УДК 004.85+ 004.032.26+004.382.2
DOI: 10.15827/0236-235X.130.218-228

Дата подачи статьи: 27.02.20
2020. Т. 33. № 2. С. 218–228

Применение машинного обучения для прогнозирования времени выполнения суперкомпьютерных заданий

А.В. Баранов [1], *к.т.н., доцент, ведущий научный сотрудник, antbar@mail.ru, abaranov@jscc.ru*
Д.С. Николаев [2], *старший научный сотрудник, dmitry.s.nikolaev@gmail.com*

[1] *Межведомственный суперкомпьютерный центр РАН – филиал ФНЦ НИИСИ РАН, г. Москва, 119334, Россия*
[2] *Научно-исследовательский институт «Квант», г. Москва, 125438, Россия*

Статья посвящена методам и алгоритмам машинного обучения для прогнозирования времени выполнения суперкомпьютерных заданий. Статистика работы суперкомпьютерных систем коллективного пользования показывает, что фактическое время выполнения большинства заданий существенно расходится со временем, запрошенным пользователем. Это снижает эффективность планирования заданий, поскольку неточная оценка времени выполнения приводит к неоптимальному расписанию запусков заданий.

В статье рассмотрена классификация заданий, в основу которой положено отношение фактического времени выполнения задания к запрошенному. Всего было определено шесть классов заданий, причем отношение фактического времени выполнения к запрошенному для каждого класса на порядок отличается от предыдущего класса. Прогнозирование времени выполнения задания осуществляется на основе данных статистики суперкомпьютерной системы коллективного пользования путем отнесения поступающего в систему задания к одному из классов. В качестве исходных данных была использована представленная в формате SWF статистика суперкомпьютера RIKEN Integrated Cluster of Clusters (RICC). Результаты анализа этих статистических данных позволили выявить значимые для машинного обучения признаки потока заданий, было проведено ранжирование признаков по важности и обнаружены скрытые закономерности, влияющие на точность прогнозирования, в частности, определена взаимная корреляция отобранных признаков.

Для распространенных алгоритмов машинного обучения, таких как логистическая регрессия, дерево решений, k ближайших соседей, линейный дискриминантный анализ, метод опорных векторов, случайный лес, градиентный бустинг, нейронная сеть прямого распространения, были получены оценки вероятности верного прогноза. Наилучшие значения показали алгоритмы дерева решений, случайного леса и нейронные сети прямого распространения.

Ключевые слова: *высокопроизводительные вычисления, системы управления заданиями, планирование суперкомпьютерных заданий, машинное обучение, прогнозирование времени выполнения задания.*

Современные суперкомпьютеры, как правило, функционируют в режиме коллективного пользования. Для производства расчетов пользователь должен сформировать так называемое вычислительное задание, состоящее из расчетной программы, исходных данных и требований к объему (число процессорных ядер или узлов суперкомпьютера, размер оперативной или дисковой памяти и др.) вычислительных ресурсов и времени выполнения задания. Формализованное описание задания называется его паспортом. Система коллективного пользования суперкомпьютером, принимая в виде паспортов входной поток заданий от разных пользователей, формирует из поступивших заданий очередь. Процесс формирования и ведения очереди часто называют планированием заданий, которое осуществляется специальными программными системами [1], такими как SLURM, PBS, LSF, часто называемыми *системами управления заданиями* (СУЗ) [2–4].

Планировщик СУЗ на основе информации из паспортов заданий формирует расписание их запусков. В соответствии с этим расписанием планировщик СУЗ выдает прогноз времени запуска каждого находящегося в очереди задания, основанный на заказанном (запрошенном) пользователем времени выполнения задания. При превышении заказанного времени задание в большинстве СУЗ снимается с выполнения с вероятной потерей результатов расчетов. Стараясь избежать таких потерь, пользователи часто при формировании задания существенно завышают необходимое для вы-

полнения время. Например, статистика Межведомственного суперкомпьютерного центра РАН показывает, что пользователи в среднем завышают время выполнения на 86 %. Завышенная оценка требуемого времени выполнения приводит к необходимости переформирования расписания при каждом досрочном завершении задания, при этом возможны как простои досрочно освободившихся ресурсов, так и увеличение времени ожидания в очереди отдельных заданий. Отрицательное влияние такой переоценки показано в исследовании [5], в котором неточное указание пользователями времени выполнения заданий привело к невозможности оптимизации планирования за счет заполнения системы заданиями небольшого размера. При недооценке требуемого времени выполнения задание может быть принудительно завершено СУЗ, что приведет к повторной постановке пользователем этого задания в очередь с последней сохраненной контрольной точки. В любом случае (как при переоценке, так и при недооценке требуемого времени выполнения) происходит снижение эффективности планирования заданий.

Более точное планирование возможно за счет предварительной оценки времени выполнения задания на этапе его постановки в очередь. К настоящему времени существует множество исследований по аналитической обработке накопленной статистики работы суперкомпьютеров с целью выявления различных аномалий, предупреждения сбоев, прогнозирования времени работы и т.д.

