Обучение нейронных сетей по распределенным данным

А. Н. Рукавицын[1], И. И. Холод[2], А. В. Шоров[3]
Санкт-Петербургский государственный электротехнический университет
«ЛЭТИ» им. В.И. Ульянова (Ленина)
[1]rkvtsn@gmail.com, [2]iiholod@mail.ru, [3]ashxz@mail.ru

***Аннотация*. В докладе рассматриваются подходы обучения нейронных сетей по распределенным данным. Рассматриваются особенности построения нейронных сетей в зависимости от способа распределения данных (горизонтального или вертикального). Распределение данных характерно как для информационных систем, так и для систем интернета вещей, когда об одном и том же объекте информация собирается множеством устройств. Описываются основные проблемы обучения, связанные с необходимостью объединения нейронных сетей, построенных распределено. Предлагаются принципы их решения.**

Ключевые слова: нейронные сети; распределенный анализ; распределенные данные

I. Введение

В настоящее время широкое применение получили глубокие нейронные сети всех основных задач интеллектуального анализа данных. Основной причиной массового использования является высокая точность нейронных сетей по сравнению с традиционными методами на основе статистики и machine learning (ML). Кроме преимуществ существуют и недостатки глубоких нейронных сетей, одним из основных можно выделить необходимость в высокой производительности вычислительных устройств. Процесс обучения требует больших временных и вычислительных ресурсов, что и затрудняло использование нейронных сетей ранее. На текущий момент появилось множество способов снизить время обучения с использованием распределенного или параллельного вычисления на основе многоядерных компьютеров, GPU и кластеров.

С развитием Интернета и появлением IoT острым вопросом стало использование нейронных сетей, на основе распределенных BigData. В случае IoT источниками данных является множество распределенных устройств. Это могут быть различные устройства: датчики, мобильные телефоны, компьютеры, автомобили, домашняя бытовая техника и т.д. При этом использование существующих методов распределённого вычисления зачастую сводится к распределению целевого источника, то есть объединению всех источников и дальнейшему распределению на основе выбранного метода. Также возможны условия работы сложных систем, где нет возможности передачи всех данных на один источник по причине высокой частоты передачи и большого объема данных, что может превысить максимальную нагрузку локальной вычислительной сети.

II. Распределенное обучение нейронных сетей

В общем виде можно описать нейронную сеть (рис. 1) как множество слоев L, которые состоят из множества нейронов n.

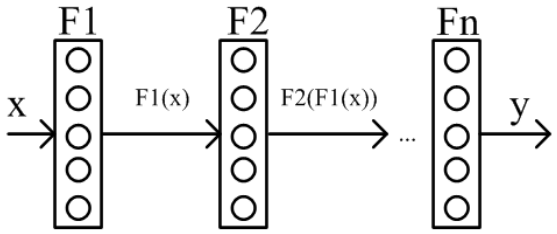

Рис. 1. Схема формального представления нейронной сети с несколькими слоями

Каждый слой L можно представить, как некоторую функцию F, которая берет в качестве аргумента значение предыдущей функции:

$$F(x) = F_i(F_i - 1(x_i - 1)),$$

где x – вектор входных данных.

Нейрон [1] получает сигнал от нейронов предыдущего слоя. Каждый сигнал умножается на весовой коэффициент w. Взвешенные входы суммируются и проводятся через блок нелинейного преобразования, который кодирует значения сумматора по некоторой функции преобразования (log, relu и тд.). Выходы лимитирующей функции передаются всем нейронам следующего слоя (рис. 2).

