УДК 004.93
DOI: 10.15827/0236-235X.118.2.207-216

Дата подачи статьи: 03.04.17
2017. Т. 30. № 2. С. 207–216

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ ПРИ РЕАЛИЗАЦИИ WEB-ИНСТРУМЕНТАРИЯ РАСПОЗНАВАНИЯ ОБРАЗОВ НА ОСНОВЕ МЕТОДОВ ПРЕЦЕДЕНТОВ

*В.В. Фомин, д.т.н., профессор, v_v_fomin@mail.ru;
И.В. Александров, аспирант, chrono555@yandex.ru
(Российский государственный педагогический университет им. А.И. Герцена,
наб. реки Мойки, 48, г. Санкт-Петербург, 191186, Россия)*

Предлагается программное решение, направленное на улучшение качества распознавания образов и повышение эффективности инструментария машинного обучения посредством реализации grid-технологий. Формулируются стратегические направления разработки инструментария распознавания образов в виде программной системы, основанной на принципах распределенных систем, распараллеливания и адаптивной настройки вычислительных ресурсов. Рассматривается структура организации web-инструментария распознавания образов с использованием концепции библиотеки алгоритмов. Даются алгоритмические решения распараллеливания алгоритмов обучения и распознавания на базе классических методов интеллектуального анализа данных, которые хорошо зарекомендовали себя на практике. К таким алгоритмам относятся методы прецедентов или методы, основанные на метриках близости. Они обладают большим потенциалом к распараллеливанию вычислительных процессов и разработке параллельных алгоритмов их реализации.

Поиски путей повышения производительности вычислительной техники, особенно при реализации web-инструментария на основе ресурсоемких вычислительных алгоритмов машинного распознавания и прогнозирования, привели к решению создать grid-систему. Рассматриваемые в статье архитектура и реализация grid-системы предполагают распараллеливание и организацию распределенных вычислений на многомашинной основе с применением интернет-технологий, что позволяет получить практически те же вычислительные мощности, что и на многопроцессорных компьютерных системах, но с гораздо меньшей стоимостью.

В статье решается задача повышения эффективности вычислительных ресурсов при возможности реконфигурирования структуры интернет-соединений, в том числе процедуры настройки структуры вычислительной сети, подключаемых каналов связи и выделенных серверов в зависимости от исходных алгоритмов и данных. Представлены зависимости параметров времени выполнения операций от дисциплины обслуживания, адаптирующей систему к запросам пользователей. При этом задачи ранжируются по ресурсоемкости и под них выделяются соответствующие их рангу вычислительные мощности.

Ключевые слова: интеллектуальные информационные системы, распознавание образов, машинное обучение, web-системы, параллельные вычисления.

В современном мире рост производительности при обработке большого количества разнообразной информации достигается только в тех случаях, когда часть интеллектуальной нагрузки берут на себя компьютеры. Одним из направлений исследований в области искусственного интеллекта [1, 2] являются интеллектуальные информационные системы, которые находятся на пике исследовательской активности, практики прикладного применения и коммерциализации. Они широко вошли в разные области промышленности, экономики, медицины и т.д. Интеллектуальные информационные системы сосредоточили в себе наиболее наукоемкие технологии с высоким уровнем автоматизации процессов не только подготовки информации для принятия решений, но и выработки вариантов решений, опирающихся на полученные информационной системой данные.

Проблемы эффективного применения, использования, внедрения интеллектуальных систем определяют сложность подходов к классификации задач искусственного интеллекта и решаются в рамках различных направлений исследований, основанных на близких моделях, методах и алгоритмах. К таким исследованиям относятся распознава-

ние образов, машинное обучение, машинный перевод, интеллектуальный анализ данных и др. [2].

Одной из базовых апорий интеллектуального анализа данных [3] являются технологии распознавания образов [4–6]. Распознавание образов – класс задач, связанных с определением принадлежности объекта к одному из классов объектов. Распознавание образов можно рассматривать как классификацию объектов. Сложность и многофакторность проблемы распознавания образов, разнообразие проблемных областей и категорий задач, отсутствие универсальных методов решения задач распознавания приводят к тому, что традиционная методология разработки систем распознавания, как правило, не позволяет достигнуть требуемой эффективности результатов в рамках промышленного применения.

В настоящее время существует значительное противоречие, обусловленное уровнем развития формального аппарата теории распознавания образов, сравнительной ограниченностью и узкой направленностью большинства из существующих прикладных систем распознавания. Данное противоречие частично может быть преодолено путем разработки интегрированных многофункциональ-

ных автоматизированных информационных систем распознавания, ориентированных на решение широкого класса задач и допускающих адаптацию и настройку на новую предметную область и категорию задач.

Описание программы. В рамках исследований по тематике «Web-инструментарий машинного обучения» в качестве прототипа облачного ресурса разработана программа «Web-инструментарий распознавания образов на основе расширяющейся библиотеки методов машинного обучения» [7, 8]. Программа может быть использована для принятия решений широким кругом лиц, в том числе студентами, преподавателями, бизнесменами, специалистами в различных областях.

В базовой версии программы реализован класс методов прецедентов как пользующаяся популярностью классика в области data-mining. Методы прецедентов состоят из следующих циклов: 1) извлечение похожего прецедента из библиотеки; 2) использование выбранного прецедента для построения решения; 3) проверка полученного решения; 4) занесение решения как нового прецедента в библиотеку. Таким образом, системой накапливается опыт (прецеденты) и реализуется машинное обучение, что способствует повышению качества принимаемого решения [9].

Главным достоинством методов прецедентов является их высокая эффективность для множества задач с точки зрения достоверности результата и компактности алгоритмов анализа, а главным недостатком – требование значительных вычислительных ресурсов. Особенно это актуально при работе с большими объемами данных.