В работе [6] представлен метод прогнозирования отказов и сбоев суперкомпьютерной системы на основе структурированных данных мониторинга. Предполагается, что каждому отказу или сбою предшествует критический период времени, в который определенным образом изменяются наблюдаемые системой мониторинга характеристики системы. Модель прогнозирования создается с использованием машинного обучения с учителем, при этом обучение осуществляется на основе данных мониторинга критических и обычных временных периодов. В статье [7] авторы представляют модель машинного обучения, основанную на алгоритме Random forest (случайный лес), который применяется для задач классификации, регрессии и кластеризации. Рассматривается модель классификации для прогнозирования аварийного завершения заданий. В [8] предлагается метод обнаружения аномалий и отклонений в производительности в суперкомпьютерах и облачных системах. Однако в этом исследовании применяются искусственно созданные аномалии, внедренные в статистику работы суперкомпьютера, и следует учитывать, что в реальной ситуации характер аномалий может быть другим. В исследовании [9] для прогнозирования времени выполнения и использования ресурсов ввода-вывода заданий на основе их сценариев запуска применяются несколько алгоритмов машинного обучения (k ближайших соседей, дерево решений, случайный лес). С использованием аналогичных алгоритмов машинного обучения, на основе паспортов заданий и статистики работы системы планирования в статье [10] прогнозируются время выполнения, время ожидания в очереди и использование оперативной памяти заданиями. В работе [11] приведено исследование работы алгоритмов машинного обучения на предмет возможности предупреждения (на раннем этапе исполнения задания) о вероятной невозможности выполнения задания в запрошенное пользователем время.

К настоящему моменту с помощью алгоритмов машинного обучения проведено множество исследований статистики работы суперкомпьютерной системы коллективного пользования на предмет выявления аномалий и прогнозирования различных параметров задания. В настоящей работе исследуется возможность использования статистики работы суперкомпьютера в формате Standard Workload Format (SWF) [12, 13] для классификации заданий по времени выполнения и прогнозирования попадания задания в тот или иной класс по характеристикам задания.

Анализ исходных данных и отбор признаков

В качестве исходного набора данных использовались 100 тыс. записей из SWF-файла загрузки системы RICC (RIKEN Integrated Cluster of Clusters) [14] за 2010 год (RICC-2010-2.swf) [15]. Эти данные отражают работу суперкомпьютера, состоящего из 1 024 узлов, каждый из которых имеет по два 4-ядерных процессора и по 12 Гб ОЗУ.

В формате SWF каждое задание описывается одной строкой, содержащей в полях такие сведения, как номер задания, время его постановки и ожидания в очереди, запрошенное и реальное время выполнения, количество запрошенных и выделенных процессоров, утилизация процессорного времени, запрошенный и

использованный объем памяти, статус завершения задания, идентификаторы пользователя и группы и другие.

Одной из важных процедур предобработки данных в алгоритмах их анализа является отбор значимых признаков. Его цель заключается в том, чтобы отобрать наиболее существенные признаки для решения рассматриваемой задачи классификации. Отбор признаков позволяет:

– достигнуть лучшего понимания задачи, поскольку человеку легче разобраться с небольшим количеством признаков;

– ускорить работу алгоритмов обучения, анализа и прогнозирования;

– повысить качество прогнозирования за счет устранения шумовых признаков и уменьшения ошибки алгоритма на тестовой выборке.

Отбор значимых признаков может осуществляться как вручную – на основе анализа содержательной постановки задачи, так и автоматически – с помощью универсальных алгоритмов.

Из набора полей строки SWF-формата необходимо выделить признаки, которые будут участвовать в машинном обучении и прогнозировании. Изначально все множество полей из SWF-формата является избыточным, соответственно, необходимо провести выбор наиболее информативных, отличительных и независимых признаков. В первую очередь исключаются признаки, у которых значение среднеквадратического отклонения составляет 0, то есть значение этих признаков является статическим и не изменяется, сюда попадают многочисленные незаполненные поля SWF-файла. На этом этапе были отсеяны такие признаки, как утилизация процессорного времени, использованный объем памяти, номера раздела, исполняемого приложения и предшествующего задания, время ожидания после завершения предшествующего задания. Далее из набора признаков была исключена информация, недоступная на момент постановки задания в очередь и, соответственно, никак не влияющая на прогноз времени выполнения задания: номер задания, время ожидания в очереди, количество выделенных процессоров и статус завершения задания.

Признак «время постановки задания в очередь» (SubmitTime) в формате времени UNIX был преобразован в формат, указывающий только час поступления задания в очередь. Время суток, в которое задание отправлено на исполнение, может быть полезным при прогнозировании, так как более короткие тестовые задания могут выполняться в начале рабочего дня, более длительные ставиться в очередь ближе к его концу.

На рисунке 1 представлена матрица корреляции следующих признаков: время постановки задания в очередь (SubmitTime), запрошенное (RequestedTime) и фактическое (RunTime) время выполнения задания, число запрошенных процессоров (RequestedCPU), объем запрошенной памяти (RequestedMemory), идентификаторы пользователя (UserID) и группы (GroupID), номер очереди (QueueNum), час постановки задания в очередь (Hour).

Как видно из рисунка 1, фактическое время выполнения (RunTime) в наибольшей степени зависит от запрошенного времени выполнения (RequestedTime), идентификатора пользователя (UserID) и времени суток (Hour).

На рисунке (см. http://www.swsys.ru/uploaded/image/2020-2/2020-2-dop/14.jpg) показано отношение запрошенного на выполнение