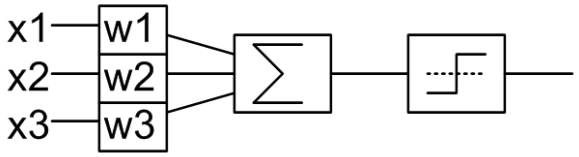

Рис. 2. Схема формального представления нейронна нейронной сети

Введем формальную модель распределенного обучения нейронной сети, где W_g – глобальный параметр весов, F –

функция получения глобального веса, p – количество узлов, W_i – полученный параметр весов на i-ом узле [2, 3, 4], тогда:

$$W_g = F_{i=1,p}(W_i)$$

На данный момент существует ряд методов для оптимизации глобальных экстремумов в нейронных сетях. При этом основным для обучения нейронных сетей априори [2] стал алгоритм градиентного спуска и его улучшенные аналоги (например, асинхронный градиентный спуск).

Для градиентного спуска можно выделить два подхода распределенного обучения нейронных сетей [2]: обучение data-parallel и model-parallel.

С точки зрения ML, источник данных представляет собой массив векторов, где каждый вектор состоит из атрибутов. Атрибутам соответствуют колонки, а векторам – строки источника данных. При обучении model-parallel на разные модели используется один вектор данных, который дублируется на вычислительные узлы для расчета разных параметров нейронной сети. Для data-parallel на вычислительные узлы отправляются данные в распределенном виде по векторам.

Распределенное обучение model-parallel схематично представлено (рис. 3) в виде двух вычислительных узлов N1, N2 и источника данных с векторами x1 и x2.

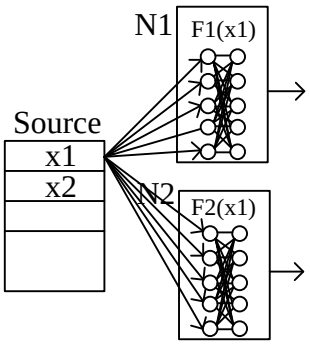

Рис. 3. Схема model-parallel обучения, где вектора данных: x1, x2, вычислительные узлы: N1, N2, нейронные сети: F1, F2

Каждое ядро отвечает за оценку различных параметров нейронной сети. Затем ядра меняют свою оценку друг с другом, чтобы получить правильную оценку для всех параметров. Формально обучение model-parallel можно представить, как:

$$W_1 = F_1(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ 0,\ 0\}$$
$$W_2 = F_2(x_1),\ x_1 = \{\ 0,\ 0,\ 0,\ x_{14},\ x_{15}\}$$

В случае распределенного обучения по схеме data-parallel (рис. 4) алгоритм разделяет данные между различными ядрами. Каждое ядро пытается независимо оценить каждый параметр одной и той же нейронной сети, но на основе разных данных. Затем ядра обмениваются своими оценками друг с другом, чтобы получить более точную оценку для шага.

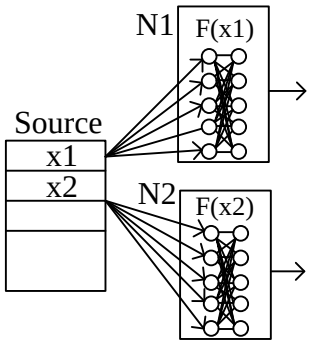

Рис. 4. Схема data-parallel обучения, где вектора данных: x1, x2, вычислительные узлы: N1, N2, нейронные сети: F1, F2

Формально обучение data-parallel можно представить, как:

$$W_1 = F(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ x_{14},\ x_{15}\}$$
$$W_2 = F(x_2),\ x_2 = \{x_{21},\ x_{22},\ x_{23},\ x_{24},\ x_{25}\}$$

Data-parallel подход полезен, когда в кластере меньше узлов, а количество оцениваемых параметров нейронной сети невелико, тогда как model-parallel подход полезен в обратном случае.

Распределение данных по источникам может выполнятся горизонтально и вертикально. Если представить данные в виде двухмерной таблицы, тогда горизонтальное распределение – это разделение таблицы по строкам или группировке строк на несколько источников. Вертикальное распределение – разделение таблицы по колонкам или группе колонок на несколько источников данных.

При распределенном хранении, данные, разделенные между узлами-источниками, представляют собой части матрицы-данных d:

$$d = d_1 \cup d_2 \cup \ldots \cup d_h \cup \ldots \cup d_w,$$

где подматрица-данных d_h размещается на узле h-м источнике.