В качестве основных алгоритмов анализа в программе (рис. 1) были реализованы такие методы прецедентов, как метод k-ближайших соседей, метод потенциальных функций, метод эталонов, метод коррекционных приращений, метод наименьших среднеквадратических ошибок, метод наивный байесовский. Методы основаны на алгоритмах

метрической классификации и оценивании сходства объектов, для всех методов характерно наличие обучающей выборки.

Программа развивает концепцию облачных вычислений и осуществляет распознавание образов на основе методов машинного обучения посредством реализации web-сервиса и использования web-технологий. Компоненты web-технологий в программе реализованы с помощью сервлетов, которые расширяют функциональность web-серверов. Выбор в пользу сервлетов обусловлен тем, что они позволяют создавать платформенно-независимый код, предоставляют возможность масштабирования программы и могут выполняться параллельно в рамках одного процесса на сервере.

Алгоритмы распараллеливания методов прецедентов. Борьба за эффективное использование вычислительных ресурсов привела к появлению технологий распределенных систем и параллельных вычислений [10, 11], в частности, grid- и интернет-технологий. Grid-система строится по принципу клиент–сервер, состоит из одного или нескольких компьютеров-серверов и множества компьютеров-клиентов свободной конфигурации и представляет собой эффективную для распараллеливания структуру вычислительных ресурсов.

Так как у методов прецедентов наличие обучающей выборки обязательно, в их работе можно выделить два последовательных этапа: этап обучения и этап распознавания. Рассмотрим схемы реализации этих этапов на одном компьютере (базовом для расчетов компьютере K_0) более подробно. В рамках исследований множества обучающих и классифицируемых объектов представлены как отдельные файлы, а объекты в них – как строки атрибутов. Файл с обучающими объектами ($F_{\text{обуч.}}$) содержит w объектов (строк), а файл с классифицируемыми объектами ($F_{\text{классиф.}}$) имеет m объектов.

Этап обучения. Этап обучения на K_0 компьютере заключается в следующем: 1) компьютер K_0 поочередно считывает один из w обучающих объектов (q_k -й объект) из файла $F_{\text{обуч.}}$ и подает его на вход классификатора; 2) классификатор, получив q_k -й обучающий объект, строит на его основе решение с учетом меры близости применяемого метода прецедента. После построения решения на основе всех w объектов из файла $F_{\text{обуч.}}$ на выходе классификатора образуется система решающих правил, способная определить класс классифицируемого объекта.

Обозначим время обучения классификатора на q_k -м объекте как $T_{\text{обуч.}}(q_k)$. Тогда время обучения на основе файла $F_{\text{обуч.}}$ на K_0 компьютере рассчитывается как суммарное время

Рис. 1. Меню управления программы «Web-инструментарий машинного обучения»

Fig. 1. The program control menu "Machine learning web-tools"

обучения классификатора на w обучающих объектах:

$$T_{0 \text{ обуч.}} = \sum_{k=1}^w T_{0 \text{ обуч.}}(q_k). \quad (1)$$

Этап распознавания. Этап распознавания на K_0 компьютере (рис. 2) заключается в следующем: 1) компьютер K_0 поочередно занимается предобработкой одного из m классифицируемых объектов (x_k -й объект) из файла $F_{\text{классиф.}}$; обозначим этот этап как функцию $F_{\text{предобр.}}(x_k)$, а время, затраченное на предобработку x_k -го объекта, как $T_{\text{предобр.}}(x_k)$; 2) затем x_k -й объект поступает на вход системы решающих правил, которая определяет его класс (c_k); обозначим этот этап как функцию $F_{\text{опред.}}(x_k)$, а время, затраченное на определение класса x_k -го объекта, как $T_{\text{опред.}}(x_k)$; 3) в заключение класс x_k -го объекта (c_k) добавляется в выходной файл, этот этап обозначим как функцию $F_{\text{добав.}}(c_k)$, а время, затраченное на добавление c_k -го класса, как $T_{\text{добав.}}(c_k)$.

Тогда время распознавания на K_0 компьютере рассчитывается по формуле

$$T_{0 \text{ расп.}} = \sum_{k=1}^m (T_{0 \text{ предобр.}}(x_k) + T_{0 \text{ опред.}}(x_k) + T_{0 \text{ добав.}}(c_k)). \quad (2)$$

Или, сгруппировав время по функциям, перепишем формулу (2):

$$T_{0 \text{ расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \sum_{k=1}^m T_{0 \text{ опред.}}(x_k) + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (3)$$

Рассмотрим и проанализируем входящие в распознавание классифицируемых объектов процессы на K_0 компьютере более подробно.

Процесс предобработки (функция $F_{\text{предобр.}}(x_k)$) на K_0 компьютере состоит из двух этапов: 1) чтение x_k -го объекта из файла $F_{\text{классиф.}}$; обозначим чтение как функцию $F_{\text{чтен.}}(x_k)$, а время, затраченное на чтение x_k -го объекта, как $T_{\text{чтен.}}(x_k)$; 2) запись x_k -го объекта в выходной файл; обозначим запись как функцию $F_{\text{зап.}}(x_k)$, а время, затраченное на запись прочитанного x_k -го объекта в выходной файл, как $T_{\text{зап.}}(x_k)$. Другими словами, время предобработки x_k -го объекта на K_0 компьютере определяется по формуле $T_{0 \text{ предобр.}}(x_k) = T_{0 \text{ чтен.}}(x_k) + T_{0 \text{ зап.}}(x_k)$.

Процесс определения класса специфичен для каждого из методов прецедентов, следовательно, время определения класса будет зависеть от примененного метода прецедента.

Процесс добавления результатов распознавания x_k -го объекта в виде класса c_k в выходной файл (функция $F_{\text{добав.}}(c_k)$) на одном K_0 компьютере представляет собой функцию записи класса c_k в конец выходного файла и рассчитывается следующим образом: $T_{0 \text{ добав.}}(c_k) = T_{0 \text{ зап.}}(c_k)$.