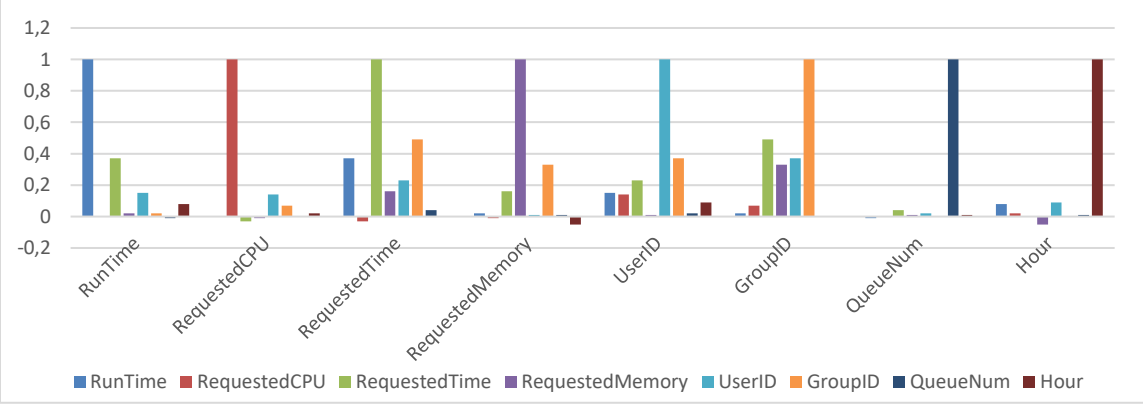

Рис. 1. Матрица корреляции отобранных признаков

Fig. 1. Selected feature correlation matrix

задания времени (RequestedTime) к фактическому (RunTime). Относительно точные запросы времени выполнения на диаграмме располагаются недалеко от диагонали. В данном случае видно, что у подавляющей части заданий запрошенное время выполнения значительно превосходит фактическое, большое число заданий запрашивают максимально возможное время (259,2 тыс. сек.) независимо от фактически требуемого (сплошная линия в верхней части диаграммы). Часть заданий не укладываются в запрошенное время и продолжают работу. Видно, что СУЗ RICC не завершает такие задания принудительно, на диаграмме эти задания расположены ниже диагональной линии.

Классификация заданий и подготовка признаков к машинному обучению

Определим отношение фактически затраченного времени на выполнение задания к запрошенному времени как

$$Ratio = \frac{RequstedTime}{RunTime}. \quad (1)$$

Добавим это отношение в качестве дополнительного признака к отобранным. Анализ признака показал, что максимальное превышение запрошенного времени составляет 86 400 раз, то есть превышение на 5 порядков. Кроме этого, в выборке [15] присутствуют задания без указания требуемого времени выполнения. Напомним, что, чем меньше разрыв между фактическим временем выполнения задания и запрошенным, тем более эффективной будет работа планировщика СУЗ суперкомпьютера. По этой причине будем классифицировать задания по превышению запрошенного времени по сравнению с фактически затраченным. Нулевой класс образуют задания, у которых отношение (1) менее 1, то есть эти задания не уложились в запрошенное время. В каждом последующем классе заданий отношение фактически затраченного времени на выполнение к запрошенному на порядок увеличивается по сравнению с предыдущим классом. Например, класс 2 образуют задания, фактическое время выполнения которых превышает запрошенное от 10 до 100 раз. Распределение заданий по классам для рассматриваемой выборки [15] приведено на рисунке 2.

Предварительное прогнозирование принадлежности задания к тому или иному классу на этапе его размещения в очереди позволит получить более точное представление о длительности выполнения задания и тем самым повысить эффективность планирования СУЗ.

Разделим отобранные признаки на числовые и категориальные. К числовым признакам отнесем запрошенные количество процессоров RequestedCPU, время RequestedTime, объем памяти RequestedMemory, а также время суток Hour. К категориальным – идентификаторы пользователя UserID и группы GroupID, номер очереди QueueNum.

Категориальные признаки перед подачей на вход алгоритмов машинного обучения необходимо преобразовать в числовые. Перечисленные категориальные признаки не являются бинарными, поэтому для их преобразования применяется метод векторизации. Признак *j*, принимающий *s* значений, заменяется на *s* признаков, принимающих значения 0 или 1 в зависимости от того, чему равно значение исходного признака *j*.

Следует учитывать, что многие алгоритмы машинного обучения чувствительны к масштабированию данных. Поэтому перед построением моделей числовые признаки следует нормировать, так как результаты анализа не должны зависеть от выбора единиц измерения. Если не делать нормировку, то признаки с более мелкой мерой получат больший диапазон значений и, как следствие, большие веса. Для рассматриваемого набора данных применим нормировку: $x_i^{'} = \frac{x_i - \min(x)}{\max(x) - \min(x)}$, где минимальная и максимальная величины вычисляются автоматически по предоставленным данным.

В результате предварительной обработки исходных данных были получены матрицы размерностью 100 000×214 в случае использования всех имеющихся признаков и размерно-

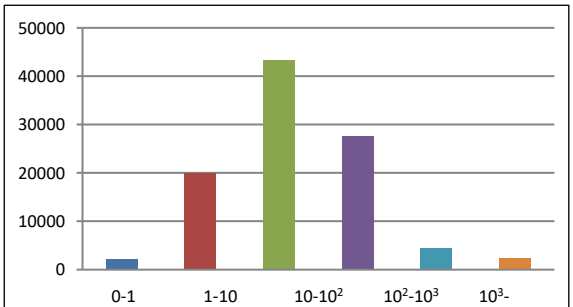

Рис. 2. Распределение заданий по классам для рассматриваемой выборки

Fig. 2. Job distribution by class for the sample selection

стью 100 000×119 в случае использования трех признаков с наибольшей корреляцией с фактическим временем выполнения задания (RequestedTime, UserID, Hour).

Прогнозирование с помощью моделей машинного обучения

Для построения моделей была использована программная библиотека Scikit-Learn [16], предоставляющая реализацию целого ряда алгоритмов для обучения с учителем (Supervised Learning) и без учителя (Unsupervised Learning) через интерфейс для языка программирования Python. Библиотека распространяется под лицензией Simplified BSD License и имеет дистрибутивы для множества различных платформ. Библиотека Scikit-Learn ориентирована в первую очередь на моделирование данных.

Построение моделей проводилось на обучающей выборке, а проверка качества построенной модели – на тестовой. После того как модель обучена, появляется возможность прогнозирования значения целевого признака по входным признакам для новых объектов.

