УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ноябрь–декабрь 2020 Том 20 № 6 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
November–December 2020 Vol. 20 No 6 ISSN 2226-1494 http://ntv.ifmo.ru/en/
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК УНИВЕРСИТЕТ ИТМО ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

УДК 004.021:004.852:004.832.23 doi: 10.17586/2226-1494-2020-20-6-828-834

ОПТИМИЗАЦИЯ ГИПЕРПАРАМЕТРОВ НА ОСНОВЕ ОБЪЕДИНЕНИЯ АПРИОРНЫХ И АПОСТЕРИОРНЫХ ЗНАНИЙ О ЗАДАЧЕ КЛАССИФИКАЦИИ

В.С. Смирнова[a,b], В.В. Шаламов[a], В.А. Ефимова[a], А.А. Фильченков[a]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[b] ООО «Ю.Би.Си», Санкт-Петербург, 192019, Российская Федерация
Адрес для переписки: valeryefimova@gmail.com

Информация о статье
Поступила в редакцию 02.10.20, принята к печати 25.10.20
Язык статьи — русский

Ссылка для цитирования: Смирнова В.С., Шаламов В.В., Ефимова В.А., Фильченков А.А. Оптимизация гиперпараметров на основе объединения априорных и апостериорных знаний о задаче классификации // Научно-технический вестник информационных технологий, механики и оптики. 2020. Т. 20. № 6. С. 828–834. doi: 10.17586/2226-1494-2020-20-6-828-834

Аннотация

Предмет исследования. Исследован метод байесовской оптимизации для настройки гиперпараметров алгоритмов, применяемых в машинном обучении для решения задач классификации. Подробно рассмотрено применение априорных и апостериорных знаний о задаче классификации для улучшения качества оптимизации гиперпараметров. **Метод.** Расширен существующий алгоритм байесовской оптимизации для настройки гиперпараметров алгоритмов классификации. Предложены поправка для функции выгоды, вычисляемая на основе гиперпараметров, найденных для подобных задач, и метрика для определения подобия задач классификации по сгенерированным мета-признакам. **Основные результаты.** Проведенные эксперименты на реальных наборах данных из открытой базы OpenML позволили подтвердить, что предложенный алгоритм в целом за фиксированное время достигает существенно лучших результатов, чем существующий алгоритм байесовской оптимизации. **Практическая значимость.** Предложенный алгоритм может быть использован для настройки гиперпараметров алгоритма классификации на любой задаче, например, в медицине, при обработке изображений или в химии.

Ключевые слова
машинное обучение, классификация, настройка гиперпараметров, байесовская оптимизация, гауссовские процессы

Благодарности
Работа выполнена при финансовой поддержке Правительства Российской Федерации, грант 08-08.

doi: 10.17586/2226-1494-2020-20-6-828-834

HYPERPARAMETER OPTIMIZATION BASED ON A PRIORI AND A POSTERIORI KNOWLEDGE ABOUT CLASSIFICATION PROBLEM

V.S. Smirnova[a,b], V.V. Shalamov[a], V.A. Efimova[a], A.A. Filchenkov[a]

[a] ITMO University, Saint Petersburg, 197101, Russian Federation
[b] OOO U.B.C., Saint Petersburg, 192019, Russian Federation
Corresponding author: valeryefimova@gmail.com

Article info
Received 02.10.20, accepted 25.10.20
Article in Russian

For citation: Smirnova V.S., Shalamov V.V., Efimova V.A., Filchenkov A.A. Hyperparameter optimization based on a priori and a posteriori knowledge about classification problem. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2020, vol. 20, no. 6, pp. 828–834 (in Russian). doi: 10.17586/2226-1494-2020-20-6-828-834

Abstract

Subject of Research. The paper deals with Bayesian method for hyperparameter optimization of algorithms, used in machine learning for classification problems. A comprehensive survey is carried out about using a priori and a posteriori knowledge in classification task for hyperparameter optimization quality improvement. **Method.** The existing Bayesian optimization algorithm for hyperparameter setting in classification problems was expanded. We proposed a target function modification calculated on the basis of hyperparameters optimized for the similar problems and a metric

for determination of similarity classification problems based on generated meta-features. **Main Results.** Experiments carried out on the real-world datasets from OpenML database have confirmed that the proposed algorithm achieves usually significantly better performance results than the existing Bayesian optimization algorithm within a fixed time limit. **Practical Relevance.** The proposed algorithm can be used for hyperparameter optimization in any classification problem, for example, in medicine, image processing or chemistry.

Keywords
machine learning, classification, hyperparameter optimization, Bayesian optimization, Gaussian processes

Acknowledgements
This work was financially supported by the Government of the Russian Federation, Grant 08-08.

Введение

Классификация заключается в максимально точном определении класса объекта и относится к задачам обучения с учителем, т. е. алгоритм имеет доступ к ограниченному множеству уже размеченных объектов. Задача классификации возникает во многих областях, среди которых медицинская диагностика, геологоразведка, оптическое распознавание текстов и синтез химических соединений [1].