Дополнительно введем два параметра: $T_{0 \text{ средн. опред.}}$ и $V_{0 \text{ средн. опред.}}$.

$T_{0 \text{ средн. опред.}}$ — среднее время определения класса на K_0 компьютере одного классифицируемого объ-

$$\text{екта: } T_{0 \text{ средн. опред.}} = \frac{\sum_{k=1}^m T_{0 \text{ опред.}}(x_k)}{m}.$$

$V_{0 \text{ средн. опред.}}$ — средняя скорость определения класса объектов на K_0 компьютере, то есть сколько классов объектов (число может быть нецелым) K_0 компьютер определяет в единицу времени:

$$V_{0 \text{ средн. опред.}} = \frac{1}{T_{0 \text{ средн. опред.}}}. \quad (4)$$

Стоит отметить, что $T_{0 \text{ средн. опред.}}$ и $V_{0 \text{ средн. опред.}}$ для всех объектов распознавания на K_0 компьютере одинаковы, потому что порядок и количество операций для определения класса любого объекта одинаково, а скорость выполнения операции на K_0 компьютере постоянна. Следовательно, чем больше скорость выполнения операции на K_0 компьютере (производительность), тем меньше $T_{0 \text{ средн. опред.}}$ и больше $V_{0 \text{ средн. опред.}}$.

С учетом формулы (4) формулу (3) можно переписать так:

$$T_{0 \text{ расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{m}{V_{0 \text{ средн. опред.}}} +$$

$$+ \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (5)$$

Реализация параллельной вычислительной системы. Анализ основных процессов этапа распознавания и формулы (2) позволяет сделать вывод, что распознавание x_k -го объекта никак не связано с распознаванием остальных классифицируемых

объектов. Таким образом, распознавание объектов можно делать параллельно на n компьютерах, а записывать результаты распознавания в один выходной файл на центральном сервере. Другими словами, подключив n компьютеров к K_0 , можно получить вычислительную сеть, где K_0 будет играть роль центрального сервера.

На рисунке 3 показана структура такой параллельной вычислительной системы из n компьютеров K_1, K_2, K_n и сервера K_0 , где функции $F_{\text{предобр.}}(x_k)$, $F_{\text{добав.}}(c_k)$ такие же, как в случае одного компьютера K_0 , потому что сервером является K_0 компьютер. Особую роль в вычислительной системе играет функция распределения t классифицируемых объектов между n компьютерами, то есть функция $P(m, n)$. Функция $P(m, n)$ по мере того, как компьютеры закончили обработку одного объекта, передает им по заданному алгоритму новый классифицируемый объект для распознавания. Совокупность объектов для обработки на K_i -м компьютере обозначим как $X_i = \langle x_{i1}; x_{i2}; \dots \rangle$.

Суть работы такой вычислительной системы заключается в следующем: 1) сервер поочередно занимается предобработкой одного из t классифицируемых объектов (x_k -й объект) из файла $F_{\text{классиф.}}$ и передает его функции $P(m, n)$; 2) функция $P(m, n)$ распределяет полученный x_k -й объект на K_i -й компьютер согласно алгоритму реализации функции $P(m, n)$; 3) после распознавания на выходе K_i -го компьютера образуется результат в виде класса c_k ; 4) сервер, получив класс c_k , добавляет его в выходной файл с помощью функции $F_{\text{добав.}}(c_k)$.

Необходимо учитывать, что скорость распознавания (V_i средн. опред.) у компьютеров может быть различной, так как компьютеры могут обладать разной производительностью. Отсортируем компьютеры по скорости V_i средн. опред. в порядке возрастания. Если скорости V_i средн. опред. у нескольких компьютеров одинаковые, будем сортировать их в порядке возрастания индексов i (компьютеры представлены в виде K_i).

Введем обозначение K_i мин. произ. для отсортированного по возрастанию скорости V_i средн. опред. ряда

компьютеров, а их скорость V_i средн. опред. обозначим V_i мин. произ. средн. опред. Например, K_1 мин. произ. — первый компьютер в ряду, то есть компьютер, у которого скорость V_1 мин. произ. средн. опред. $\leq V_i$ мин. произ. средн. опред., где $1 < i \leq n$, то есть скорости V_i средн. опред. меньше, чем у него, среди n компьютеров нет.

Дополнительно отметим, что K_2 мин. произ. — второй компьютер в ряду, то есть, если из множества n компьютеров исключить K_1 мин. произ., то скорости V_i средн. опред. меньше V_2 мин. произ. средн. опред. среди множества $n-1$ компьютеров не будет.

Рассчитаем время распознавания файла $F_{\text{классиф.}}$ такой системой. Исследовав этапы работы параллельной вычислительной системы из n компьютеров, можно сделать вывод, что время распознавания будет зависеть от работы самого долгого по времени звена. Обозначим компьютер из этого звена как $K_{\text{долг. звена}}$, количество объектов, поступивших на $K_{\text{долг. звена}}$, как p долг. звена, скорость $V_{\text{средн. опред.}}$ компьютера $K_{\text{долг. звена}}$ как $V_{\text{долг. звена средн. опред.}}$. Стоит заметить, что компьютер $K_{\text{долг. звена}}$ может не совпадать с K_1 мин. произ.

Для расчетов времени распознавания также необходимо отметить следующее: 1) как говорилось выше, K_i -й компьютер получает объект для распознавания только после распознавания x_k -го объекта и записи сервером его класса в выходной файл; 2) из-за того, что чтение объектов из файла $F_{\text{классиф.}}$ происходит последовательно, а распознавание параллельно, может возникнуть ситуация ожидания чтения K_i -м компьютером своего объекта для распознавания, то есть K_i -й компьютер вынужден ждать, пока остальные компьютеры не прочитают свои объекты; 3) запись в выходной файл тоже происходит последовательно, что предполагает наличие возможности ожидания K_i -м компьютером своей очереди записи. Другими словами, необходима синхронизация процессов распознавания.