Сравним точность прогнозирования следующих алгоритмов машинного обучения:

- логистическая регрессия;
- дерево решений;
- k ближайших соседей;
- линейный дискриминантный анализ;
- машина опорных векторов.

Первая часть алгоритмов относится к так называемым базовым (отдельным). Логистическая регрессия [17] – это способ определения зависимости между переменными, одна из которых категориально зависима, а другие независимы. Для этого применяется логистическая функция (аккумулятивное логистическое распределение). Практическое значение логистической регрессии заключается в том, что она является мощным статистическим методом предсказания значений зависимой переменной на основе значений одной или нескольких независимых переменных. Дерево принятия решений [18] – это метод поддержки принятия решений, основанный на использовании древовидного графа: модели принятия решений, которая учитывает их потенциальные последствия (с расчетом вероятности наступления того или иного события), эффективность, ресурсозатратность. Метрический алгоритм k ближайших соседей [19] служит для автоматической классификации объектов или регрессии. В случае использования метода для классификации объект присваивается классу, являющемуся наиболее распространенным среди k соседей данного элемента, классы которых уже известны. В случае использования метода для регрессии объекту присваивается среднее значение по k ближайшим к нему объектам, значения которых уже известны. Линейный дискриминантный анализ [20] является обобщением линейного дискриминанта Фишера, метода, используемого в статистике, распознавании образов и машинном обучении для поиска линейной комбинации признаков, которая описывает или разделяет два или более классов или событий. Получившуюся комбинацию можно использовать как в качестве линейного классификатора, так и для снижения размерности перед классификацией. Метод опорных векторов – это набор схожих алгоритмов обучения с учителем, использующихся для задач классификации и регрессионного анализа [21]. Принадлежит семейству линейных классификаторов и может также рассматриваться как специальный случай регуляризации по Тихонову. Особым свойством метода опорных векторов является непрерывное уменьшение эмпирической ошибки классификации и увеличение зазора, поэтому метод также известен как метод классификатора с максимальным зазором. Основная идея метода – перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве. Две параллельные гиперплоскости строятся по обеим сторонам гиперплоскости, разделяющей классы. Разделяющей гиперплоскостью будет гиперплоскость, максимизирующая расстояние до двух параллельных гиперплоскостей. Алгоритм работает в предположении, что, чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора.

Вторая часть алгоритмов относится к так называемым ансамблевым. Случайный лес – алгоритм машинного обучения, заключающийся в использовании комитета (ансамбля) решающих деревьев [22]. Алгоритм применяется для задач классификации, регрессии и кластеризации. Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе дает очень невысокое качество классификации, но за счет их большого количества результат получается хорошим. Градиентный бустинг – это техника машинного обучения для задач класси-

Таблица 1

Вероятность верного прогноза порядка времени выполнения задания для базовых алгоритмов

Table 1

The correct forecast probability of job-performance time for basic algorithms

Выборка	Логистическая регрессия	Дерево решений	k ближайших соседей	Линейный дискриминантный анализ	Метод опорных векторов
Обучающая № 1	0,60	0,84	0,81	0,58	0,59
Тестовая № 1	0,60	0,83	0,81	0,59	0,59
Обучающая № 2	0,60	0,83	0,79	0,58	0,59
Тестовая № 2	0,60	0,82	0,79	0,58	0,59

фикации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений [23]. На каждой итерации вычисляются отклонения предсказаний уже обученного ансамбля на обучающей выборке. Следующая добавленная в ансамбль модель будет предсказывать эти отклонения. Таким образом, добавив предсказания нового дерева к предсказаниям обученного ансамбля, мы можем уменьшить среднее отклонение модели. Новые деревья добавляются в ансамбль до тех пор, пока ошибка уменьшается либо пока не выполнится одно из правил «ранней остановки».

Результаты работы базовых алгоритмов машинного обучения по классификации заданий приведены в таблице 1. Выборка 1 содержит все доступные на момент размещения задания в очередь признаки, выборка 2 содержит только три признака с наибольшей корреляцией – запрошенное время выполнения (RequestedTime), идентификатор пользователя (UserID) и время суток (Hour). Полная размерность выборки 1 составила 100 000×214, выборки 2 – 100 000×119. Выборки были разбиты на обучающие и тестовые в соотношении 70 % и 30 % с рандомизацией.

Как видно из таблицы 1, наибольшую вероятность правильного прогноза порядка времени выполнения задания обеспечивают алгоритмы дерева решений и k ближайших соседей. Точность работы этих алгоритмов повышается при наличии дополнительных признаков.

Результаты работы ансамблевых алгоритмов машинного обучения по классификации заданий приведены в таблице 2.

Ансамблевые алгоритмы показали себя более мощными инструментами по сравнению с базовыми моделями прогнозирования, поскольку:

– сводят к минимуму влияние случайностей, усредняя ошибки каждого базового классификатора;

– уменьшают дисперсию, поскольку несколько разных моделей, исходящих из разных гипотез, имеют больше шансов прийти к правильному результату, чем одна отдельно взятая;

– исключают выход за рамки множества: если агрегированная гипотеза оказывается вне множества базовых гипотез, то на этапе формирования комбинированной гипотезы при помощи того или иного способа множество расширяется так, чтобы гипотеза входила в него.

Таблица 2

Вероятность верного прогноза порядка времени выполнения задания для ансамблевых алгоритмов

Table 2

The correct forecast probability of job-performance time for ensemble algorithms

Выборка	Случайный лес	Градиентный бустинг
Обучающая № 1	0,83	0,78
Тестовая № 1	0,83	0,78
Обучающая № 2	0,82	0,78
Тестовая № 2	0,82	0,78

Для ансамблевых алгоритмов было проведено ранжирование признаков по важности, представленное на рисунке 3.

