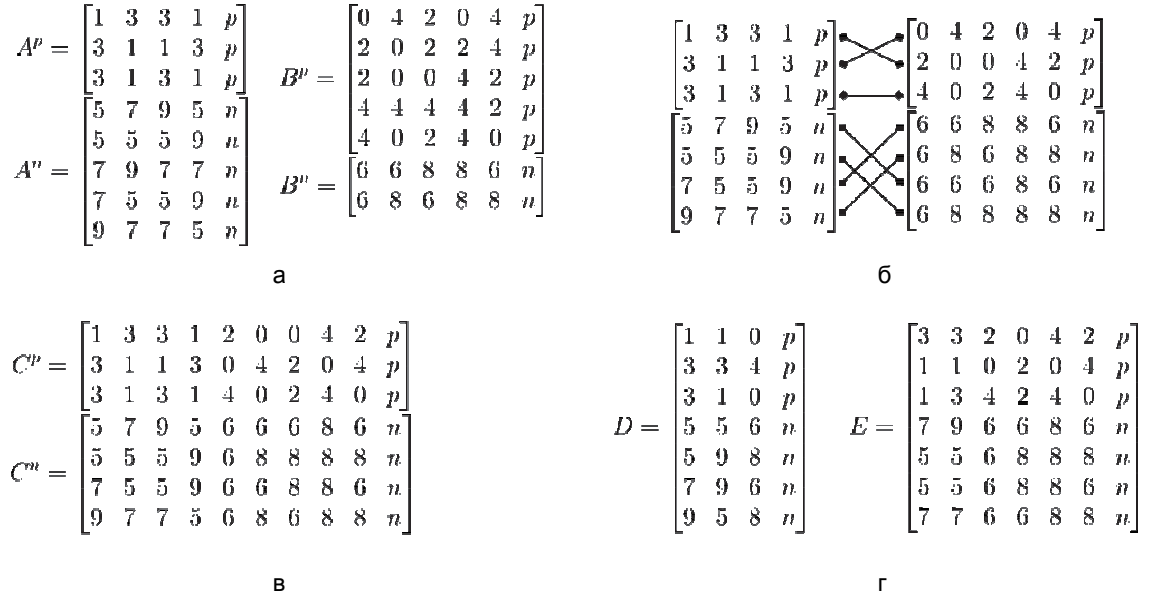

Рис. 1. Пример этапов кроссовера двух экземпляров задачи классификации: начальные наборы данных A и B, разбитые на классы (а); модифицированные наборы равных размеров с отмеченной группировкой на пары (б); объединенный экземпляр C (в); результат кроссовера (г). На первом шаге набор A^p остался неизменным, из A^n был удален третий, а из B^p – второй и четвертый объекты, в B^n были добавлены два объекта. На втором шаге модифицированные экземпляры объединялись в большой экземпляр C согласно разбиению на пары. На последнем шаге C разбился на D и E: первый, четвертый и шестой признаки перешли в D, а оставшиеся – в E

Используемые в описанных ранее операторах задачи заполнения и удаления признаков являются стандартными задачами машинного обучения, поэтому различные алгоритмы для них можно использовать для модификации этих операторов. Стоит отметить, что описанные выше операторы являются базовыми для алгоритмов оптимизации, основанных на эволюционных вычислениях. Например, мутацию можно применить в алгоритме имитации отжига [21].

Схема генетического алгоритма

В данной работе генетический алгоритм выполнял заданное число итераций. Каждая итерация состояла из нескольких шагов. Сначала для каждой особи x считалась фитнесс-функция – расстояние до искомой точки d_x. Затем среди полученных значений искался максимум $m = \max(d_x)$. Далее для каждой

особи x считался ее вес $w_x = m - d_x$, из которого путем нормирования получалось вероятностное распределение $p_x = w_x / \sum_y p_y$.

Полученное распределение использовалось в методе рулетки [22] для двух независимых отборов половины кандидатов для скрещивания и мутации. Таким образом, чем ближе текущая особь к искомой точке, тем больше у нее шанс участвовать в кроссовере или мутации. После этого все особи сортировались по значению фитнесс-функции, и из них удалялись худшие так, чтобы размер нового поколения был равен размеру поколения в начале итерации генетического алгоритма.