Заметим, что из-за синхронизации компьютер $K_{\text{долг. звена}}$ может оказаться последним в очереди на чтение и ему придется ждать, пока все $n-1$ компьютеры не прочитают свои объекты из файла $F_{\text{классиф.}}$. То же самое относится и к записи класса в

Рис. 3. Реализация параллельного вычисления с помощью $n+1$ компьютеров

Fig. 3. Implementing parallel computing with $n+1$ computer

выходной файл. С учетом этих условий рассчитаем время распознавания файла $F_{\text{классиф.}}$:

$$T_{\text{расп.}} = \left(\sum_{k=1}^n T_{0 \text{ предобр.}}(x_k) + \frac{1}{V_{\text{долг. звена средн. опред.}}} + \sum_{k=1}^n T_{0 \text{ добав.}}(c_k) \right) \times P_{\text{долг. звена}}, \quad (6)$$

где $\sum_{k=1}^n T_{0 \text{ предобр.}}(x_k)$ – сумма времени ожидания, пока все $n-1$ компьютеры не прочитают свои объекты, и времени чтения компьютером $K_{\text{долг. звена}}$ своего объекта; $\frac{1}{V_{\text{долг. звена средн. опред.}}}$ – время определения

класса одного классифицируемого объекта компьютером $K_{\text{долг. звена}}$; $\sum_{k=1}^n T_{0 \text{ добав.}}(c_k)$ – сумма времени ожидания, пока все $n-1$ компьютеры не запишут свои классы, и времени записи компьютером $K_{\text{долг. звена}}$ класса своего объекта; $P_{\text{долг. звена}}$ – количество объектов, которое распознает компьютер $K_{\text{долг. звена}}$.

Или, сгруппировав время по функциям, перепишем формулу (6):

$$T_{\text{расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{P_{\text{долг. звена}}}{V_{\text{долг. звена средн. опред.}}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (7)$$

Необходимо принять во внимание, что от реализации алгоритма распределения объектов функцией $P(m, n)$ будет зависеть и результат обработки, так как от функции $P(m, n)$ зависит число $P_{\text{долг. звена}}$. Рассмотрим несколько вариантов реализации этой функции.

Первый вариант. Функция $P(m, n)$ равномерно распределяет классифицируемые объекты между n компьютерами по следующему принципу: $X_i = \langle x_i; x_{i+n}; x_{i+2n}; \dots \rangle$.

Рассчитаем количество классифицируемых объектов, которые получит каждый компьютер. Для этого учтем тот факт, что m (количество объектов) может быть не кратным числу n (количество компьютеров). Тогда некоторые компьютеры получат меньше объектов, а другие больше объектов ($P_{\text{макс.}}$). Рассчитаем максимальное количество объектов, которые получают компьютеры при условии, что $m \geq n$:

$$P_{\text{макс.}} = \left\lceil \frac{m}{n} \right\rceil, \quad (8)$$

где $\left\lceil \frac{m}{n} \right\rceil$ – округление числа m/n до целого числа в большую сторону.

Чтобы воспользоваться формулой (7), необходимо определить $K_{\text{долг. звена}}$, а точнее, его скорость $V_{\text{долг. звена средн. опред.}}$ и количество объектов, которое он распознает. Из (8) и определения K_1 мин. произ.

можно сделать вывод, что $K_{\text{долг. звена}} = K_1$ мин. произ., $V_{\text{долг. звена средн. опред.}} = V_1$ мин. произ. средн. опред., а $P_{\text{долг. звена}} = P_{\text{макс.}}$. Тогда по формуле (7) рассчитаем время распознавания файла $F_{\text{классиф.}}$ при равномерном распределении объектов между n компьютерами:

$$T_{\text{равн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{P_{\text{макс.}}}{V_1 \text{ мин. произ. средн. опред.}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (9)$$

Проанализировав формулу (9), можно сделать вывод, что скорость V_1 мин. произ. средн. опред. обратно пропорциональна времени распознавания.

Выразим V_1 мин. произ. средн. опред. через V_0 средн. опред.: $V_1 \text{ мин. произ. средн. опред.} = \alpha * V_0 \text{ средн. опред.}$. Параметр α показывает, во сколько раз K_1 мин. произ. медленнее K_0 , если $0 < \alpha < 1$, и быстрее, если $\alpha > 1$.

Перепишем формулу (9) с учетом введенного параметра α :

$$T_{\text{равн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{P_{\text{макс.}}}{\alpha \times V_0 \text{ средн. опред.}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (10)$$

Рассмотрев формулу (10), можно заметить, что первое и третье слагаемые всегда будут вносить одинаковый вклад в общее время распознавания несмотря на изменение параметра α . Другими словами, как бы ни изменялся параметр α , сумма $\sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k)$ всегда будет нижней границей времени распознавания.

Исследуем влияние параметра α на время распознавания:

1) $\alpha = 1$, тогда $V_1 \text{ мин. произ. средн. опред.} = V_0 \text{ средн. опред.}$, а общее время распознавания, согласно формуле (10), является суммой времени распознавания $P_{\text{макс.}}$ объектов, времени предобработки m объектов и времени добавления результатов распознавания m объектов в выходной файл:

$$T_{\text{равн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{P_{\text{макс.}}}{V_0 \text{ средн. опред.}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k);$$

2) $\alpha = 2$, тогда $V_1 \text{ мин. произ. средн. опред.} = 2 * V_0 \text{ средн. опред.}$, а общее время распознавания рассчитывается следующим образом:

$$T_{\text{равн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{P_{\text{макс.}}}{2 \times V_0 \text{ средн. опред.}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k);$$

3) раз V_1 мин. произ. средн. опред. обратно пропорционально общему времени распознавания, определим α_{limit} , при котором $T_{0 \text{ расп.}} = T_{\text{равн. расп.}}$, то есть не будет выигрыша по времени при распознавании; приравняв формулы (5) и (10), определим $\alpha_{\text{limit}} = P_{\text{макс.}}/m$.

