Компьютерная Математика

УДК 004.3

Я.О. ТУПАЛО

МЕТОДЫ ОБРАБОТКИ БОЛЬШИХ ДАННЫХ

Приведены причины появления больших данных, методы их обработки, а также практический пример применения методов обработки больших данных с использованием распределенной вычислительной системы.

Введение. Согласно данным компании Cisco, к концу 2016 года по всему миру было передано 1,1 зеттабайта данных. А в 2019 году объем трафика удвоится, достигнув 2 зеттабайт в год [1]. Здесь рассматриваются основные концепции анализа больших данных, а также приведен пример их обработка с помощью методов machine learning. Большими данными обычно называют получение, обработку и анализ данных, отличающихся от обычных структурированных данных в реляционных базах; к ним относят неструктурированные файлы, цифровое видео, изображения, данные датчиков, log-файлы и вообще любые данные, не содержащие в записях специальных поисковых полей. Неструктурированные данные интересны, но из них трудно синтезировать бизнес-аналитику или сделать выводы, если они не коррелируют со структурированными данными. Появляются и новые источники больших данных, такие как машинное генерирование (например, log-файлы или данные сенсорных сетей), мобильные устройства (видео, фотографии и текстовые сообщения) и системы машина-машина, когда IoT («Интернет вещей») сообщает о состоянии для планирования обслуживания парка автомобилей или самолетов либо для телеметрического мониторинга. Большие данные – это прежде всего объем. По оценкам IBM, в настоящее время создается 2.5 квинтиллиона (2 500 000 000 000 000 000) байт данных ежедневно [2]. Во-вторых, это скорость передачи данных, растущая с ростом пропускной способности сети; в настоящее время это, как правило, гигабитные, (а уже не мегабитные) скорости

(GigE, 10G, 40G, 100G). В-третьих, это разнообразие типов неструктурированных данных, таких как цифровые видеопотоки, данные сенсорных сетей и log-файлы. В итоге, это достоверность данных – насколько при принятии решений можно доверять данным, собранным в таких больших объемах на таких высоких скоростях. Трудно быть уверенным, что данные не подделаны, не повреждены и поступают из ожидаемого источника, – если это, например, данные от тысяч видеокамер, ежечасно снимающих многие тысячи видеокадров. Кратко охарактеризуем некоторые ключевые аспекты больших данных, приложений и систем. Возможно, лучший способ понять большие данные проанализировав их историю, как в журнале Forbes [3]. Прошло много времени с момента выхода статьи, за этот период рост больших данных ускорился и на сегодня составляет более 2.5 эксабайтов в день. Интересно, что большинство этих данных технически невозможно проанализировать силами человека (для этого каждый из 7 миллиардов жителей планеты должен был бы просматривать более 300 МБ информации ежедневно). С учетом этой проблемы единственный логичный способ использовать такой объем данных – это системы машина-машина или интеллектуальные запросы к большим данным. Кроме того, если данные хранятся длительное время, как узнать, не были ли они повреждены? Можно, конечно, хранить дайджесты данных (например, MD5 – одну из форм контрольной суммы) и использовать избыточный массив независимых дисков (RAID-зеркалирование, XOR-четность или стирающие коды для обнаружения и восстановления поврежденных данных), но существует возможность не отслеживаемого повреждения данных [4]. Организация Internet Archive (Архив Интернета), выполняющая долгосрочное хранение данных, исследовала эту проблему. В целом достоверность больших данных – это проблема, но стирающие коды и методы создания расширенных дайджестов данных обещают ее решение. Традиционные методы, такие как XOR-четность или простое зеркалирование (которые обеспечивают только защиту от потери данных при сбоях устройств хранения и не работают в сценариях повреждений, вызванных ошибками в программном обеспечении, ошибками оператора датацентра или выходом носителя из строя с течением времени), в настоящее время заменяются массивами RAID-6 и усовершенствованными кодами стирания. Концепция долговечности больших данных – важная проблема. При таком количестве данных идея проверки их достоверности человеком просто нереализуема, а отсутствующие данные могут остаться необнаруженными, пока к ним не будет выполнен запрос в далеком будущем.