Описание эксперимента

Была проведена экспериментальная проверка работы предложенного метода заполнения характеристического пространства, который генерирует по характеристическому описанию максимально ему соответствующий набор данных для классификации. Очевидно, что достаточно несложно сгенерировать набор данных с простыми характеристиками, такими как его размерность. Исходя из этого, для экспериментов в качестве характеристического пространства использовалось двумерное пространство из нетривиальных признаков – информационно-статистических и параметров дерева принятия решения, которое построено для текущего набора данных – прообраза точки характеристического пространства.

Предварительно каждая координата характеристического пространства нормировалась своим стандартным отклонением и сдвигалась так, чтобы искомая точка оказалась в начале отсчета. В качестве искомой точки для каждого эксперимента бралась точка, соответствующая математическому ожиданию точек текущего пространства. После этого алгоритму оптимизации требовалось найти прообраз точки, который минимизирует норму вектора, отложенного от начала модифицированных координат до этой точки.

Для нормирования характеристического пространства, а также в качестве начальной популяции для генетического алгоритма использовались существующие наборы данных для задачи бинарной классификации с ресурса OpenML.org [23]. Для ускорения работы брались только наборы размером менее 200 килобайт. Всего было получено 640 наборов. На рис. 2, а, представлен пример распределения наборов в модифицированном пространстве, образованном из математического ожидания попарного коэффициента корреляции признаков и числа ветвей дерева принятия решений.

Характеристики наборов данных

Все используемые характеристики для удобства были разбиты на две группы.

Первая группа – статистико-информационные величины.

SI1. Средняя корреляция между всеми парами признаков.
SI2. Средний коэффициент асимметрии признаков.
SI3. Средний коэффициент эксцесса признаков.
SI4. Нормализованная энтропия класса.
SI5. Средняя энтропия признаков.
SI6. Средняя взаимная информация признаков и класса.
SI7. Максимум взаимной информации.
SI8. Число равных признаков.
SI9. Отношение сигнал-шум.
SI10. Среднее среднеквадратичное отклонение.
SI11. Средняя дисперсия.

Вторая группа характеристик – параметры деревьев принятия решения, построенных на наборе данных.

DT1. Среднее число использованных атрибутов.
DT2. Дисперсия числа использованных атрибутов.
DT3. Среднее число узлов.
DT4. Дисперсия числа узлов.
DT5. Средняя высота.
DT6. Дисперсия высоты.

Реализации перечисленных характеристик были взяты из работы [24], в которой показано, что эти характеристики обладают достаточной дескриптивностью для построения на их основе метаклассификаторов, предсказывающих лучшие алгоритмы для произвольных задач классификации.

Сравнение погрешности

Для оценки погрешности в настоящей работе используется расстояние в модифицированном характеристическом пространстве от искомой точки до ближайшей из найденных.

Одним из альтернативных методов заполнения пространства можно считать случайную генерацию наборов данных. Работа генетического алгоритма была ограничена числом новых наборов данных, которое равняется 6500. В связи с этим в альтернативном методе было рассмотрено столько же случайно сге-

нерированных наборов. Вначале случайно выбиралось число признаков и объектов с положительным и отрицательным классом из равномерного распределения на отрезке [1; 200]. Затем к полученному набору применялся оператор добавления признаков, пока их число было меньше выбранного. Пример заполнения случайной генерацией наборов описанного ранее пространства представлен на рис. 2, б.

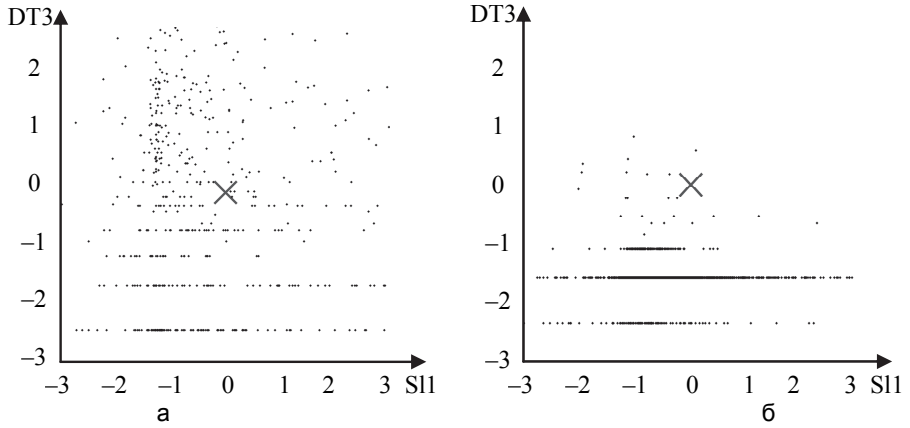

Рис. 2. Распределение наборов в модифицированном пространстве, образованном из перечисленных ранее характеристик SI1 и DT3: реальные наборы данных (а); случайно сгенерированные (б). Крестиком отмечен центр пространства – искомая точка