Рассмотрев формулу (10) и сравнив между собой время распознавания при различных значениях параметра α , можно сделать следующие выводы:

- если параметр $\alpha < p_{\text{макс.}}/m$, то выигрыша по времени при распознавании не будет;
- если параметр $\alpha = p_{\text{макс.}}/m$, то $T_0 \text{ расп.} = T_{\text{равн. расп.}}$;
- если $\alpha > p_{\text{макс.}}/m$, то можно будет наблюдать уменьшение времени распознавания.

Дополнительно отметим, что с ростом α время распознавания тоже будет уменьшаться, но только до своей нижней границы:

$$\sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k).$$

Рассмотрим достоинства и недостатки данной реализации функции $P(m, n)$. Среди достоинств можно выделить простоту реализации функции $P(m, n)$, отсутствие трудоемких вычислений и то, что заранее известно, какие объекты на какие компьютеры попадут.

Главным недостатком данной функции $P(m, n)$ является то, что она не учитывает особенности скорости обработки компьютерами объектов. Другими словами, у компьютеров может быть разная производительность, а время работы вычислительной системы n компьютеров определяется временем работы самым медленным K_1 мин. произ. компьютером. Отсюда вытекает возможность простоя вычислительных ресурсов других компьютеров.

Одним из способов решения проблемы простоя компьютеров является возврат системе компьютера, который выполнил свою работу, чтобы он мог выполнять работу уже другой задачи. Определить объем работы (количество объектов для распознавания) компьютера возможно благодаря тому, что заранее известно распределение объектов по компьютерам.

Но этот способ не поможет решить проблему, если задачу нужно решить в минимальные сроки, то есть критерием оптимальности выступит время распознавания. Несмотря на наличие свободных ресурсов, система не сможет ими воспользоваться для сокращения времени распознавания объектов.

Второй вариант. В предыдущем примере существует проблема простая компьютеров. Для ее решения можно предложить реализацию функции $P(m, n)$ на основе жадного алгоритма, суть которого заключается в том, чтобы отправлять объект для распознавания первому освободившемуся компьютеру, тем самым уменьшая время простоя.

Введем следующие параметры: число b – число классифицируемых объектов, которое пришло на вход K_1 мин. произ. компьютера; число $z = p_{\text{макс.}} - b$ – число перераспределенных объектов, то есть, если на вход K_1 мин. произ. компьютера поступит число b и $b < p_{\text{макс.}}$, то жадный алгоритм перераспределит z объектов с K_1 мин. произ. компьютера на другие компьютеры, тем самым уменьшая время их простоя.

В результате работы этого алгоритма могут сложиться две различные ситуации.

1. Жадному алгоритму не удалось переопределить число объектов, пришедших на вход K_1 мин. произ. компьютера, то есть $b = p_{\max}$. Тогда $T_{\text{жадн. расп.}} = T_{\text{равн. расп.}}$ и нет никакого выигрыша во времени. Такая ситуация возможна, когда скорости V_i средн. опред. всех n компьютеров равны между собой. В таком случае распределение классифицируемых объектов между n компьютерами жадного алгоритма будет совпадать с распределением равномерного алгоритма.

2. Жадному алгоритму удалось переопределить число объектов, пришедших на вход K_1 мин. произ. компьютера, то есть $b < p_{\text{макс}}$. Тогда $T_{\text{жзн. расп.}} < T_{\text{равни. расп.}}$. Для доказательства обратимся к рисунку 4.

Рис. 4. Диаграмма производительности компьютеров
в работе жадного алгоритма

Fig. 4. Computer performance graph in the work of the greedy algorithm

Рассмотрим точку времени T_b – время, когда завершит свою работу $K_{1 \text{ мин. произ.}}$ компьютер по распознаванию b объектов. Стоит заметить, что в это время уже не будет объектов для распознавания. Иначе жадный алгоритм должен будет назначить $K_{1 \text{ мин. произ.}}$ компьютеру объект для распознавания, что противоречит тому, что на вход $K_{1 \text{ мин. произ.}}$ компьютера было подано только b объектов.

В точке времени T_b возможны два варианта работы жадного алгоритма.

1. Все остальные ($n-1$) компьютеры уже выполнили свою работу, то есть простаивают. Тогда время распознавания равно времени обработки K_1 мин. произ. компьютером b объектов:

$$T_{\text{ж.д.н. рас.}} = \sum_{k=1}^m T_0 \text{предобр.}(x_k) + \frac{b}{V_{1 \text{ мин. произв. средн. опед.}}} + \sum_{k=1}^m T_0 \text{добав.}(c_k). \quad (11)$$

2. Часть компьютеров из $n-1$ еще обрабатывают свои последние объекты. Определим компьютер с минимальной производительностью среди $n-1$ компьютеров, то есть компьютер K_2 мин. произ. Для простоты сдвинем время начала обработки последнего объекта K_2 мин. произ. компьютера с точки T_2 мин. произ. до точки T_b , то есть $T_b = T_2$ мин. произ. $+ \Delta T$.

Тогда время распознавания равно сумме времени обработки K_1 мин. произ. компьютером b объектов и времени обработки K_2 мин. произ. своего последнего объекта:

$$T_{\text{жадн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{b}{V_{1 \text{ мин. произв. средн. опред.}}} + \frac{1}{V_{2 \text{ мин. произв. средн. опред.}}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (12)$$

Очевидно, что время распознавания по формуле (12) больше времени распознавания по формуле (11). Поэтому дальше будем ориентироваться на время, рассчитанное по формуле (12).