Как видно, в обоих случаях совпадает набор из первых трех признаков, но порядок важности признаков отличается.

Прогнозирование с помощью нейронных сетей прямого распространения

Нейронные сети прямого распространения, или многослойные персептроны, являются хо-

рошо известными многоуровневыми классификаторами логистической регрессии [24]. Нейроны в таких сетях расположены слоями и имеют однонаправленные связи между слоями. Сети прямого распространения являются статическими в том смысле, что на заданный вход они вырабатывают одну совокупность выходных значений, не зависящих от предыдущего состояния сети.

Для прогнозирования времени выполнения заданий была создана нейронная сеть, состоящая из трех слоев. Входной слой содержит 50 нейронов с функцией активации «гиперболический тангенс» и принимает на вход полный набор из 214 признаков. Скрытый слой содержит 100 нейронов с функцией активации «гиперболический тангенс». Выходной слой состоит из 6 нейронов с функцией активации softmax. Softmax – это обобщение логистической функции для многомерного случая. Функция преобразует вектор z размерности K в вектор s той же размерности, где каждая координата s_i полученного вектора представлена вещественным числом в интервале [0, 1] и сумма координат равна 1. Таким образом, на выходе каждого нейрона выходного слоя будет значение вероятности принадлежности задания к одному из 6 ранее определенных классов.

Обучение созданной нейронной сети проводилось на том же наборе данных на протяжении 200 эпох. Кривые точности модели и функции потерь в процессе обучения представлены на рисунке 4.

Приведем результаты точности прогнозирования нейронной сети прямого распространения по классификации заданий: обучающая выборка 1 – 0,83, тестовая выборка 1 – 0,83, обучающая выборка 2 – 0,81, тестовая выборка 2 – 0,81.

Ранжирование признаков по важности для случая нейронной сети прямого распространения приведено в таблице 3.

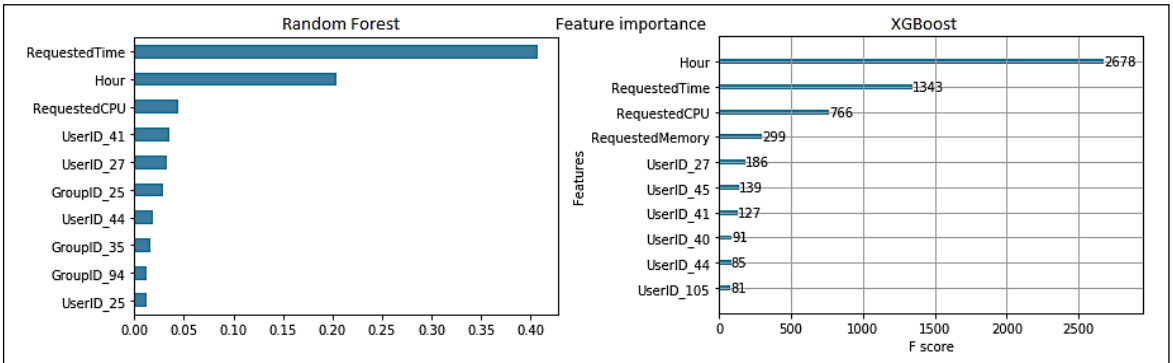

Рис. 3. Ранжирование признаков по важности для алгоритмов случайного леса (Random Forest) и градиентого бустинга (XGBoost)

Fig. 3. Features ranking by importance for the random forest algorithms (Random Forest) and gradient boosting (XGBoost)

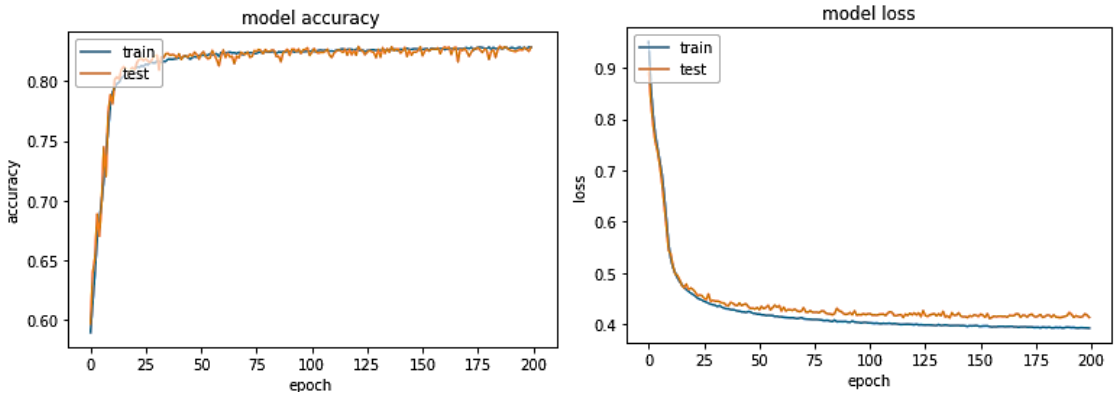

Рис. 4. Кривые точности модели и функции потерь в процессе обучения нейронной сети

Fig. 4. The model accuracy curves and loss functions in the neural network training process

Таблица 3

Ранжирование признаков по важности для нейронной сети прямого распространения

Table 3

Ranking features by importance for a feedforward neural network

Признак	Важность (вес)
RequestedTime	3,4722 ± 0,0358
UserID_27	1,3669 ± 0,0085
GroupID_25	1,3261 ± 0,0079
Hour	0,8419 ± 0,0105
UserID_41	0,7907 ± 0,0077
GroupID_94	0,6142 ± 0,0067
RequestedCPU	0,5172 ± 0,0050
GroupID_31	0,3862 ± 0,0064
UserID_38	0,3635 ± 0,0072
UserID_45	0,2707 ± 0,0061
GroupID_36	0,2662 ± 0,0063
GroupID_35	0,2619 ± 0,0038
GroupID_34	0,2532 ± 0,0083
GroupID_4	0,2328 ± 0,0030
UserID_44	0,2244 ± 0,0032
GroupID_99	0,2129 ± 0,0062
UserID_40	0,2059 ± 0,0074
RequestedMemory	0,1946 ± 0,0039
UserID_47	0,1812 ± 0,0016
GroupID_37	0,1698 ± 0,0022

Таблица 4 содержит сводные данные по качеству прогнозирования, включая время на обучение и прогнозирование (оценку).