Результаты

На рис. 3 представлены распределения наборов данных из нескольких популяций генетического алгоритма. В таблице представлена погрешность работы различных подходов по поиску одной и той же точки.

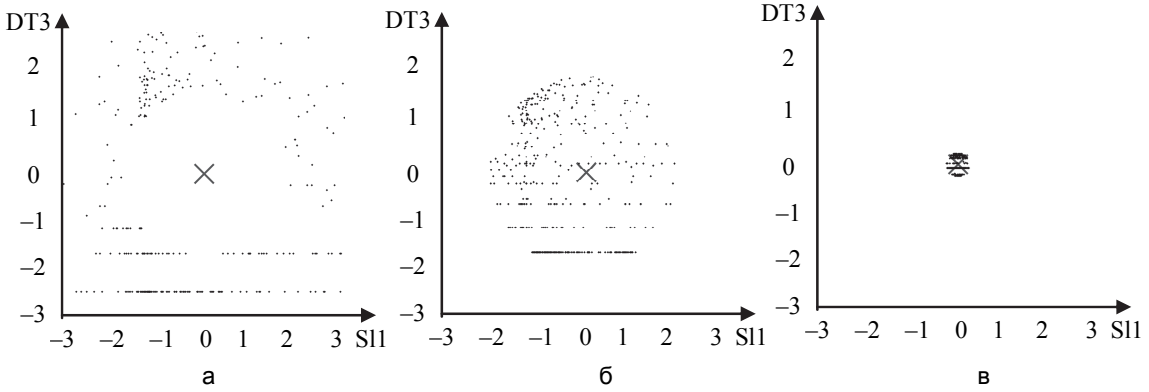

Рис. 3. Распределение наборов в модифицированном пространстве, образованном из перечисленных ранее характеристик SI1 и DT3, из разных популяций генетического алгоритма: начальная популяция (а); после первого шага алгоритма (б); конечная популяция (в). Крестиком отмечен центр пространства – искомая точка

	DT1		DT2		DT3		DT4		DT5		DT6	
	Naïve	Gen	Naïve	Gen	Naïve	Gen	Naïve	Gen	Naïve	Gen	Naïve	Gen
SI1	0,231	**0,010**	0,096	**0,008**	0,181	**0,015**	0,326	**0,007**	0,280	**0,002**	0,075	**0,003**
SI2	0,086	**0,002**	0,131	**0,007**	0,204	**0,005**	0,289	**0,001**	0,128	**0,011**	0,089	**0,007**
SI3	0,218	**0,004**	0,064	**0,007**	0,207	**0,017**	0,289	**0,001**	0,061	**0,012**	0,052	**0,008**
SI4	0,057	**0,004**	0,098	**0,007**	0,119	**0,012**	0,098	**0,007**	0,187	**0,002**	0,128	**0,004**
SI5	0,300	**0,004**	0,090	**0,010**	0,097	**0,026**	0,262	**0,011**	0,205	**0,004**	0,072	**0,009**
SI6	0,098	**0,003**	0,051	**0,002**	0,056	**0,012**	0,056	**0,008**	0,088	**0,002**	0,072	**0,003**
SI7	0,118	**0,005**	0,234	**0,012**	0,121	**0,021**	0,294	**0,010**	0,173	**0,002**	0,072	**0,004**
SI8	0,076	**0,007**	0,117	**0,014**	0,088	**0,019**	0,106	**0,004**	0,241	**0,015**	0,221	**0,026**
SI9	0,034	**0,007**	0,179	**0,011**	0,129	**0,010**	0,315	**0,006**	0,183	**0,016**	0,167	**0,011**
SI10	0,071	**0,043**	0,059	**0,022**	0,070	**0,024**	0,069	**0,016**	0,072	**0,002**	0,093	**0,044**
SI11	0,040	**0,039**	0,042	**0,039**	0,076	**0,039**	0,079	**0,039**	0,041	**0,040**	0,064	**0,039**