Одним из популярных методов обработки больших данных на сегодняшний день являются методы машинного обучения (Machine Learning). Машинное обучение имеет давнюю историю, уходя корнями в фундаментальную математику и статистику. В наши дни машинное обучение использует аналитические модели и алгоритмы, которые непрерывно совершенствуются, обучаясь на основе данных, благодаря чему компьютеры могут улавливать скрытый смысл, не будучи запрограммированными на то, где его следует искать. Это значит, что ученые и специалисты по анализу данных могут научить компьютеры решать

проблемы, не задавая правила для каждого нового набора данных. Используя алгоритмы, которые обучаются путем изучения сотен и тысяч образцов данных, компьютеры могут решать аналогичную проблему в новой ситуации, составляя прогноз исходя из накопленного опыта. И они делают это с точностью, которая уже сравнима с интеллектом человека. Использование машинного обучения стало массовой тенденцией, способствуя получению конкурентных преимуществ в самых различных отраслях: от розничной торговли до охраны правопорядка, от курортных комплексов до кабельного телевидения, от спортивных команд до исследователей космоса.

Machine Learning и Традиционная Статистика: суть и подходы. «Machine Learning (ML)» и Традиционная Статистика (ТС) различаются как по своей сути, так и в подходах к данным. Из Википедии: Машинное обучение (англ. Machine Learning) – обширный подраздел искусственного интеллекта, математическая дисциплина, использующая разделы математической статистики, численных методов оптимизации, теории вероятностей, дискретного анализа, и извлекающая знания из данных. Когда говориться об «аналитике», то в это понятие включается как data mining/machine learning, так и традиционные статистические методы. Но не стоит забывать о том, что они различаются как в сферах применения, так и в общей философии. ML – подвид Искусственного Интеллекта (ИИ). Основная задача ИИ – понимание и воспроизведение мыслительного процесса человека и его воспроизведение в нечеловеческой (машинной) среде. ML нацелено на автоматическое обнаружение закономерностей в данных с помощью вычислительных алгоритмов и их дальнейшая структуризация в новые, но схожие данные. То есть основная задача – изучение и создание систем, способных делать выводы из данных с помощью обучения на примерах. ML – это другие, чем «data mining» или «предсказательная аналитика», но при этом является неотъемлемой частью их обоих. Основные понятия ML начали формулироваться в 1950-х годах, а в конце 80-х – начале 90-х появлялись различные успешные стартапы по этой теме с самыми различными способами применений, как, например, реально-временное отслеживание случаев мошенничества, распознавание символов и рекомендательные сервисы (первое поколение систем ML). ML также тесно связано с «Теорией Распознавания Образов» (ТРО). Но несмотря на это, они являются двумя сторонами одной и той же отрасли, чья задача – интеллектуальный анализ данных. Исторически, методы и подходы практикующих ML сильно зависели от вычислительной мощности компьютеров, на сегодняшний день эта проблема не так актуальна, сейчас вполне естественно обрабатывать терабайтные массивы данных в реальном времени. С другой стороны, методы ТС разрабатывались в условиях, в которых не было возможности полагаться на компьютеры. Как результат, ТС в основном оперирует малыми выборками и большими допущениями в вопросе данных и их распространения. ML, как правило, старается уменьшить число изначальных допущений и более свободно в использовании различных методов и средств для решения поставленной задачи, часто прибегая к эвристике. Предпочитаемый

метод обучения в ML – индуктивный. Другой спектр инструментов ML – аналитическое (или дедуктивное) обучение, где данных либо мало, либо предпочтительна работа с их малыми наборами. В реальном мире, ML обычно чередует использование обоих методов [5]. Традиционная статистика крайне консервативна в своих подходах к задаче и часто делает слишком большие допущения, особенно в вопросе распределения данных. В табл. 1 приводится разница в подходах и идеологии этих двух направлений.