Существует множество алгоритмов классификации, среди них метод ближайших соседей, дерево принятия решений, наивный байесовский классификатор [2], случайный лес [3], многослойный перцептрон Румельхарта [4, 5] и многие другие. Найти и обучить эффективный для данной задачи алгоритм классификации — трудоемкая задача, которая включает в себя выбор самого классификатора, сбор и разметку данных, и непосредственно настройку гиперпараметров классификатора. Гиперпараметрами алгоритма могут быть количество деревьев в случайном лесу, число слоев в перцептроне. Как и другие алгоритмы машинного обучения, алгоритмы классификации при разных гиперпараметрах выдают разный результат на одном наборе данных [6], т. е. подобрав оптимальные гиперпараметры можно значительно улучшить качество классификации [7].

На практике настройка гиперпараметров алгоритма часто выполняется экспертами вручную методом проб и ошибок. Это требует значительных временных и вычислительных затрат. Раньше экспертам были доступны только компьютеры малых мощностей, что оправдывало этот подход. Сейчас вычисления стали значительно дешевле и быстрее, из-за чего выгоднее потратить больше процессорного времени, чтобы сэкономить время человека-эксперта, автоматизировав процесс оптимизации гиперпараметров.

Цель данного исследования — предложить алгоритм настройки гиперпараметров для алгоритмов классификации, который бы учитывал априорные и апостериорные (т. е. получаемые в процессе оптимизации гиперпараметров) знания о задачах.

Оптимизация гиперпараметров алгоритмов машинного обучения

Сначала формально определим задачу. Моделью A будем называть алгоритм классификации. Каждая модель задается некоторым набором гиперпараметров, лежащем в пространстве гиперпараметров именно этой модели, $\lambda = \{\lambda_1, \lambda_2,\ldots,\lambda_n\} \in \Lambda$, будем называть такой набор *конфигурацией*. Модель с конкретным набором гиперпараметров λ обозначим как A_λ. Пусть задана некоторая мера качества классификации Q. Задача оптимизации гиперпараметров заключается в подборе таких $\lambda^* \in \Lambda$, при которых заданная модель алгоритма классификации A достигнет наилучшего качества на наборе объектов X: $Q(A_\lambda, X) \to \max_{\lambda\in\Lambda}$. Отметим, что даже один алгоритм с разными гиперпараметрами по-разному классифицирует множество объектов, что, следовательно, дает разные оценки качества классификации. Соответственно, возникает задача оптимизации гиперпараметров алгоритма классификации [7]. Она достаточно нова и не имеет под собой десятки лет исследований, но уже существует много подходов к ее решению, рассмотрим наиболее популярные из них и оценим их преимущества и недостатки [8].

Поиск по решетке. Алгоритм заключается в полном переборе всех возможных конфигураций классификатора. Он реализован в библиотеках LIBSVM [9], Scikit-learn [10]. В случае полного перебора гарантированно будет найдена лучшая конфигурация, но на это потребуются слишком большие затраты ресурсов.

Случайный поиск. В этом алгоритме проверяются не все возможные конфигурации, а некоторая случайная их выборка. Реализован в библиотеках Hyperopt [11], Scikit-learn [10], H_2O AutoML[1]. Затраты на обучение в среднем существенно меньше, чем при полном переборе, но нельзя точно определить, сколько времени потребуется на поиск наилучшей конфигурации.

Оптимизация на основе градиентов. Применяется только для некоторых алгоритмов, поиск лучших гиперпараметров происходит с помощью градиентного спуска. Метод реализован в библиотеке hypergrad [12]. При небольших затратах на обучение достигается сравнительно высокое качество.

Эволюционная оптимизация. Алгоритм классификации представляется «черным ящиком» с шумом, для поиска оптимальной конфигурации используются эволюционные подходы [13]. Реализована в библиотеках TPOT [14], DEAP [15]. Данный подход используется только для статистических алгоритмов, но дает хороший результат при сравнительно малых затратах.

Байесовская оптимизация (БО). Метод, основанный на обращении к функции «черного ящика» с шумом. В рассматриваемом случае «черный ящик» — алгоритм классификации. Для задачи оптимизации гиперпараметров строит стохастическую модель отображения из конфигурации в целевую функцию. Байесовская оптимизация — один из самых распро-

[1] https://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html (дата обращения: 08.11.2020).

страненных и эффективных подходов к задаче оптимизации гиперпараметров [8]. Реализована в библиотеках Spearmint [16], Bayesopt [17], AutoWEKA [18], Autosklearn [19], mlrMBO [20], tuneRanger [21], SMAC [22]. Метод позволяет адаптировать поиск на основе значимости каждого гиперпараметра для конкретной задачи, хоть и сложен в реализации и при использовании. Рассмотрим байесовскую оптимизацию более подробно.