Таблица. Погрешность алгоритмов поиска заданной точки в различных пространствах, образованных из всех возможных пар характеристик первой и второй группы, для двух различных подходов: Naïve – наивный метод; Gen – предложенный метод на основе генетического алгоритма

Выводы

Как видно по результатам работы, алгоритму удалось найти достаточно близкую к заданному описанию точку и достаточно компактно заполнить пространство вокруг нее. Подход с использованием генетического алгоритма оказался гораздо лучше, чем простая генерация данных, которая не учитывает требуемое характеристическое описание. Простая генерация может строить либо простые модели, которые не дают необходимого разброса и охвата искомой области, либо слишком сложные модели, которые трудно подогнать под заданное описание.

Таким образом, благодаря предложенному в данной работе алгоритму задачу генерации данных для алгоритмов классификации по их описанию и заполнению пространства можно решить с достаточно маленькой погрешностью.

Также при использовании пространств большой размерности на минимизацию может повлиять проклятие размерности [25], в данном случае следует использовать многокритериальную оптимизацию, в которой отдельно минимизируется расстояние по каждой координате.

Заключение

В работе предложен алгоритм генерации экземпляров данных для задачи классификации по характеристическому описанию. Предложенный метод был протестирован на множестве нетривиальных метапризнаковых пространств и в каждом из тестов превзошел точность наивного метода, в среднем улучшив точность в 30 раз. Реализованный подход можно применять в любых сферах, связанных с характеристическим описанием данных, для повышения точности алгоритмов, работающих с ними. К таким сферам относятся задачи сравнения алгоритмов, а также предсказание наилучшего алгоритма.

Предложенный метод можно использовать с любыми вещественными характеристиками данных, но он легко может быть модифицирован под другие пространства с заданной метрикой. Как было сказано ранее, в алгоритме могут использоваться пространства любой размерности, а реализованные операторы могут применяться в других алгоритмах эволюционного вычисления.

Время работы алгоритма сводится к вычислению характеристического описания данных, все остальные операции выполняются на каждой итерации генетического алгоритма за время, пропорциональное размеру начального поколения. В случае, когда требуется найти всего один набор данных вместо заполнения целой области пространства, это будет являться недостатком.

Литература

1. Cook S.A., Mitchell D.G. Finding hard instances of the satisfiability problem // DIMACS Series in Discrete Mathematics and Theoretical Computer Science. 1997. V. 35. P. 1–17. doi: 10.1090/dimacs/035/01
2. Horie S., Watanabe O. Hard instance generation for SAT // Lecture Notes in Computer Science. 1997. V. 1350. P. 22–31. doi: 10.1007/3-540-63890-3_4
3. Selman B., Mitchell D.G., Levesque H.J. Generating hard satisfiability problems // Artificial Intelligence. 1996. V. 81. N 1-2. P. 17–29.
4. Xu K., Boussemart F., Hemery F., Lecoutre C. Random constraint satisfaction: easy generation of hard (satisfiable) instances // Artificial Intelligence. 2007. V. 171. N 8. P. 514–534. doi: 10.1016/j.artint.2007.04.001
5. van Hemert J.I. Evolving combinatorial problem instances that are difficult to solve // Evolutionary Computation. 2006. V. 14. N 4. P. 433–462. doi: 10.1162/evco.2006.14.4.433
6. Ajtai M. Generating hard instances of the short basis problem // Lecture Notes in Computer Science. 1999. V. 1644. P. 1–9.
7. Буздалов М.В. Генерация тестов для олимпиадных задач по программированию с использованием генетических алгоритмов // Научно-технический вестник информационных технологий, механики и оптики. 2011. № 2(72). С. 72–77.
8. Буздалов М.В. Генерация тестов для олимпиадных задач по теории графов с использованием эволюционных стратегий // Научно-технический вестник информационных технологий, механики и оптики. 2011. № 6(76). С. 123–127.
9. Smith-Miles K., Tan T.T. Measuring algorithm footprints in instance space // IEEE Congress on Evolutionary Computation. Brisbane, Australia, 2012. P. 3446–3453. doi: 10.1109/CEC.2012.6252992
10. Asahiro Y., Iwama K., Miyano E. Random generation of test instances with controlled attributes // DIMACS Series in Discrete Mathematics and Theoretical Computer Science. 1996.