Докажем теперь, что время распознавания по формуле (12) меньше времени распознавания по формуле (9). Очевидно, что можно опустить слагаемые $\sum_{k=1}^m T_{0 \text{ предобр.}}(x_k)$ и $\sum_{k=1}^m T_{0 \text{ добав.}}(c_k)$, так как они вносят одинаковый вклад в обе формулы. Для доказательства рассмотрим граничный случай, когда количество объектов в обеих формулах одинаково, то есть жадный алгоритм смог переопределить только один объект с компьютера K_1 мин. произ. Тогда формула (12) будет выглядеть следующим образом:

$$T_{\text{жадн. расп.}} = \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \frac{p_{\max} - 1}{V_{1 \text{ мин. произв. средн. опред.}}} + \frac{1}{V_{2 \text{ мин. произв. средн. опред.}}} + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k). \quad (13)$$

Проведя анализ формул (9) и (13), можно сделать вывод: для доказательства того, что $T_{\text{жадн. расп.}} < T_{\text{равн. расп.}}$, стоит сравнить между собой $1/V_{1 \text{ мин. произв. средн. опред.}}$ и $1/V_{2 \text{ мин. произв. средн. опред.}}$. Как отмечалось выше, $V_1 \text{ мин. произв. средн. опред.} \geq$

$\geq V_{2 \text{ мин. произв. средн. опред.}}$ и $T_b = T_{2 \text{ мин. произ.}} + \Delta T$. Это позволяет сделать вывод, что $1/V_{1 \text{ мин. произв. средн. опред.}} > 1/V_{2 \text{ мин. произв. средн. опред.}}$. Знак «строго» берется потому, что даже при равенстве скоростей между собой необходимо учитывать тот факт, что $T_b = T_{2 \text{ мин. произ.}} + \Delta T$, то есть K_2 мин. произ. компьютер начнет работу раньше, чем закончит работу K_1 мин. произ. компьютер.

В других же случаях с уменьшением параметра b будет уменьшаться $b/V_{1 \text{ мин. произв. средн. опред.}}$, что приводит к уменьшению времени распознавания.

Рассмотрим достоинства и недостатки реализации функции $P(m, n)$ в виде жадного алгоритма. Среди достоинств можно выделить простоту реализации функции $P(m, n)$, отсутствие трудоемких вычислений и возможность уменьшения времени простоя компьютеров, что отразится на времени распознавании.

Недостатки данной функции $P(m, n)$: неизвестно, какие объекты на какие компьютеры попадут; распределение по жадному алгоритму не всегда будет оптимальным, если критерием оптимальности выступит общее время распознавания.

Характеристики сети. Важнейшей характеристикой системы является время коммутации и передачи данных по каналам связи в сети, которое может оказать значительное влияние на длительность решения задачи. Фактически интерес представляет время, через которое после начала процесса вычисления пользователь получит его результат. Время получения результата на i -м компьютере ($T_i \text{ получ. резул.}$) складывается из времени подключения ($T_i \text{ подкл.}$) и времени на обработку данных ($T_i \text{ обраб. дан.}$): $T_i \text{ получ. резул.} = T_i \text{ подкл.} + T_i \text{ обраб. дан.}$

Рассчитаем время, которое нужно потратить на этап обучения (рис. 5).

Суть обучения n компьютеров заключается в следующем: пользователь передает на сервер файл с обучающими объектами; сервер отправляет

Рис. 5. Схема обучения n компьютеров

Fig. 5. Training scheme of n computers

далее этот файл параллельно n компьютерам; каждый компьютер обучается на этом файле. Время обучения n компьютеров рассчитывается следующим образом:

$$T_{\text{обуч.}} = T_{\text{загр.}}(F_{\text{обуч.}}) + T_{\text{макс. перед.}} + T_{\text{макс. обуч.}}, \quad (14)$$

где $T_{\text{загр.}}(F_{\text{обуч.}})$ – время загрузки файла $F_{\text{обуч.}}$ на сервер; $T_{\text{макс. перед.}}$ – максимальное время передачи файла $F_{\text{обуч.}}$ на один из n компьютеров; $T_{\text{макс. обуч.}}$ – максимальное время обучения одного из n компьютеров на основе файла $F_{\text{обуч.}}$.

Рассчитаем время, необходимое на распознавание (рис. 6).

Распознавание происходит следующим образом: 1) пользователь передает на сервер файл с классифицируемыми объектами; 2) сервер осуществляет предобработку и распределение классифицируемых объектов из этого файла параллельно n компьютерам согласно реализации функции $P(m, n)$; 3) i -й компьютер, получив классифицируемый объект, начинает процесс определения его класса; 4) класс распознанного объекта записывается в выходной файл i -го компьютера, после чего отправляется на сервер; 5) сервер после получения

класса объекта начинает добавление его в общий выходной файл; 6) после добавления всех классов в общий выходной файл сервер отправляет общий выходной файл клиенту.

Как уже отмечалось, общее время работы системы зависит от времени работы звена с максимальным временем обработки своих объектов, то есть компьютером $K_{\text{долг. звена}}$. Для равномерного распределения компьютер $K_{\text{долг. звена}}$ определяется параметром V_1 мин. произ. средн. опред., но при этом не учитывались характеристики канала связи. Поэтому теперь для K_i -го компьютера время расчета определения класса одного объекта складывается из следующих параметров: $T_{i \text{ сетев. опред.}} = T_{i \text{ перед.}}(x_k) + T_{i \text{ сред. опред.}}(x_k) + T_{i \text{ перед.}}(c_k)$, где $T_{i \text{ перед.}}(x_k)$ – время передачи x_k -го объекта с сервера на i -й компьютер; $T_{i \text{ сред. опред.}}(x_k)$ – время определения класса x_k -го объекта на i -м компьютере; $T_{i \text{ перед.}}(c_k)$ – время передачи класса x_k -го объекта с i -го компьютера на сервер.