Байесовская оптимизация

Алгоритм байесовской оптимизации заключается в следующей цепочке действий, повторяемых на каждой итерации (число итераций обычно задается вручную):

1) выбрать следующую точку в пространстве гиперпараметров;
2) получить результат целевой функции в этой точке;
3) сохранить полученное значение и конфигурацию.

Наиболее интересная и значимая часть в этом алгоритме — выбор следующей для рассмотрения точки. Именно в этой части *модель* обучается на уже рассмотренных точках и обновляет так называемую *функцию выгоды* (acquisition function), после чего следующая точка определяется как максимум функции выгоды, так как там находится место с наибольшей неопределенностью. Другими словами, получение знаний о $Q(A_\lambda, X)$ в этой точке принесет больше всего информации. Подробнее подход описан в [23].

Модель определяет, как будет обновляться пространство гиперпараметров модели оптимизатора после каждого шага. Важно отметить, что гиперпараметры оптимизатора никаким образом не связаны с гиперпараметрами классификатора. Наиболее распространенные модели: гауссовские процессы, гауссовский процесс с марковской цепью Монте-Карло, случайный лес, гамильтоновы нейронные сети Монте-Карло [24], глубокие нейронные сети (DNGO) [25]. У каждой модели есть свои преимущества и недостатки, например, в гауссовском процессе нет возможности работать с категориальными признаками, DNGO дает наименее точный результат.

Функция выгоды служит для определения наименее информативной области. В этой области меньше всего информации о значении функции, которую необходимо аппроксимировать, поэтому в экстремуме функции выгоды будет находиться наиболее интересная на данный момент конфигурация гиперпараметров. Наиболее часто в качестве функции выгоды используется *ожидаемое улучшение* (expected improvement, EI) [22].

Расширение байесовской оптимизации для настройки гиперпараметров

В настоящей работе рассматривается задача оптимизации гиперпараметров алгоритма классификации и расширение существующего алгоритма до получения лучших результатов при таком же временном лимите на обучение.

Итеративные методы (поиск по сетке, случайный поиск и пр.) предполагают, что вычисление функции Q бесплатно, тогда как БО учитывает затратность вычисления этой функции. За основу предложенного решения выбрана байесовская оптимизация на основе гауссовского процесса.

В работе используются апостериорные знания, т. е. полученные при решении других задач. Для этого введем оценку похожести задач, а именно функцию *расстояния между задачами*, для ее корректного подсчета необходимо, чтобы аргументы имели одинаковую размерность, что достигается с помощью извлечения мета-признаков [19] из задачи. Для каждого признака возьмем некую характеристику его связи с классом, попарные корреляции между признаками и характеристики структуры дерева принятия решений[1]. После чего на каждом полученном векторе посчитаем статистики (минимум, максимум, среднее), тогда полученный вектор и будет вектором мета-признаков. Необходимо также нормализовать векторы полученных мета-признаков. После нормализации получим не только подходящие значения, но и устраним ковариации, и уровняем дисперсию. В качестве нормализации используем расстояние Махаланобиса — это мера расстояния между векторами случайных величин, обобщающая понятие евклидова расстояния. Именно данный подход к нормализации учитывает в себе ковариации. Формально расстояние Махаланобиса от рассматриваемого вектора $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ до множества со средним значением $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_N)^T$ и матрицей ковариации $\mathbf{S}$ определяется следующим образом: $D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}$.

Для удобства изначально домножим на $\mathbf{S}^{-1/2}$, чтобы сократить затраты на расчет. После того, как получим набор нормализованных признаков, определим расстояние между задачами как обыкновенное евклидово расстояние между векторами $\mathbf{p}$ и $\mathbf{q}$:

$$D(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}. \quad (1)$$

Для старта оптимизации гиперпараметров с помощью гауссовского процесса необходимы три начальные точки, в общем случае эти точки задаются случайно. В настоящей работе учитываются результаты обучения на подобных задачах, а также мера подобия задач, и можно выбрать начальные точки на основе апостериорных знаний.

Определим понятие *достоверности* (reliability), которое будет показывать, на сколько можно «доверять» апостериорным знаниям о той или иной задаче. Введенную ранее функцию расстояния использовать некорректно, так как не происходит нормализации для модели оптимизации, и значения могут значительно меняться с добавлением новых задач. Потому введем следующее определение достоверности для задачи k на итерации t:

$$R_k^t = (\alpha^t - D(d_k, d_i) + (1 - \alpha^t)\varsigma_{k,i}^{global}),$$

где α — мера «забывания» апостериорной информации, лежащая в интервале [0; 1]; d_i — расстояние до задачи i;

[1] Полный список: https://github.com/tina80lvl/prior_posterior_hpo/blob/master/metrics/raw_meta_features.csv (дата обращения: 09.11.2020).