ТАБЛИЦА 1

Машинное обучение (ML)	Традиционная статистика (ТС)
Задача: «изучение» всех типов данных	Задача: анализ и классификация данных
Никаких обязательных допущений касательно задачи и распределения данных	Большие допущения относительно вопроса и распределения данных
Более свободно в методах и подходах	Консервативна в методах и подходах
Обобщение достигается эмпирически, с помощью подтверждения предположений на практике в дополнении к подготовленным тестам	Обобщение преследуется на основе заранее подготовленных тестов
Использует эвристические подходы для поиска лучшего решения	Большие допущения о данных и решаемом вопросе, поиск оптимального решения ведется в заданных рамках
Предпочтительно использование алгоритмов, способных к обработке большого количества функций	Часто требует независимых функций, предпочтительно использование меньшого количества входных функций
Уменьшение анализируемых данных не нужно («чем больше данных, тем лучше»)	Активно продвигает анализ меньших объемов данных (выборки, уменьшение входов)
Решает более сложные проблемы обучения, восприятия, воспроизведения значений	В основном сосредоточены на традиционных методах анализа

Обучение может быть достигнуто путем написания программы, учитывающей все возможные комбинации данных. Этот процесс будет крайне ресурсоемким и невозможным в выполнении применительно к реальным рабочим ситуациям. Такая программа никогда не будет хорошей, как тщательно написанный и выверенный алгоритм обучения. Подобные алгоритмы автоматически учатся

на примерах (как и люди), а затем обобщают данные на основе того, что они узнали (индуктивное обучение). Способность к обобщению и категоризации – ключевой показатель оценки производительности обучающейся платформы. Самые популярные алгоритмы обучения можно разделить на контролируемые и неконтролируемые, а затем каждую категорию разделить по возможностям и функциям [5].

Контролируемое обучение включает:

– классификация: определение к какой категории принадлежит то или иное понятие (чаще всего используется двоичная классификация), например, платежеспособный ли покупатель;

– регрессия: прогнозирование конкретного показателя на протяжении выбранного промежутка времени, например, сколько средств со своей кредитной карты покупатель потратит в следующем месяце при учете всей прочей доступной информации;

– прогнозирование: оценка макропеременных (суммарных) как, например, месячные продажи конкретного продукта;

– определение важности показателя: распознает наиболее значимые в расчете прочей классификации или регрессии показатели.

Неконтролируемое обучение включает:

– кластеризация: поиск уже существующих групп данных.

Ассоциативные модели: «анализ потребительской корзины» (например, комбинации продуктов, которые часто приобретают вместе).

Использование методов Машинное обучение на практике. В данном разделе автор приводит пример как можно обрабатывать большие данные с помощью Машиного обучения. Данная система разрабатывалась с использованием математической библиотеки FAAN и тестировалась в распределенной вычислительной системе Института кибернетики имени В.М. Глушкова НАН Украины [6]. Для достижения параллельной обработки данных был использован MPI.