Соответственно каждая подматрица-данных d_h характеризуется своим набором атрибутов A_h и набором векторов X_h. В этом случае возможны следующие варианты распределения матрицы данных:

- с горизонтальным распределением данных:

$$d_1 = \left(x_{j.k}\right)_{j=1,k=1}^{y,p},\ d_2 = \left(x_{j.k}\right)_{j=y+1,k=1}^{x,p},\ \ldots,\ d_w = \left(x_{j.k}\right)_{j=x+1,k=1}^{z,p}$$

- с вертикальным распределением данных:

$$d_1 = \left(x_{j.k}\right)_{j=1,k=1}^{z,g},\ d_2 = \left(x_{j.k}\right)_{j=1,k=g+1}^{z,r},\ \ldots,\ d_w = \left(x_{j.k}\right)_{j=1,k=r+1}^{z,p}$$

Рассмотрим разные схемы обучения нейронных сетей на распределенных данных по горизонтали и вертикали: data-parallel vertical (DV), data-parallel horizontal (DH), model-parallel horizontal (MH), model-parallel vertical (MV).

В варианте DH получаем:

$$W_1 = F(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ x_{14},\ x_{15}\}$$
$$W_2 = F(x_2),\ x_2 = \{x_{21},\ x_{22},\ x_{23},\ x_{24},\ x_{25}\}$$

Формальная модель не изменилась, так как данные распределены по векторам. Соответственно, для горизонтального распределения данных может быть использован подход data-parallel.

Рассмотрим следующий вариант DV:

$$W_1 = F(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ 0,\ 0\}$$
$$W_2 = F(x_2),\ x_2 = \{0,\ 0,\ 0\ ,\ x_{24},\ x_{25}\}$$

В этом случае на узлы поступает неполный вектор данных. Это означает, что обучение будет выполняться для каждой модели по разным атрибутам, что может привести к значительному снижению точности модели. Кроме того, не вся выборка данных будет использована.

При варианте MH:

$$W_1 = F_1(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ x_{14},\ x_{15}\}$$
$$W_2 = F_2(x_1),\ x_1 = \{\ 0,\ 0,\ 0,\ 0,\ 0\}$$

В таком случае передаваемый вектор для 2 узла будет пустым, так как отсутствует на втором источнике данных, если не выполнить его дублирование. Так как для каждого следующего вектора один будет получать нулевой вектор. Это означает снижение эффективности вычисления, так как один из узлов будет простаивать. Кроме того, нулевой вектор может добавлять шум при обучении модели.

Вариант MV имеет следующую ситуацию:

$$W_1 = F_1(x_1),\ x_1 = \{x_{11},\ x_{12},\ x_{13},\ 0,\ 0\}$$
$$W_2 = F_2(x_1),\ x_1 = \{\ 0,\ 0,\ 0,\ x_{14},\ x_{15}\}$$

Обучение выполняется по неполным векторам данных. При таком условии точность модели может снизиться, а также возможно появление шумов и выбросов из-за нулевых значений.

Таким образом, при схемах model-parallel в случае распределении данных вертикально, выполнение обучения по формальной модели является затруднительной задачей. При вертикальном распределении атрибуты разделены на разные источники данных. Количество входов у нейронных сетей обычно определяется количеством атрибутов, что приводит к ситуации, когда на вход модели могут поступать нулевые значения или модели могут отличаться количеством входов. Это приводит к снижению точности модели или к задаче объединения нейронных сетей с различными топологиями.

III. Возможные варианты решения

Для обучения, на вертикально распределенных данных в основном используется несколько нейронных сетей для каждого из источников [5, 6] и последующего объединения в единую модель – ensemble. Такой вариант не рассматривает подходы model-parallel или data-parallel. Нейронные сети обучаются для каждого из источников, а их значения передаются в decision function, которая выдает результат анализа данных (рис. 5).