$D(d_k, d_i)$ — евклидово расстояние между задачами k и i, вычисленное по формуле (1) для нормализованных векторов мета-признаков; $\varsigma_{k,i}^{global}$ — число, найденное по следующей формуле, которое определяет глобальное подобие гиперпараметров λ для задач k и i:

$$\varsigma_{k,i}^{local} = \{\Delta\lambda_1, \Delta\lambda_2, \ldots, \Delta\lambda_N\}, \varsigma_{k,i}^{global} = \text{average}_{j=1,N}\varsigma_{k,i}^{local}.$$

Иначе говоря, ς^{global} определяет расстояние на графике зависимости $Q(\lambda)$ между значениями целевой функции Q для каждой пары задач.

Таким образом, на каждой итерации оптимизатора будет получено новое значение достоверности для решаемой задачи и подобных, которая считается в качестве поправки для значения целевой функции Q. Гауссовский процесс тренируется как раз в момент выбора следующего кандидата конфигурации, в качестве кандидата возвращается набор координат максимума функции выгоды. Непосредственно поправка, посчитанная относительно подобной задачи, добавляется в момент обучения модели, а именно, возводится в квадрат и добавляется по диагонали ковариационной матрицы для гауссовского процесса. В связи с этим значение «неопределенности» в конкретных точках будет изменяться пропорционально значению достоверности подобной задачи.

Описание экспериментов

Для запуска алгоритма БО нужно зафиксировать алгоритм классификации, стоит учесть, что в качестве модели оптимизатора выступает гауссовский процесс, который не предусматривает работу с категориальными гиперпараметрами. Кроме того, оптимизатор определяет значимость гиперпараметров в ходе настройки, поэтому для лучшей валидации результатов стоит выбрать классификатор с обширным набором гиперпараметров. Требования к классификатору были проанализированы, и выбран многослойный перцептрон Румельхарта [4, 5]. В классификаторе выделены гиперпараметры, которые варьировались в ходе решения, представленные в таблице.

Итого, в качестве модели для БО взят гауссовский процесс, в качестве оптимизатора — случайное сэмплирование, в качестве функции выгоды — логарифмическая функция ожидаемого улучшения, за целевую функцию принята обратная f_1-мера для многослойного перцептрона Румельхарта ($1 - f_1$), так как в БО происходит минимизация целевой функции. Для проведения достоверных экспериментов из открытого ресурса OpenML было отобрано 45 максимально различных (по количеству объектов, признаков и классов) наборов данных, подходящих для задачи классификации. За основу для расширенной байесовской оптимизации взята реализация из пакета RoBo [26].

Так как и алгоритм оптимизации, и сама модель частично основываются на случайных значениях, то будет некорректно сравнивать ответы по одному запуску. Исходя из этого, на каждом наборе данных запускалось по 100 итераций байесовской оптимизации 10 раз, и сравнивались средние значения. Для оценки статистических результатов использовался критерий Вилкоксона: на основе нулевой гипотезы критерий считает попарную значимость результатов, т. е. действительно ли один подход лучше другого, либо различие результатов в данном случае незначительно. Оценивались две выборки: статистические результаты по 10 запускам классической БО и статистические результаты по 10 запускам расширенной БО. Между этими парными выборками и считается критерий Вилкоксона или, как его называют для независимых выборок, критерий Манна–Уитни [27].

Для оценки результата использовалось значение целевой функции, достигаемой на *действующем образце* (incumbent) — наилучшем на данный момент значении. В настоящей работе выполнено сравнение результатов классической байесовской оптимизации с результатами предложенного модифицированного алгоритма.

Для того чтобы определить критерии сравнения результатов, выделены все возможные случаи по завершении последней итерации:

1) получено лучшее оптимальное значение целевой функции на более ранней итерации — абсолютная победа предложенного решения;
2) получено лучшее оптимальное значение целевой функции на той же итерации — победа предложенного решения по абсолютному значению, ничья по номеру итерации;
3) получено лучшее оптимальное значение целевой функции, но на более поздней итерации — победа предложенного решения по абсолютному значению, проигрыш по номеру итерации;

Таблица. Гиперпараметры многослойного перцептрона Румельхарта

Гиперпараметр	Диапазон значений
Число скрытых слоев	1–50
Штраф L_2	0,00001–0,01
Начальная скорость обучения	0,0001–0,1
Максимальное число итераций	50–300
Доля тренировочных данных, отведенных в качестве проверки для преждевременной остановки	0,01–0,9
Скорость экспоненциального убывания для оценок вектора первого момента	0,09–0,9
Скорость экспоненциального убывания для оценок вектора второго момента	0,0999–0,999
Максимальное число эпох, пройденных без прогресса	5–15

4) оптимальные значения одинаковы (в пределах значащих цифр), но в предложенном решении это значение получено на более ранней итерации — ничья по абсолютному значению, победа предложенного решения по номеру итерации;
5) оптимальное значение хуже, но получено на более ранней итерации — проигрыш по абсолютному значению, победа по номеру итерации.