References

1. Cook S.A., Mitchell D.G. Finding hard instances of the satisfiability problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 1997, vol. 35, pp. 1–17. doi: 10.1090/dimacs/035/01
2. Horie S., Watanabe O. Hard instance generation for SAT. *Lecture Notes in Computer Science*, 1997, vol. 1350, pp. 22–31. doi: 10.1007/3-540-63890-3_4
3. Selman B., Mitchell D.G., Levesque H.J. Generating hard satisfiability problems. *Artificial Intelligence*, 1996, vol. 81, no. 1-2, pp. 17–29.
4. Xu K., Boussemart F., Hemery F., Lecoutre C. Random constraint satisfaction: easy generation of hard (satisfiable) instances. *Artificial Intelligence*, 2007, vol. 171, no. 8, pp. 514–534. doi: 10.1016/j.artint.2007.04.001
5. van Hemert J.I. Evolving combinatorial problem instances that are difficult to solve. *Evolutionary Computation*, 2006, vol. 14, no. 4, pp. 433–462. doi: 10.1162/evco.2006.14.4.433
6. Ajtai M. Generating hard instances of the short basis problem. *Lecture Notes in Computer Science*, 1999, vol. 1644, pp. 1–9.
7. Buzdalov M.V. Tests generation for olympiad programming tasks using genetic algorithms. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2011, no. 2, pp. 72–77. (In Russian)
8. Buzdalov M.V. Test generation for programming challenge tasks in graph theory by evolution strategies. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2011, no. 6, pp. 123–127. (In Russian)
9. Smith-Miles K., Tan T.T. Measuring algorithm footprints in instance space. *IEEE Congress on Evolutionary Computation*. Brisbane, Australia, 2012, pp. 3446–3453. doi: 10.1109/CEC.2012.6252992
10. Asahiro Y., Iwama K., Miyano E. Random generation of test instances with controlled attributes. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 1996, vol. 26, pp. 377–393. doi: 10.1090/dimacs/026/18

V. 26. P. 377–393. doi: 10.1090/dimacs/026/18
11. Smith-Miles K., Wreford B., Lopes L., Insani N. Predicting metaheuristic performance on graph coloring problems using data mining // Studies in Computational Intelligence. 2013. V. 434. P. 417–432. doi: 10.1007/978-3-642-30671-6-16
12. Smith-Miles K., Baatar D., Wreford B., Lewis R. Towards objective measures of algorithm performance across instance space // Computers and Operations Research. 2014. V. 45. P. 12–24. doi: 10.1016/j.cor.2013.11.015
13. Giraud-Carrier C. Metalearning – a tutorial // Tutorial at 7th Int. Conf. on Machine Learning and Applications, ICMLA. San Diego, California, 2008. P. 11–13.
14. Brazdil P., Giraud-Carrier C., Soares C., Vilalta R. Metalearning. Applications to Data Mining. Springer, 2009. 176 p. doi: 10.1007/978-3-540-73263-1
15. Wolpert D.H., Macready W.G. No free lunch theorems for optimization // IEEE Transactions on Evolutionary Computation. 1997. V. 1. N 1. P. 67–82. doi: 10.1109/4235.585893
16. Wolpert D.H. The supervised learning no-free-lunch theorems // Soft Computing and Industry. 2002. P. 25–42. doi: 10.1007/978-1-4471-0123-9_3
17. Воронцов К.В. Математические методы обучения по прецедентам (теория обучения машин) [Электронный ресурс]. Режим доступа: http://docplayer.ru/2064-K-v-voroncov-http-www-ccas-ru-voron-voron-ccas-ru.html, свободный. Яз. рус. (дата обращения 24.03.2017). 140 с.
18. Николенко С.И., Тулупьев А.Л. Самообучающиеся системы. М.: МЦНМО, 2009. 287 с.
19. Jin Y., Branke J. Evolutionary optimization in uncertain environments – a survey // IEEE Transactions on Evolutionary Computation. 2005. V. 9. N 3. P. 303–317. doi: 10.1109/TEVC.2005.846356
20. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. Москва, Физматлит, 2006. 319 с.
21. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing // Science. 1983. V. 220. N 4598. P. 671–680.
22. Скобцов Ю.А. Основы эволюционных вычислений. Донецк: ДонНТУ, 2008. 326 с.
23. Vanschoren J., van Rijn J.N., Bischl B., Torgo L. OpenML: networked science in machine learning // ACM SIGKDD Explorations Newsletter. 2014. V. 15(2). P. 49–60. doi: 10.1145/2641190.2641198
24. Filchenkov A., Pendryak A. Datasets meta-feature description for recommending feature selection algorithm // Proc. Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT. St. Petersburg, Russia, 2015. P. 11–18. doi: 10.1109/AINL-ISMW-FRUCT.2015.7382962
25. Indyk P., Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality // Proc. 13th Annual ACM Symposium on Theory of Computing. Dallas, USA, 1998. P. 604–613.