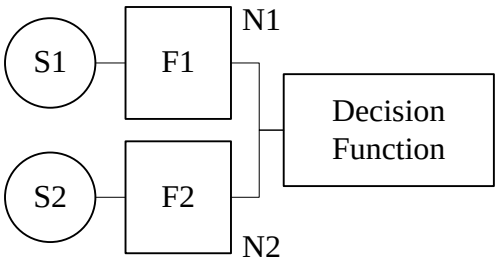

Рис. 5. Схема ensemble обучения. где источники данных: S1, S2, нейронные сети: F1, F2 на узлах: N1, N2

В качестве decision function может выступать любой алгоритм выбора верного ответа, например, voting, слой нейронной сети или нейронная сеть. В таком случае можно построить обучение на основе множества нейронных сетей на соответствующих вычислительных узлах, где выходные значения для одних будут входными для других. В качестве улучшения такого способа обучения может быть создание «распределенной» нейронной сети с не полносвязными слоями, в таком случае для источников данных будут установлены «распределенные слой», которые в дальнейшем могут объединяться на разных этапах передачи данных (рис. 6).

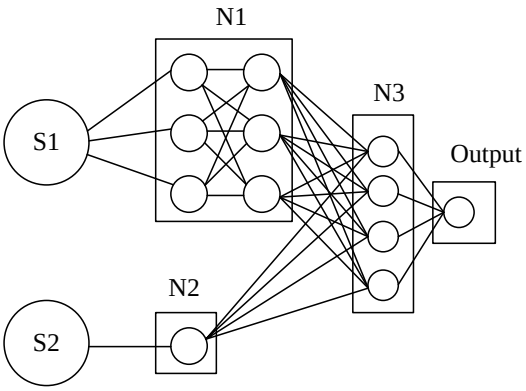

Рис. 6. Схема «распределенной» нейронной сети с не полносвязными слоями, где N1, N2, N3 – вычислительные узлы, S1,S2 – источники

В случае горизонтального распределения очевидно использование data-parallel. При вертикальном распределении данных возможно использование model-parallel с изменением функции получения W_b нейронной сети. Или созданием методики распределения данных по моделям нейронных сетей и дальнейшим объединением (merge) сетей на основе их параметров. В данном направлении будет вестись дальнейшее исследование.

Заключение

В работе рассмотрены основные подходы для распределенного обучения нейронных сетей на основе стохастического градиентного спуска: data-parallel и

model-parallel. Выявлены существующие сложности для применения на распределенных BigData при вертикальном распределении данных, которые могут встречаться в сложных системах космонавтики, энергетики, IoT и тд. Проблема получения глобальных весов при использовании моделей разной топологии на вычислительных узлах или при обучении на неполных данных, что может повлечь к критическому снижение точности.

Описаны и предложены возможные методы решения сложности обучения нейронных сетей на вертикально распределенных данных с использованием ensemble и на основе model-parallel с объединением нейронных сетей.

Список литературы

[1] McCulloch W. S., Pitts W. A logical calculus of the ideas immanent in nervous activity // The bulletin of mathematical biophysics. 1943, V. 5, Issue 4, p. 155-133.

[2] Hegde V., Usmani S. Parallel and Distributed Deep Learning // Stanford University, 2016.

[3] Chen K, Huo Q. Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering // Conference: Conference: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2016.

[4] Gupta S., Zhang W., Milthrope J. Model accuracy and runtime tradeoff in distributed deep learning. // arXiv preprint arXiv:1509.04210, 2015.

[5] Park E., Han X. Combining Multiple Sources of Knowledge in Deep CNNs for Action Recognition. // Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on, 7-10 March 2016.

[6] Christodoulidis S., Anthimopoulos M. Multi-source Transfer Learning with Convolutional Neural Networks for Lung Pattern Analysis. // IEEE Journal of Biomedical and Health Informatics, V. 21, Issue 1, 2017, p. 76-84.