Для случаев 1, 2, 4 превосходство предложенного решения очевидно, однако, в случае 3 несправедливо утверждать, что предлагаемое решение проиграло по номеру итерации, так как возможны (и весьма вероятны) случаи, когда на момент нахождения оптимального значения классической БО, абсолютное значение в предлагаемом решении уже было лучше, т. е. оптимальное значение классической оптимизации было получено раньше. В случае 5 вовсе нельзя говорить о победе предложенного решения, так как в первую очередь важно абсолютное значение.

Полный список результатов для классической и расширенной БО представлен в приложении[1]. В репозитории[2] находится реализация описываемого метода, а также код, использованный для проведения экспериментов.

Теперь сравним результаты работы классической и расширенной байесовской оптимизации для наборов данных *balance-scale*, *cardiotocography*, *robot-failures-ip1*, *shuttle*. На графиках, представленных на рисунке розовой линией, показаны результаты работы предложенного расширения БО на основе апостериорной информации. Можно отметить, что оптимальный результат получен значительно раньше и по абсолютному значению также побеждает классическую байесовскую оптимизацию. По вертикали располагаются лучшие на данный момент значения целевой функции от действующего образца. Заметим также что действующий образец описывает лучшее на данный момент времени (на данной итерации) значение целевой функции, т. е. на графике на участках-плато нельзя утверждать, что значение целевой функции совпадает со значением на предыдущей итерации, наоборот, вероятнее всего значение получилось хуже и просто не включилось в век-

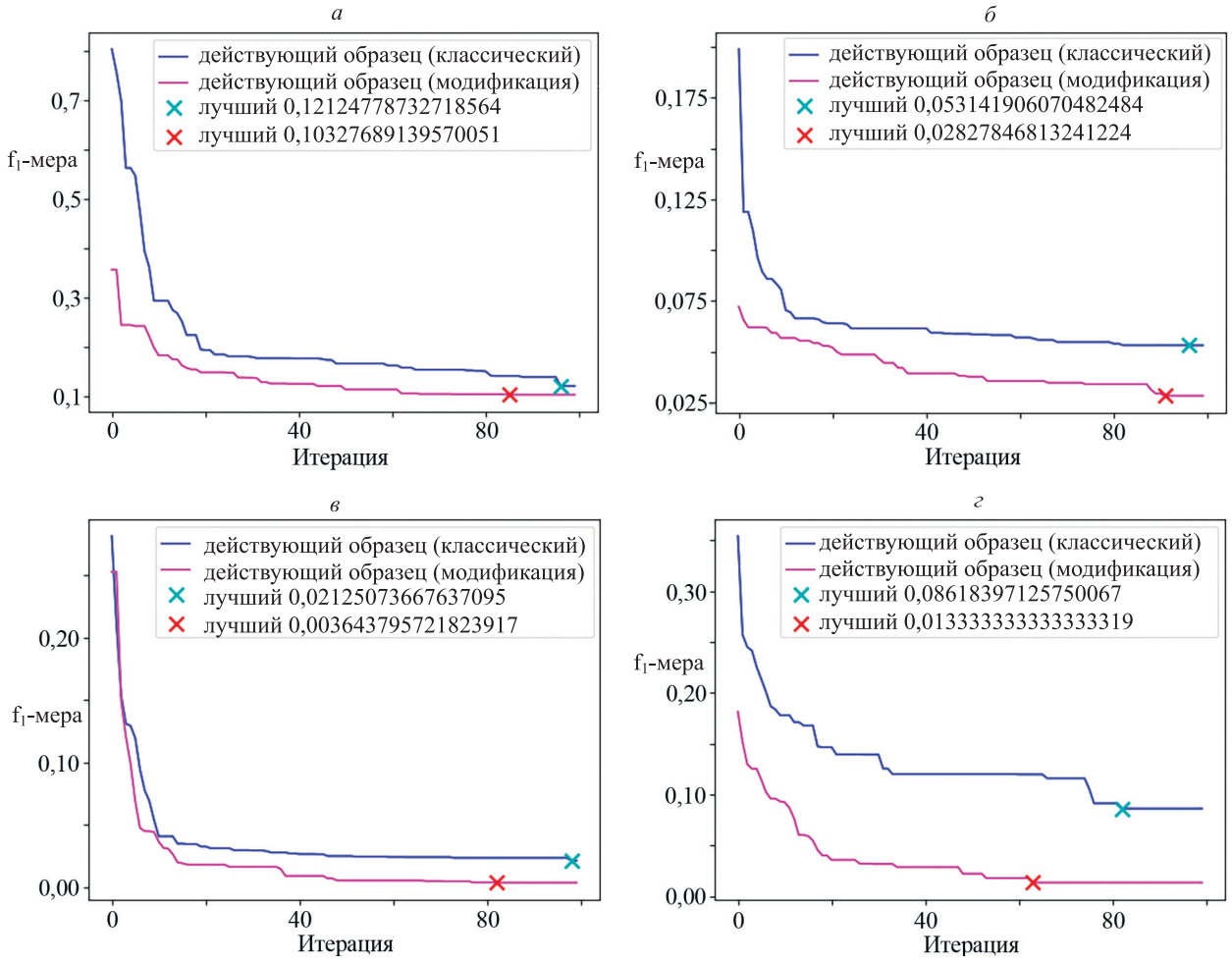

Рисунок. Результат работы расширенной байесовской оптимизации на наборах данных: cardiotocography (*а*); balance-scale (*б*); shuttle (*в*); robot-failures-ip1 (*г*)

[1] https://www.dropbox.com/s/jdkve1zz3p6juh5/results.pdf (дата обращения: 09.11.2020).