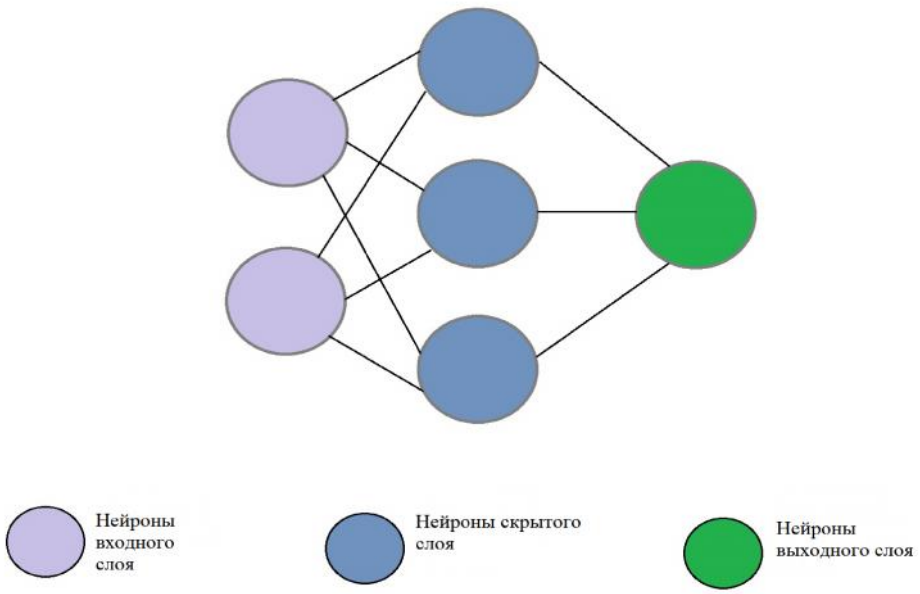
Описание сети: задачу, которую будет решать система – исключение ИЛИ из области булевой алгебры. В реальном мире система Машиного обучения оперирует огромными данными, данный пример логического исключения ИЛИ к выключателю приведен автором лишь для наглядности, но данный пример можно без труда масштабировать для решения задач связанных с большими данными. Как правило, выключатель имеет 2 режима: ВКЛ (1) и ВЫКЛ (– 1). Поэтому теоретически возможны четыре типа ситуаций: ВКЛ и ВЫКЛ.

$$
\begin{aligned}
1 \quad &\text{и} \quad -1 \quad &&= 1 \\
-1 \quad &\text{и} \quad 1 \quad &&= 1 \\
1 \quad &\text{и} \quad 1 \quad &&= -1 \\
-1 \quad &\text{и} \quad -1 \quad &&= -1
\end{aligned}
$$

В первом случае выключатель включен, а результат операции равен 1. Во втором случае выключатель выключен, а результат операции так же равен 1.

А теперь, обратите внимание, что функции ВКЛ и ВЫКЛ включены одновременно, что в реальности невозможно, то есть результат операции имеет код равный 1. И в четвертом случае функции ВКЛ и ВЫКЛ выключаются одновременно, такое состояние невозможно, поэтому 1.

Как правило, для решения такой задачи нет необходимости создавать нейронную сеть. Автор делает это исключительно в демонстративных целях. Наша нейронная сетка как показано на рисунке, будет иметь 3 уровня. Входной уровень состоит из 2 нейронов. На каждом из них будем подавать номер (1 или – 1). Скрытый уровень состоит из трех нейронов. Последний уровень-выход. В данном уровне всего 1 нейрон. Он будет сообщать о возможности той или иной комбинации (1 или – 1). Нейроны входного слоя, нейроны скрытого слоя, нейроны выходящего слоя.



РИСУНОК

Приступая к работе: FANN обычно работает в двух различных режимах, режим обучения и режим выполнения. Хотя можно совместить оба режима в одной программе, рекомендуется использовать различные программы. Есть несколько причин, почему появилась идея написать отдельно две программы, которые отвечают за подготовку и выполнение отдельно, но наиболее очевидным является тот факт, что типичная система FANN готовится только один раз, при этом выполняется многократно.

Обучение: далее приводится простая программа, которая готовит FANN с набором данных, а затем сохраняет их в файл.

```
#include "fann.h"
int main()
{
const unsigned int num_input = 2;
const unsigned int num_output = 1;
const unsigned int num_layers = 3;
const unsigned int num_neurons_hidden = 3;
const float desired_error = (const float) 0.001;
const unsigned int max_epochs = 500000;
const unsigned int epochs_between_reports = 1000;
struct fann *ann = fann_create_standard(num_layers, num_input,
num_neurons_hidden, num_output);
fann_set_activation_function_hidden(ann, FANN_SIGMOID_SYMMETRIC);
fann_set_activation_function_output(ann, FANN_SIGMOID_SYMMETRIC);
fann_train_on_file(ann, "xor.data", max_epochs,
epochs_between_reports, desired_error);
fann_save(ann, "xor_float.net");
fann_destroy(ann);
return 0;
}
```