Другими словами, компьютер $K_{\text{долг. звена}}$ будет определяться не только производительностью компьютера, но и характеристиками канала связи. Те-

Рис. 6. Схема распознавания n компьютеров

Fig. 6. The recognition scheme of n computers

перь с учетом вышесказанных замечаний и формулы (7) рассчитаем общее время распознавания:

$$T_{\text{расп.}} = T_{\text{загр.}}(F_{\text{классиф.}}) + \sum_{k=1}^m T_{0 \text{ предобр.}}(x_k) + \\ + \sum_{k=1}^{P_{\text{долг. звена}}} T_{\text{долг. звена перед.}}(x_k) + \frac{P_{\text{долг. звена}}}{V_{\text{долг. звена средн. опред.}}} + \\ + \sum_{k=1}^{P_{\text{долг. звена}}} T_{\text{долг. звена перед.}}(c_k) + \sum_{k=1}^m T_{0 \text{ добав.}}(c_k) + T_{\text{отпр.}}, \quad (15)$$

где $T_{\text{загр.}}(F_{\text{классиф.}})$ – время загрузки файла $F_{\text{классиф.}}$ на сервер; $\sum_{k=1}^{P_{\text{долг. звена}}} T_{\text{долг. звена перед.}}(x_k)$ – время передачи

$P_{\text{долг. звена}}$ объектов на компьютер $K_{\text{долг. звена}}$;

$\sum_{k=1}^{P_{\text{долг. звена}}} T_{\text{долг. звена перед.}}(c_k)$ – время передачи классов

$P_{\text{долг. звена}}$ объектов на сервер; $T_{\text{отпр.}}$ – время отправки общего выходного файла клиенту.

Если время решения задачи $T_{0 \text{ реш. зад.}} = T_{0 \text{ обуч.}} + T_{0 \text{ расп.}}$ на K_0 компьютере меньше времени $T_{\text{паралл. реш. зад.}} = T_{\text{паралл. обуч.}} + T_{\text{паралл. расп.}}$ (с учетом времени коммутации и передачи данных по сети), то распараллеливание не повысит общую производительность системы. Здесь $T_{0 \text{ обуч.}}$ рассчитывается по формуле (1), $T_{0 \text{ расп.}}$ – по формуле (3), $T_{\text{паралл. обуч.}}$ – по формуле (14), $T_{\text{паралл. расп.}}$ – по формуле (15). Такая ситуация может возникнуть при значительном влиянии характеристик канала связи, в том числе времени коммутации и передачи данных.

Минимизировать $T_{i \text{ обраб. дан.}}$ можно за счет установки высокопроизводительного оборудования. Минимизация $T_{\text{расп.}}$ достигается путем установки дополнительных компьютеров, тем самым снижая среднее значение $T_{i \text{ обраб. дан.}}$, а также стремлением к максимально низкому значению $T_{i \text{ подкл.}}$, размещая сервер и компьютеры в одной локальной сети. Следует помнить, что при $T_{i \text{ обраб. дан.}}$, близком к $T_{i \text{ подкл.}}$, требуется решить задачу выбора оптимального количества компьютеров, дабы избежать отрицательного эффекта от применения системы. Оптимизировать параметры $T_{i \text{ подкл.}}$ и $T_{i \text{ обраб. дан.}}$ стоит исходя из соображений целесообразности и общей эффективности.

В ходе исследования было разработано web-ориентированное инструментальное средство по распознаванию образов в идее расширяемой библиотеки методов на основе прецедентов, способное не только работать на одном сервере, но и распараллеливать обработку данных между компьютерами.

Реализация такой параллельности была представлена в статье [12], где описаны теоретические и экспериментальные разработки двух схем распараллеливания на основе grid-технологий: с распараллеливанием алгоритма обучения и с распараллеливанием алгоритма распознавания, а также представлены элементы практических наработок различных этапов параллельного распознавания на

n компьютерах на основе метода k ближайших соседей.

Перспективность исследования определяется практической эффективностью разработанного web-приложения. Практика показывает, что распараллеливание позволяет эффективно применять вычислительные ресурсы для решения задач в минимальные сроки, а применение web-технологии – повысить объем обрабатываемых данных и скорость доступа к ним.

Теоретические выкладки и расчеты работы являются базовыми для практических разработок данного web-приложения. В статье обозначены ограничения на применение вычислительных ресурсов, несоблюдение которых приводит к ухудшению результатов времени распознавания. Также рассчитана минимальная граница времени распознавания вне зависимости от количества используемых компьютеров.

Реализуемая архитектура инструментария позволяет решать задачу повышения эффективности вычислительных ресурсов через процедуры настройки структуры подключаемых методов и алгоритмов обработки данных и реконfigurирования подключаемых технических вычислительных средств Internet-сети.

Литература

1. Козлов А.Н. Интеллектуальные информационные системы. Пермь: Изд-во Перм. ГСХА, 2013. 278 с.
2. Громов Ю.Ю., Иванова О.Г., Алексеев В.В. [и др.]. Интеллектуальные информационные системы и технологии. Тамбов: Изд-во ТГТУ, 2013. 244 с.
3. Han J., Kamber M., Pei J. Data mining: concepts and techniques, Elsevier, 2012, 744 p.
4. Фомин Я.А. Распознавание образов: теория и применение. М.: Фазис, 2014. 460 с.
5. Мерков А.Б. Распознавание образов: введение в методы статистического обучения. М.: Едиториал УРСС, 2011. 254 с.
6. Sandoval J., Hernández G. Learning of natural trading strategies on foreign exchange high-frequency market data using dynamic bayesian networks. Machine Learning and Data Mining in Pattern Recognition. Proc. 10th Intern. Conf., 2014, pp. 408–421.
7. Сикудер Д.В., Фомин В.В. Проектные решения web-ресурса интеллектуального анализа данных и поддержки принятия решений на основе унифицированного класса методов решающих функций. СПб: Изд-во РГПУ, 2016. 102 с.
8. Марманис Х., Бабенко Д. Алгоритмы интеллектуального Интернета. Передовые методики сбора, анализа и обработки данных; [пер. с англ.]. СПб: Символ-Плюс, 2011. 466 с.
9. Варшавский П.Р., Еремеев А.П. Моделирование рассуждений на основе прецедентов в интеллектуальных системах поддержки принятия решений // Искусственный интеллект и принятие решений. 2009. № 1. С. 45–57.
10. Петров Е.В., Фомин В.В. Organization of parallel processing for implementation of web data mining system // Ученые записки РГТУ. 2014. № 33. С. 149–154.
11. Алпатов А.Н. Развитие распределенных технологий и систем // Перспективы науки и образования. 2015. № 2. С. 60–66.
12. Фомин В.В., Александров И.В. Об одном опыте применения параллельных вычислений при разработке web-инструментария распознавания образов // Программные продукты, системы и алгоритмы. 2017. № 1. URL: <http://swsys-web.ru/experience-in-parallel-computing-web-tool-for-pattern-recognition.html> (дата обращения: 02.04.2017).