[2] https://github.com/tina80lvl/prior_posterior_hpo (дата обращения: 09.11.2020).

тор ответов (но учлось при дальнейшей оптимизации, так как каждая итерация добавляет определенности апостериорной информации).

Так как все наборы данных разные и содержат величины совершенно разной размерности и порядка, общий результат без нормализации оценить количественно не получится. Определим, улучшило ли предложенное расширение результат классической БО, дав количественную оценку этому результату.

Абсолютными победами назовем те задачи, в которых предложенный алгоритм показал статистически значимые улучшения. *Условными победами* назовем те случаи, когда значение Q при обоих вариантах оптимизации гиперпараметров получилось одинаковым, либо оказалось статистически незначимым, однако значимым оказался номер итерации, на котором это значение было достигнуто. *Поражениями* — остальные случаи.

В результате получили для 45 задач: 27 абсолютных побед, 9 условных побед, 9 поражений. Итого, с учетом всех критериев оценки результата получено 36 побед и 9 поражений, что говорит о том, что в 80 % задач предложенный алгоритм дает значительно лучший результат либо значительно ускоряет оптимизацию. С учетом того, что на данный момент байесовская оптимизация считается самым эффективным алгоритмом для задачи оптимизации гиперпараметров, то полученные результаты можно считать значительными и наилучшими на сегодняшний день.

Заключение

В работе предложено расширение существующего алгоритма настройки гиперпараметров на основе объединения априорных и апостериорных знаний о задачах классификации. Предложена метрика для определения подобия задач классификации по построенным мета-признакам.

Оценка результатов произведена на реальных данных. С учетом специфики модели и оптимизатора, проведено необходимое для корректной статистической оценки результатов количество опытов, посчитаны все необходимые меры и критерии, что позволило качественно оценить результаты. Предложенное изменение существенно улучшило существующий алгоритм байесовской оптимизации. В итоге получен лучший на сегодняшний день результат для задачи оптимизации гиперпараметров, что говорит о том, что данная работа является актуальной в исследуемой области и может быть использована для дальнейших исследований.

В дальнейшем планируется запустить предложенный метод на других классификаторах. В алгоритме байесовской оптимизации планируется использовать другие оптимизаторы, в том числе случайный лес.

Литература

1. Zhang G.P. Neural networks for classification: a survey // IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2000. V. 30. N 4. P. 451–462. doi: 10.1109/5326.897072
2. Aly M. Survey on multiclass classification methods: Technical Report, Caltech. California Institute of Technology, 2005. 9 p.
3. Liaw A., Wiener M. Classification and regression by randomForest // R news. 2002. V. 2. N 3. P. 18–22.
4. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Science & Business Media, 2009. 745 p.
5. Van Der Malsburg C. Frank Rosenblatt: principles of neurodynamics: perceptrons and the theory of brain mechanisms // Brain Theory. Springer, 1986. P. 245–248. doi: 10.1007/978-3-642-70911-1_20
6. Муравьёв С.Б., Ефимова В.А., Шаламов В.В., Фильченков А.А., Сметанников И.Б. Автоматическая настройка гиперпараметров алгоритмов кластеризации с помощью обучения с подкреплением // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 3. С. 508–515. doi: 10.17586/2226-1494-2019-19-3-508-515
7. Ефимова В.А., Фильченков А.А., Шалыто А.А. Применение обучения с подкреплением для одновременного выбора модели алгоритма классификации и ее структурных параметров // Машинное обучение и анализ данных. 2016. Т. 2. № 2. С. 244–254.
8. Yu T., Zhu H. Hyper-parameter optimization: A review of algorithms and applications // arXiv preprint. arXiv:2003.05689. 2020.
9. Chang C.-C., Lin C.-J. LIBSVM: A library for support vector machines // ACM Transactions on Intelligent Systems and Technology. 2011. V. 2. N 3. P. 27. doi: 10.1145/1961189.1961199
10. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay É. Scikit-learn: Machine learning in Python // Journal of Machine Learning Research. 2011. V. 12. P. 2825–2830.
11. Bergstra J., Komer B., Eliasmith C., Yamins D., Cox D.D. Hyperopt: a Python library for model selection and hyperparameter optimization // Computational Science & Discovery. 2015. V. 8. N 1. P. 014008. doi: 10.1088/1749-4699/8/1/014008