Файл xor.data, использующий для обучения xor функции:

```
4 2 1
– 1 – 1
– 1
– 1 1
1
1 – 1
1
1 1
– 1
```

Первая строка состоит из трех цифр: первое число – пары обучения в файле, второе – число входов, а третье – число выходов. Другая часть файла есть не что иное как данные обучения, состоящие из одной линии с входами и выходами. Результат выполнения обучения приведем в табл. 2.

ТАБЛИЦА 2

Max epochs	500000	Desired error	0.0010000000	
Epochs	1	Current error	0.2500359416	Bit fail 4
Epochs	27	Current error	0.0009003473	Bit fail 0

Как видно в моем случае, сеть научилась отвечать с погрешностью 0.01 за 27 циклов. Current error – сообщает нам о состоянии обучения нашей сети, если погрешность не меняется, то нужно совершенствовать нейронную сеть. Bit fail – сообщает о том, что после 27 цикла сеть перестала делать ошибки в выборке.

Выполнение: в следующем примере показана простая программа, выполняющая один проход.

```
#include
#include "floatfann.h"
int main()
{
   fann_type *calc_out;
   fann_type input[2];
   struct fann *ann = fann_create_from_file("xor_float.net");
   input[0] = –1;
   input[1] = 1;
   calc_out = fann_run(ann, input);
   printf("xor test (%f,%f) –> %f\n", input[0], input[1], calc_out[0]);
   fann_destroy(ann);
   return 0;
}
```

Результат программы выполнения приведем в табл. 3.

ТАБЛИЦА 3

Xor test	(1.000000, 1.000000)	→	0.920417	1

В итоге наша нейронная сеть выдала нам результат, который показывает, что при входных параметрах 1 и 1 с вероятностью 92 % будет 1. В этом маленьком демонстрационном примере автор хотел сказать, что с развитием вычислительной техники математики могут создавать более эффективные алгоритмы машинного обучения и внедрять их на практике.

Выводы. В результате вычислительного эксперимента показано преимущества распараллеливания обработки больших данных на кластере СКИТ-3.

Я.О. Тупало

МЕТОДИ ОБРОБКИ ВЕЛИКИХ ДАНИХ

Наведено причини виникнення великих даних та методи їх обробки, а також практичний приклад застосування методів обробки великих даних з використанням розподіленої обчислювальної системи.

Y.O. Tupalo

THE METHODS FOR BIG DATA PROCESSING

The origin of the big data problem, the methods for data processing, and practical example of applying the methods of big data processing with the use of a distributed computing system are given.

Список литературы

1. The Zettabyte Era: Trends and Analysis (https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html)
2. Книга IBM о больших данных (https://www.ibm.com/analytics/)
3. Статья краткая история больших данных (Джил Пресс (Gil Press), Forbes, май 2013 г.) (https://www.forbes.com/sites/gilpress/2013/05/09/a-very-short-history-of-big-data/#caa8ee465a18)
4. Революционные методы решения проблем долговечности больших данных (https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/big-data-amplidata-storage-paper.pdf)
5. Machine Learning Algorithms: A reference guide to popular algorithms for data science and machine learning by Giuseppe Bonaccorso.
6. Головинский А.Л., Маленко А.Л., Сергиенко И.В., Тульчинский В.Г. Энергоэффективный суперкомпьютер СКИТ-4. *Вестн. НАН Украины.* 2013. № 2. С. 50 – 59.

Получено 06.11.2018

Об авторе:

Тупало Ярослав Олегович,
научный сотрудник
Института кибернетики имени В.М. Глушкова НАН Украины.