PARALLEL COMPUTING WHEN IMPLEMENTING WEB-BASED PATTERN RECOGNITION TOOLS BASED ON USE CASE METHODS

V.V. Fomin¹, Dr.Sc. (Engineering), Professor, v_v_fomin@mail.ru

I.V. Aleksandrov, Postgraduate Student, chrono555@yandex.ru

¹ Herzen State Pedagogical University of Russia, Reki Moyki Quay 48, St. Petersburg, 191186, Russian Federation

Abstract. The article considers a software solution aimed at improving image recognition quality and effectiveness of machine learning tools using grid technologies. The paper formulates strategic directions for pattern recognition tools as a software system based on the principles of distributed systems, parallelism and adaptive configuration of computing resources. The article considers the structure of web tools for pattern recognition using the concept of the algorithm library. It also describes the algorithmic solutions for parallelizing learning and recognition algorithms on the basis of classical data mining algorithms that are proved themselves in practice. Such algorithms include precedent (case-based reasoning) methods or methods based on proximity metrics. They have a great potential to parallelize computational processes and develop parallel algorithms for their implementation.

The search for ways to improve the productivity of computers, especially when implementing web tools based on resource-intensive computer algorithms for machine recognition and prediction, led to the decision to create a grid system. The architecture and implementation of the grid system considered in the article assumes parallelization and organization of distributed computations on a multi-machine basis using Internet technologies. It allows obtaining similar computing power as multiprocessor computer systems but the cost is much lower.

The article considers the problem of increasing the efficiency of computing resources with the possibility of reconfiguring the structure of Internet connections, including the procedure of configuring a computer network structure, connectable communication channels and dedicated servers depending on the original algorithms and data. The paper presents dependencies of operation execution time parameters on the service discipline that adapts the system to user requests. Here the tasks are ranked according to resource intensity and computing power corresponding to their rank is allocated to them.

Keywords: intelligent information systems, pattern recognition, machine learning, web systems, parallel computing.

References

1. Kozlov A.N. *Intellektualnye informatsionnye sistemy* [Intelligent Information Systems]. Perm, Permskaya GSKhA Publ., 2013, 278 p.
2. Gromov Yu.Yu., Ivanova O.G., Alekseev V.V. *Intellektualnye informatsionnye sistemy i tekhnologii* [Intelligent Information Systems and Technologies]. Tambov, TGTU Publ., 2013, 244 p.
3. Han J., Kamber M., Pei J. *Data mining: concepts and techniques*. Elsevier Publ., 2012, 744 p.
4. Fomin Ya.A. *Raspoznavanie obrazov: teoriya i primeneniya* [Image Identification: Theory and Applications]. Moscow, Fazis Publ., 2014, 460 p.
5. Merkov A.B. *Raspoznavanie obrazov: vvedenie v metody statisticheskogo obucheniya* [Image Identification: Introduction and Statistical Learning Methods]. Moscow, Editorial URSS Publ., 2011, 254 p.
6. Sandoval J., Hernández G. Learning of Natural Trading Strategies on Foreign Exchange High-Frequency Market Data Using Dynamic Bayesian Networks. *Machine Learning and Data Mining in Pattern Recognition: Proc. 10th Int. Conf.* St. Petersburg, Russia, Springer Publ., 2014, pp. 408–421.
7. Sikuler D.V., Fomin V.V. *Proektnye resheniya web-resursa intellektualnogo analiza dannykh i podderzhki prinyatiya resheniy na osnove unifikirovannogo klassa metodov reshayushchikh funktsiy* [Project Decisions of a Web Source of Data Mining and Decision Support Based on a Decision Function Method Unified Class]. St. Petersburg, RGPU Publ., 2016, 102 p.
8. Marmanis Kh., Babenko D. *Algorithms of the Intelligent Web*. Manning Publ., 2009, 368 p. (Russ.ed.: St. Petersburg, Simvol-Plyus Publ., 2011, 466 p.).
9. Varshavsky P.R., Ereemeev A.P. Modeling Case-Based Reasoning in Intelligent Decision Support Systems. *Iskusstvenny intellekt i prinyatie resheniy* [Artificial Intelligence and Decision Making]. 2009, no. 1, pp. 45–57 (in Russ.).
10. Petrov E.V., Fomin V.V. Organization of parallel processing for implementation of web data mining system. *Uchenye zapiski Rossiyskogo gos. gidrometeorologicheskogo universiteta: nauchno-teoretichesky zhurnal* [RSHU Proc. Jour.]. 2014, no. 33, pp. 149–154 (in Russ.).
11. Alpatov A.N. Problems of distributed systems. *Perspektivy nauki i obrazovaniya* [Perspectives of Science and Education]. 2015, no. 2 (14), pp. 60–66 (in Russ.).
12. Fomin V.V., Aleksandrov I.V. On one operational experience of parallel computing when developing image identification web tools. *Programmnye produkty, sistemy i algoritmy* [Software Products, Systems and Algorithms]. 2017, no. 1. (in Russ.). Available at: <http://swsys-web.ru/experience-in-parallel-computing-web-tool-for-pattern-recognition.html> (accessed April 2, 2017).