References

1. Zhang G.P. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 2000, vol. 30, no. 4, pp. 451–462. doi: 10.1109/5326.897072
2. Aly M. *Survey on multiclass classification methods*. Technical Report, Caltech, California Institute of Technology, 2005, 9 p.
3. Liaw A., Wiener M. Classification and regression by randomForest. *R news*, 2002, vol. 2, no. 3, pp. 18–22.
4. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009, 745 p.
5. Van Der Malsburg C. Frank Rosenblatt: principles of neurodynamics: perceptrons and the theory of brain mechanisms. *Brain Theory*, Springer, 1986, pp. 245–248. doi: 10.1007/978-3-642-70911-1_20
6. Muravyov S.B., Efimova V.A., Shalamov V.V., Filchenkov A.A., Smetannikov I.B. Automatic hyperparameter optimization for clustering algorithms with reinforcement learning. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2019, vol. 19, no. 3, pp. 508–515. (in Russian). doi: 10.17586/2226-1494-2019-19-3-508-515
7. Efimova V.A., Filchenkov A.A., Shalyto A.A. Reinforcement-based simultaneous classification model and its hyperparameters selection. *Machine Learning and Data Analysis*, 2016, vol. 2, no. 2, pp. 244–254. (in Russian)
8. Yu T., Zhu H. Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint*, arXiv:2003.05689, 2020.
9. Chang C.-C., Lin C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011, vol. 2, no. 3, pp. 27. doi: 10.1145/1961189.1961199
10. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay É. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011, vol. 12, pp. 2825–2830.
11. Bergstra J., Komer B., Eliasmith C., Yamins D., Cox D.D. Hyperopt: a Python library for model selection and hyperparameter optimization.

12. Maclaurin D., Duvenaud D., Adams R. Gradient-based hyperparameter optimization through reversible learning // Proceedings 32nd International Conference on Machine Learning. 2015. P. 2113–2122.
13. Bergstra J.S., Bardenet R., Bengio Y., Kégl B. Algorithms for hyper-parameter optimization // Advances in Neural Information Processing Systems: Proc. 5th Annual Conference on Neural Information Processing Systems (NIPS 2011). 2011. P. 2546–2554.
14. Gijsbers P., Vanschoren J., Olson R.S. Layered TPOT: Speeding up tree-based pipeline optimization // arXiv preprint. arXiv:1801.06007. 2018.
15. Fortin F.-A., De Rainville F.-M., Gardner M.-A., Parizeau M., Gagńe C. DEAP: Evolutionary algorithms made easy // Journal of Machine Learning Research. 2012. V. 13. P. 2171–2175.
16. Hazan E., Klivans A., Yuan Y. Hyperparameter optimization: A spectral approach // arXiv preprint. arXiv:1706.00764. 2017.
17. Martinez-Cantin R. BayesOpt: A bayesian optimization library for nonlinear optimization, experimental design and bandits // Journal of Machine Learning Research. 2014. V. 15. P. 3735–3739.
18. Thornton C., Hutter F., Hoos H.H., Leyton-Brown K. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms // Proc. 19th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2013). 2013. P. 847–855. doi: 10.1145/2487575.2487629
19. Feurer M., Klein A., Eggensperger K., Springenberg J.T., Blum M., Hutter F. Efficient and robust automated machine learning // Advances in Neural Information Processing Systems. 2015. P. 2962–2970.
20. Bischl B., Richter J., Bossek J., Horn D., Thomas J., Lang M. mlrMBO: A modular framework for model-based optimization of expensive black-box functions // arXiv preprint. arXiv:1703.03373. 2017.
21. Probst P., Wright M.N., Boulesteix A.-L. Hyperparameters and tuning strategies for random forest // Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2019. V. 9. N 3. P. e1301. doi: 10.1002/widm.1301
22. Hutter F., Hoos H.H., Leyton-Brown K. Sequential model-based optimization for general algorithm configuration // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2011. V. 6683. P. 507–523. doi: 10.1007/978-3-642-25566-3_40
23. Feurer M., Hutter F. Hyperparameter optimization // Automated Machine Learning. Springer, 2019. P. 3–33. doi: 10.1007/978-3-030-05318-5_1
24. Springenberg J.T., Klein A., Falkner S., Hutter F. Bayesian optimization with robust Bayesian neural networks // Advances in Neural Information Processing Systems. 2016. P. 4141–4149.
25. Snoek J., Ripped O., Swersky K., Kiros R., Satish N., Sundaram N., Patwary M.M.A., Prabhat, Adams R.P. Scalable Bayesian optimization using deep neural networks // Proc. 32nd International Conference on Machine Learning (ICML). 2015. P. 2171–2180.
26. Klein A., Falkner S., Mansur N., Hutter F. RoBO: A flexible and robust bayesian optimization framework in Python // Proc. 31st Conference on Neural Information Processing Systems. 2017.
27. Mann H.B., Whitney D.R. On a test of whether one of two random variables is stochastically larger than the other // Annals of Mathematical Statistics. 1947. V. 18. N 1. P. 50–60.