Таблица 4

Сводные данные по алгоритмам прогнозирования

Table 4

Summary data by predictive algorithms

Алгоритм	Обучение	Оценка	Точность
Логистическая регрессия	23 сек.	<1 сек.	0,60
Дерево решений	<1 сек.	<1 сек.	0,83
k ближайших соседей	12 сек.	8 мин.	0,81
Линейный дискриминантный анализ	3 сек.	<1 сек.	0,59
Метод опорных векторов	15 мин.	15 мин.	0,59
Случайный лес	<1 сек.	<1 сек.	0,83
Градиентный бустинг	28 сек.	<1 сек.	0,78
Нейронная сеть прямого распространения	30 мин.	<1 сек.	0,83

На последнем этапе экспериментов было осуществлено сужение границ классов заданий до двоичного порядка, то есть соотношение (1) для некоторого класса заданий было в два (не в десять) раза больше, чем для предыдущего. При таком разбиении на классы ожидаемо снизилась вероятность правильного прогноза, как показано в таблице 5.

Таблица 5

Вероятность правильного прогноза двоичного порядка времени выполнения заданий

Table 5

The correct forecast probability binary exponent time for job execution

Выборка	Случайный лес	Градиентный бустинг	Нейронная сеть прямого распространения
Обучающая №1	0,70	0,64	0,69
Тестовая № 1	0,70	0,64	0,69

Заключение

Представленное в настоящей статье исследование показало возможности использования алгоритмов машинного обучения для прогнозирования времени выполнения заданий на основе статистики работы суперкомпьютерной системы коллективного пользования и данных сформированного пользователем паспорта задания. Результаты анализа статистических данных работы суперкомпьютера RIKEN Integrated Cluster of Clusters (RICC), представленных в формате SWF, позволили отобрать значимые признаки потока заданий, было проведено ранжирование признаков по важности и обнаружены скрытые закономерности, влияющие на точность прогнозирования.

Все задания были разделены на шесть классов, основой классификации послужило отношение фактического времени выполнения задания к запрошенному. В каждом классе это отношение возрастало на порядок. Прогнозирование времени выполнения задания осуществлялось путем отнесения задания к одному из классов. Для распространенных алгоритмов машинного обучения были получены оценки вероятности верного прогноза. Наилучшие значения показали алгоритмы дерева решений, случайного леса и нейронные сети прямого распространения.

К недостаткам статистической выборки суперкомпьютера RICC следует отнести скудное информационное наполнение, например, незаполненные поля «номер исполняемого приложения» и «номер раздела» могли бы стать значимыми признаками и иметь заметную корреляцию с фактическим временем выполнения задания.

В дальнейших исследованиях предполагается использовать более полную по сравнению с форматом SWF статистическую информацию работы суперкомпьютера. Это позволит выделить большее количество важных признаков и, соответственно, повысить точность прогнозирования времени выполнения заданий.

Работа выполнена в МСЦ РАН в рамках проекта по гранту РФФИ № 18-29-03236.

Литература

1. Reuther A., Byun C., Arcand W., Bestor D., Bergeron B., Hubbell M., Jones M., Michaleas P., Prout A., Rosa A., Kepner J. Scalable system scheduling for HPC and big data. J. of Parallel and Distributed Computing, 2018, vol. 111, pp. 76–92. DOI: 10.1016/j.jpdc.2017.06.009.

2. Yoo A.B., Jette M.A., Grondona M. SLURM: simple Linux utility for resource management. Proc. JSSPP, LNCS, 2003, vol. 2862, pp. 44–60. DOI: 10.1007/10968987_3.

3. Henderson R.L. Job scheduling under the portable batch system. Proc. JSSPP, LNCS, 1995, vol. 949, pp. 279–294. DOI: 10.1007/3-540-60153-8_34.

4. IBM spectrum LSF overview. URL: https://www.ibm.com/support/knowledgecenter/en/SSWRJV_10.1.0/lsf_foundations/chap_lsf_overview_foundations.html (дата обращения: 06.02.2020).

5. Баранов А.В., Киселев Е.А., Ляховец Д.С. Квазипланировщик для использования простаивающих вычислительных модулей многопроцессорной вычислительной системы под управлением СУППЗ // Вестн. ЮУрГУ. 2014. Т. 3. № 4. С. 75–84. DOI: 10.14529/cmse140405.

6. Klinkenberg J., Terboven C., Lankes S., Müller M.S. Data mining-based analysis of hpc center operations. Proc. IEEE Intern. Conf. on Cluster Computing, Honolulu, HI, 2017, pp. 766–773. DOI: 10.1109/CLUSTER.2017.23.

7. Yoo W., Sim A., Wu K. Machine learning based job status prediction in scientific clusters. Proc. Computing Conf. SAI, London, 2016, pp. 44–53. DOI: 10.1109/SAI.2016.7555961.

8. Tuncer O., Ates E., Zhang Y., Turk A., Brandt J., Leung V.J., Egele M., Coskun A.K. Diagnosing performance variations in HPC applications using machine learning. Proc. ISC, 2017, vol. 10266, pp. 355–373. DOI: 10.1007/978-3-319-58667-0_19.