11. Smith-Miles K., Wreford B., Lopes L., Insani N. Predicting metaheuristic performance on graph coloring problems using data mining. *Studies in Computational Intelligence*, 2013, vol. 434, pp. 417–432. doi: 10.1007/978-3-642-30671-6-16
12. Smith-Miles K., Baatar D., Wreford B., Lewis R. Towards objective measures of algorithm performance across instance space. *Computers and Operations Research*, 2014, vol. 45, pp. 12–24. doi: 10.1016/j.cor.2013.11.015
13. Giraud-Carrier C. Metalearning – a tutorial. *Tutorial at 7th Int. Conf. on Machine Learning and Applications, ICMLA*. San Diego, California, 2008, pp. 11–13.
14. Brazdil P., Giraud-Carrier C., Soares C., Vilalta R. *Metalearning. Applications to Data Mining*. Springer, 2009, 176 p. doi: 10.1007/978-3-540-73263-1
15. Wolpert D.H., Macready W.G. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1997, vol. 1, no. 1, pp. 67–82. doi: 10.1109/4235.585893
16. Wolpert D.H. The supervised learning no-free-lunch theorems. *Soft Computing and Industry*, 2002, pp. 25–42. doi: 10.1007/978-1-4471-0123-9_3
17. Vorontsov K.V. *Matematicheskie metody obucheniya po pretsedentam (teoriya obucheniya mashin)*. Available at: http://docplayer.ru/2064-K-v-voroncov-http-www-ccas-ru-voron-voron-ccas-ru.html (accessed 24.03.2017).
18. Nikolenko S.I., Tulup'ev A.L. *Self-Learning Systems*. Moscow, MTsNMO Publ., 2009, 287 p. (In Russian)
19. Jin Y., Branke J. Evolutionary optimization in uncertain environments – a survey. *IEEE Transactions on Evolutionary Computation*, 2005, vol. 9, no. 3, pp. 303–317. doi: 10.1109/TEVC.2005.846356
20. Gladkov L.A., Kureichik V.V., Kureichik V.M. *Genetic Algorithms*. Moscow, Fizmatlit Publ., 2006, 319 p. (In Russian)
21. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing. *Science*, 1983, vol. 220, no. 4598, pp. 671–680.
22. Skobtsov Yu.A. *Fundamentals of Evolutionary Computation*. Donetsk, DonNTU Publ., 2008, 326 p. (in Russian)
23. Vanschoren J., van Rijn J.N., Bischl B., Torgo L. OpenML: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 2014, vol. 15, pp. 49–60. doi: 10.1145/2641190.2641198
24. Filchenkov A., Pendryak A. Datasets meta-feature description for recommending feature selection algorithm. *Proc. Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT*. St. Petersburg, Russia, 2015, pp. 11–18. doi: 10.1109/AINL-ISMW-FRUCT.2015.7382962
25. Indyk P., Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality. *Proc. 13th Annual ACM Symposium on Theory of Computing*. Dallas, USA, 1998, pp. 604–613.

Авторы

Забашта Алексей Сергеевич – программист, студент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, azabashta@corp.ifmo.ru

Фильченков Андрей Александрович – кандидат физико-математических наук, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, afilchenkov@corp.ifmo.ru

Authors

Alexey S. Zabashta – programmer, student, ITMO University, Saint Petersburg, 197101, Russian Federation, azabashta@corp.ifmo.ru

Andrey A. Filchenkov – PhD, Associate professor, ITMO University, Saint Petersburg, 197101, Russian Federation, afilchenkov@corp.ifmo.ru