Computational Science & Discovery, 2015, vol. 8, no. 1, pp. 014008. doi: 10.1088/1749-4699/8/1/014008
12. Maclaurin D., Duvenaud D., Adams R. Gradient-based hyperparameter optimization through reversible learning. *Proceedings 32nd International Conference on Machine Learning*, 2015, pp. 2113–2122.
13. Bergstra J.S., Bardenet R., Bengio Y., Kégl B. Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems: Proc. 5th Annual Conference on Neural Information Processing Systems (NIPS 2011)*, 2011, pp. 2546–2554.
14. Gijsbers P., Vanschoren J., Olson R.S. Layered TPOT: Speeding up tree-based pipeline optimization. *arXiv preprint*, arXiv:1801.06007, 2018.
15. Fortin F.-A., De Rainville F.-M., Gardner M.-A., Parizeau M., Gagńe C. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 2012, vol. 13, pp. 2171–2175.
16. Hazan E., Klivans A., Yuan Y. Hyperparameter optimization: A spectral approach. *arXiv preprint,* arXiv:1706.00764, 2017.
17. Martinez-Cantin R. BayesOpt: A bayesian optimization library for nonlinear optimization, experimental design and bandits. *Journal of Machine Learning Research*, 2014, vol. 15, pp. 3735–3739.
18. Thornton C., Hutter F., Hoos H.H., Leyton-Brown K. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. *Proc. 19th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2013)*, 2013, pp. 847–855. doi: 10.1145/2487575.2487629
19. Feurer M., Klein A., Eggensperger K., Springenberg J.T., Blum M., Hutter F. Efficient and robust automated machine learning. *Advances in Neural Information Processing Systems*, 2015, pp. 2962–2970.
20. Bischl B., Richter J., Bossek J., Horn D., Thomas J., Lang M. mlrMBO: A modular framework for model-based optimization of expensive black-box functions. *arXiv preprint*, arXiv:1703.03373, 2017.
21. Probst P., Wright M.N., Boulesteix A.-L. Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2019, vol. 9, no. 3, pp. e1301. doi: 10.1002/widm.1301
22. Hutter F., Hoos H.H., Leyton-Brown K. Sequential model-based optimization for general algorithm configuration. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6683, pp. 507–523. doi: 10.1007/978-3-642-25566-3_40
23. Feurer M., Hutter F. Hyperparameter optimization. *Automated Machine Learning*. Springer, 2019, pp. 3–33. doi: 10.1007/978-3-030-05318-5_1
24. Springenberg J.T., Klein A., Falkner S., Hutter F. Bayesian optimization with robust Bayesian neural networks. *Advances in Neural Information Processing Systems*, 2016, pp. 4141–4149.
25. Snoek J., Ripped O., Swersky K., Kiros R., Satish N., Sundaram N., Patwary M.M.A., Prabhat, Adams R.P. Scalable Bayesian optimization using deep neural networks. *Proc. 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 2171–2180.
26. Klein A., Falkner S., Mansur N., Hutter F. RoBO: A flexible and robust bayesian optimization framework in Python. *Proc. 31st Conference on Neural Information Processing Systems*, 2017.
27. Mann H.B., Whitney D.R. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 1947, vol. 18, no. 1, pp. 50–60.

Авторы

Смирнова Валентина Сергеевна — старший лаборант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация; разработчик, ООО «Ю.Би.Си», Санкт-Петербург, 192019, Российская Федерация, ORCID: 0000-0002-4774-3624, smirnova.vs.25@gmail.com
Шаламов Вячеслав Владимирович — программист, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57191077141, ORCID: 0000-0002-5647-6521, sslavian812@gmail.com
Ефимова Валерия Александровна — научный сотрудник, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57207459404, ORCID: 0000-0002-5309-2207, valeryefimova@gmail.com
Фильченков Андрей Александрович — кандидат физико-математических наук, руководитель лаборатории, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 55507568200, ORCID: 0000-0002-1133-8432, aaafil@mail.ru

Authors

Valentina S. Smirnova — Senior Laboratory Assistant, ITMO University, Saint Petersburg, 197101, Russian Federation; Software Engineer, OOO U.B.C., Saint Petersburg, 192019, Russian Federation, ORCID: 0000-0002-4774-3624, smirnova.vs.25@gmail.com
Viacheslav V. Shalamov — Software Engineer, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57191077141, ORCID: 0000-0002-5647-6521, sslavian812@gmail.com
Valeria A. Efimova — Researcher, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57207459404, ORCID: 0000-0002-5309-2207, valeryefimova@gmail.com

Andrey A. Filchenkov — PhD, Laboratory Head, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 55507568200, ORCID: 0000-0002-1133-8432, aaafil@mail.ru