9. McKenna R., Herbein S., Moody A., Gamblin T., Taufer M. Machine learning predictions of runtime and IO traffic on high-end clusters. Proc. IEEE Intern. Conf. on Cluster Computing, Taipei, 2016, pp. 255–258. DOI: 10.1109/CLUSTER.2016.58.

10. Rodrigues E.R., Cunha R.L.F., Netto M.A.S., Spriggs M. Helping HPC users specify job memory requirements via machine learning. Proc. 3rd Intern. Workshop on HUST, Salt Lake City, 2016, pp. 6–13. DOI: 10.1109/HUST.2016.006.

11. Guo J., Nomura A., Barton R., Zhang H., Matsuoka S. Machine learning predictions for underestimation of job runtime on HPC system. Proc. SCFA, 2018, vol. 10776, pp. 179–198. DOI: 10.1007/978-3-319-69953-0_11.

12. Standard Workload Format. URL: http://www.cs.huji.ac.il/labs/parallel/workload/swf.html (дата обращения: 12.02.2020).

13. Chapin S.J., Cirne W., Feitelson D.G., Jones J.P., Leutenegger S.T., Schwiegelshohn U., Smith W., Talby D. Benchmarks and standards for the evaluation of parallel job schedulers. Proc. SCFA, 1999, vol. 1659, pp. 67–90. DOI: 10.1007/3-540-47954-6_4.

14. Nakata M. All about RIKEN integrated cluster of clusters (RICC). IJNC, 2012, vol. 2, iss. 2, pp. 206–215. DOI: 10.15803/ijnc.2.2_206.

15. The RICC log. Parallel Workloads Archive. 2010. URL: https://www.cse.huji.ac.il/labs/parallel/workload/l_ricc/index.html (дата обращения: 07.02.2020).

16. Paper D. Introduction to Scikit-Learn. In: Hands-on Scikit-Learn for machine learning applications. Apress, 2020, pp. 1–35. DOI: 10.1007/978-1-4842-5373-1_1.

17. Tolles J., Meurer W.J. Logistic regression: relating patient characteristics to outcomes. JAMA, 2016, vol. 316, no. 5, pp. 533–534. DOI: 10.1001/jama.2016.7653.

18. Левитин А.В. Алгоритмы. Введение в разработку и анализ. М.: Вильямс, 2006. С. 409–417.

19. Altman N. An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 1992, vol. 46, no. 3, pp. 175–185. DOI: 10.2307/2685209.

20. McLachlan G.J. Discriminant analysis and statistical pattern recognition. Wiley InterScience, 1992, 545 p. DOI: 10.1002/0471725293.

21. Вьюгин В. Математические основы теории машинного обучения и прогнозирования. М., 2013. 390 с.

22. Breiman L. Random Forests. Machine Learning, 2001, vol. 45, pp. 5–32.

23. Friedman J.H. Greedy function approximation: a gradient boosting machine. Ann. Statist., 2001, vol. 29, no. 5, pp. 1189–1232. DOI: 10.1214/aos/1013203451.

24. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика. М.: Горячая линия–Телеком, 2001. 382 с.

Software & Systems
DOI: 10.15827/0236-235X.130.218-228

Received 27.02.20
2020, vol. 33, no. 2, pp. 218–228

Machine learning to predict the supercomputer jobs execution time

A.V. Baranov [1]*, Ph.D. (Engineering), Associate Professor, Leading Researcher, antbar@mail.ru, abaranov@jscc.ru*
D.S. Nikolaev [2]*, Senior Researcher, dmitry.s.nikolaev@gmail.com*

[1] *Joint Supercomputer Center of the RAS – branch of Federal State Institution «Scientific Research Institute for System Analysis of the RAS» (SRISA RAS), Moscow, 119334, Russian Federation*
[2] *Research & Development Institute «Kvant», Moscow, 125438, Russian Federation*

Abstract. The authors devoted the paper to machine learning methods and algorithms for the supercomputer job execution time predicting. The supercomputer systems statistics for multiple-access shows that the actual execution time for most jobs substantially diverges from the time requested by the user. This reduces the scheduling job efficiency because inaccurate estimates of execution time lead to a suboptimal schedule for job launches.

The paper considers the job classification, which is based on actual time for job performance to the requested one. There were six job classes moreover the ratio of the actual job completion time to the requested time for each class differs by an order of magnitude from the previous class. Statistical data from a shared-use supercomputer system is the basis for predicting the job completion time by assigning the incoming job to one of the classes. The supercomputer RIKEN Integrated Cluster of Clusters (RICC) statistics presented in SWF format were as initial data. The analyzing results for these statistics allowed us to identify significant features of the job flow for machine learning. There was a feature ranking by importance and there were hidden patterns that affect the forecasting accuracy, in particular, there was a mutual correlation for selected features.

For common machine learning algorithms, such as logistic regression, decision tree, k nearest neighbors, linear discriminant analysis, support vector method, random forest, gradient boosting, and direct propagation neural network, there were estimates of the probability of a correct forecast. Algorithms of the decision tree, random forest, and direct propagation neural networks showed the best values.

Keywords: high performance computing, grid, job management system, supercomputer job scheduling, machine learning, job complete time prediction.

Acknowledgements. *The JSCC RAS has done the work was within the RFBR project, grant no. 18-29-03236.*

References

1. Reuther A., Byun C., Arcand W., Bestor D., Bergeron B., Hubbell M., Jones M., Michaleas P., Prout A., Rosa A., Kepner J. Scalable system scheduling for HPC and big data. *J. of Parallel and Distributed Computing*, 2018, vol. 111, pp. 76–92. DOI: 10.1016/j.jpdc.2017.06.009.

2. Yoo A.B., Jette M.A., Grondona M. SLURM: simple Linux utility for resource management. *Proc. JSSPP*, LNCS, 2003, vol. 2862, pp. 44–60. DOI: 10.1007/10968987_3.

3. Henderson R.L. Job scheduling under the portable batch system. *Proc. JSSPP*, *LNCS*, 1995, vol. 949, pp. 279–294. DOI: 10.1007/3-540-60153-8_34.

4. *IBM Spectrum LSF Overview*. Available at: https://www.ibm.com/support/knowledgecenter/en/SSWRJV_10.1.0/lsf_foundations/chap_lsf_overview_foundations.html (accessed 06.02.2020).

5. Baranov A.V., Kiselev E.A., Lyakhovets D.S. The quasi scheduler for utilization of multiprocessing computing system"s idle resources under control of the management system of the parallel jobs. *Bull. of the*

South Ural State Univ., *Comp. Math. and Soft. Eng*., 2014, vol. 3, no. 4, pp. 75–84 (in Russ.). DOI: 10.14529/cmse140405.

6. Klinkenberg J., Terboven C., Lankes S., Müller M.S. Data mining-based analysis of hpc center operations. *Proc. IEEE Intern. Conf. on Cluster Computing*, Honolulu, HI, 2017, pp. 766–773. DOI: 10.1109/CLUSTER.2017.23.

7. Yoo W., Sim A., Wu K. Machine learning based job status prediction in scientific clusters. *Proc. Computing Conf. SAI*, London, 2016, pp. 44–53. DOI: 10.1109/SAI.2016.7555961.

8. Tuncer O., Ates E., Zhang Y., Turk A., Brandt J., Leung V.J., Egele M., Coskun A.K. Diagnosing performance variations in HPC applications using machine learning. *Proc. ISC*, 2017, vol. 10266, pp. 355–373. DOI: 10.1007/978-3-319-58667-0_19.

9. McKenna R., Herbein S., Moody A., Gamblin T., Taufer M. Machine learning predictions of runtime and IO traffic on high-end clusters. Proc. *IEEE Intern. Conf. on Cluster Computing*, Taipei, 2016, pp. 255–258. DOI: 10.1109/CLUSTER.2016.58.

10. Rodrigues E.R., Cunha R.L.F., Netto M.A.S., Spriggs M. Helping HPC users specify job memory requirements via machine learning. *Proc. 3rd Intern. Workshop on HUST*, Salt Lake City, 2016, pp. 6–13. DOI: 10.1109/HUST.2016.006.

11. Guo J., Nomura A., Barton R., Zhang H., Matsuoka S. Machine learning predictions for underestimation of job runtime on HPC system. *Proc. SCFA*, 2018, vol. 10776, pp. 179–198. DOI: 10.1007/978-3-319-69953-0_11.

12. *Standard Workload Format*. Available at: http://www.cs.huji.ac.il/labs/parallel/workload/swf.html (accessed February 12, 2020).

13. Chapin S.J., Cirne W., Feitelson D.G., Jones J.P., Leutenegger S.T., Schwiegelshohn U., Smith W., Talby D. Benchmarks and standards for the evaluation of parallel job schedulers. *Proc. SCFA*, 1999, vol. 1659, pp. 67–90. DOI: 10.1007/3-540-47954-6_4.

14. Nakata M. All about RIKEN integrated cluster of clusters (RICC). *IJNC*, 2012, vol. 2, iss. 2, pp. 206–215. DOI: 10.15803/ijnc.2.2_206.

15. *The RICC log. Parallel Workloads Archive*. 2010. Available at: https://www.cse.huji.ac.il/labs/parallel/workload/l_ricc/index.html (accessed February 07, 2020).

16. Paper D. Introduction to Scikit-Learn. In: *Hands-on Scikit-Learn for Machine Learning Applications*. Apress, 2020, pp. 1–35. DOI: 10.1007/978-1-4842-5373-1_1.

17. Tolles J., Meurer W.J. Logistic regression: relating patient characteristics to outcomes. *JAMA*, 2016, vol. 316, no. 5, pp. 533–534. DOI: 10.1001/jama.2016.7653.

18. Levitin A. *Algorithms. Introduction to the Design and Analysis*. Moscow, 2006, pp. 409–417 (in Russ.).

19. Altman N. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 1992, vol. 46, no. 3, pp. 175–185. DOI: 10.2307/2685209.

20. McLachlan G.J. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley InterScience, 1992, 545 p. DOI: 10.1002/0471725293.

21. Vyugin V. *Mathematical Foundations of the Theory of Machine Learning and Forecasting*. Moscow, 2013, 390 p. (in Russ.).

22. Breiman L. Random Forests. *Machine Learning*, 2001, vol. 45, pp. 5–32.

23. Friedman J.H. Greedy function approximation: A gradient boosting machine. *Ann. Statist*., 2001, vol. 29, no. 5, pp. 1189–1232. DOI: 10.1214/aos/1013203451.

24. Kruglov V.V., Borisov V.V. *Artificial Neural Networks. Theory and Practice*. Moscow, 2001, 382 p.

Для цитирования

Баранов А.В., Николаев Д.С. Применение машинного обучения для прогнозирования времени выполнения суперкомпьютерных заданий // Программные продукты и системы. 2020. Т. 33. № 2. С. 218–228. DOI: 10.15827/0236-235X.130.218-228.

For citation

Baranov A.V., Nikolaev D.S. Machine learning to predict the supercomputer jobs execution time. *Software & Systems*, 2020, vol. 33, no. 2, pp. 218–228 (in Russ.). DOI: 10.15827/0236-235X.130.218-228.